Modelling collaborative business services

Thanh Thoa Pham Thi*
Dublin City University,
Glasnevin, Dublin 9, Ireland
E-mail: thoa.pham@computing.dcu.ie
*Corresponding author

Thang Le Dinh
Université du Québec à Trois-Rivières,
C.P. 500, Trois-Rivières,
Québec, G9A 5H7, Canada
E-mail: Thang.Ledinh@uqtr.ca

Markus Helfert
Dublin City University,
Glasnevin, Dublin 9, Ireland
E-mail: markus.helfert@computing.dcu.ie

Abstract: Collaborative business services represent the collaboration of business service domains across organisations. This collaboration is characterised by the interdependencies of business activities and business resources between business partners. These interdependencies require information and process sharing within constraints to ensure consistency across organisations. Most approaches for collaborative business service analyses fall into two categories: 1) those based on business-process modelling perspectives using a business process modelling language; 2) those based on collaboration patterns such as network topology or interaction patterns. However, no studies examine the connections between the two categories even though collaboration patterns influence business-process management. In this paper, we investigate the sharing of different aspects of business objects such as data value and behaviours in the context of collaborative services. We formalise this approach with a collaboration framework, and present constraints emerged. To implement the framework, we develop a model of collaborative business services and evaluate its effectiveness.

Keywords: collaborative business service; collaborative business process modelling; service modelling; sharing data; sharing process; authorisation constraint; object behaviour.


Biographical notes: Thanh Thoa Pham Thi is currently a Postdoctoral Researcher at Dublin Institute of Technology, Ireland. Previously, she was a Postdoctoral Researcher at Dublin City University, where she realised this work. She received her PhD in Information Systems from the University of...
Geneva. Her research spans the domains of innovative methods for information systems engineering, business process and business rules management, information quality analysis, business services and web services analysis and development.

Thang Le Dinh is currently a Professor in Information Systems at the Université du Québec à Trois-Rivières in Canada. He received his PhD in Information Systems from the University of Geneva in 2004. He has long experience in teaching, training and research on the areas of information systems, knowledge management and service science at the University of Sciences at HoChiMinh city, Vietnam, University of Geneva, Switzerland and University of Moncton, Canada. His research focuses on the design and the management of collaborative business services, especially information knowledge-intensive services in a network environment.

Markus Helfert is Lecturer in Information Systems and Head of Business Informatics Group in School of Computing, Dublin City University. He received his PhD in Information Management from the University of St. Gallen, Switzerland and Master in Business Informatics from the University Mannheim, Germany. His research is centred on information quality management, service science, cloud computing and data analytics.

This paper is a revised and expanded version of a paper entitled ‘Modelling collaborative services: the COSEMO model’ presented at International Conference on Software Technologies (ICSOFT), Athens, July 2010.

1 Introduction

Business organisations are increasingly seeking collaboration as a solution to sustain business in highly competitive environments. For instance, they often apply supply chain models or form alliances with partners. Collaboration needs are indicated by resource dependencies/interdependencies, activities or competences between organisations. From the business perspective, services or business services are “the application of competences for the benefit of another, meaning that service is a kind of action, performance or promise that is exchanged for value between provider and client” (Spohrer et al., 2007).

Through collaboration, various business services combine to create new value-added services called collaborative business services. In such collaborations, agreements between organisations are set, deployed and monitored. Mutual agreements between service stakeholders on functional and non-functional aspects of a service are specified in service-level agreement (SLA) documents (Yan et al., 2007) or business contracts (BPMN, 2009). These agreements are referenced for business process modelling or workflow enactment and execution (Yan et al., 2007).

One aspect related to collaboration agreements is resource interdependency and business activities between service stakeholders. A business object belongs to resources that may be a type of intelligible entity inside the business layer or tangible product. These interdependencies are characterised by sharing information and process synchronisation. Consequently, they must be described within workflow or business
process modelling. Most approaches to collaborative business service analyses fall into two streams:

1. those based on business process modelling perspectives using business process modelling language
2. those based on collaboration patterns such as network topology or interaction patterns studies.

Regarding approaches based on business processes, the collaborative business services modelling deals primarily with inter-organisational business process modelling such as business process modelling notation (BPMN, 2009), Petri Nets and UML (Gou et al., 2000; Grossmann et al., 2008; Ryu and Yucesan, 2007; Touzi et al., 2009). These approaches describe information sharing between service partners with message exchanges, which are considered as static objects. Implementing message-oriented middleware platforms, message interaction protocols enable and control the conversation (Strano et al., 2009). To specify business relationships entirely, it is important to consider business collaboration with business orientation instead of data value exchanges alone.

A business object holds not only its value embedded in a specific structure (represented by a set of its properties), but also its behaviours. Business object behaviours are business activities available to the object; the business object participates in business activities corresponding to its behaviours. A business object in varying business contexts can adopt diverse behaviours. An organisation can decide to share the value part of its business objects or the whole business object to business partners. In addition, an organisation can coordinate its business processes with partners’ business processes. This collaboration requires shared business objects evolving their behaviours into a new business context: a business network context. When adapting current behaviours to new ones in a business network context, conflicts can occur that need to be controlled by integrity constraints to ensure the consistency and the adequate performance of the business process across organisations.

We believe that the degree of collaboration between service stakeholders is based not only on what processes and business objects are shared, but also at the level they are shared and how shared information is processed. For example, if only data value of a business object is shared with a business partner, the receiver can only refer to that object (i.e., there is a restriction on the receiver’s privileges). If the whole business object is shared, the object participates in business activities within the receiver’s business context.

Constraining authorisation is an aspect related to information and process sharing across organisations not addressed properly in most approaches. Authorisation constraints represent permissions of each participant concerning shared resources. Collaboration patterns and network topology in collaborations are studied separately in business process modelling, identifying the nature of collaboration but not reflecting it at the business process level; therefore, it does not facilitate its utility at later phases of service modelling.

This paper presents a new perspective on different levels of collaboration between organisations based purely on business exchanges which is formalised as a collaboration framework and a model for collaborative business service modelling. The model supports several characteristics developed in the framework to facilitate utility. The model satisfies requirements raised regarding the limitations on modelling business exchanges in extant approaches. Applying design science research methods, we observe practice and
facts, and analyse related work in the literature. We develop artefacts, including the collaborative framework and the model of collaborative business service, and design metrics for artefact evaluation. Evaluation is based on this principle: “A design artefact is complete and effective when it satisfies the requirements and constraints of the problem it was meant to solve” (Hevner et al., 2004). The rest of the paper is organised as follows: Section 2 discusses related work. Section 3 presents a framework of business service collaboration. The presentation of our model for collaborative business service modelling is followed in Section 4. Section 5 deals with the evaluation of our approach. Section 6 present our conclusions and give some future work. Finally, an Annex presents in detail the case study for illustrating our approach.

2 Related work

There are two main research streams related to collaborative business service analysis. The first one is based on the business process modelling perspective that uses business process modelling language or blueprinting. This perspective describes business activities from service production to service delivery or exchange. The second one includes topology of service networks, interaction patterns between organisations, or collaboration maturity. These aspects are a managerial layer in the collaboration perspective, which influences the business process collaboration level.

The second research stream (IEC TC 65/290/DC, 2002; Kosanke, 2005; Touzi et al., 2009) offers different levels of collaborative maturity an organisation can adopt: communicating (capable of exchanging and sharing information), open (capable of sharing business services and functionalities with others), federated (capable of working with others according to a set of collaborative processes that have common objectives and to ensure its own objectives) and interoperable (capable of working together without a special effort; partners appear as a homogeneous and seamless system).

Medlin (2003) adapts previous works to introduce the ideal coordination contexts in a service network, including market, relational and contract-hierarchy. Each coordination context relates to an organisational level and a time horizon.

Wang and Xu (2010) identified creational style, structural style and behaviour style of service architecture for service systems. Each style dedicates to certain situations of business models. The first style suits the case where providers and customers do not known each other, the second styles aims to support customers and providers to efficiently interact with each other, and the last one focuses on inner configurations of multiple behaviour units of service providers. These architectural styles adapt topologies such as linear, star, network or network with a central repository.

Even though many features of a service system are taken into account, but these styles stay at a ‘high level design decisions’ of service system design.

Tung and Yuan (2008) proposed a framework for service innovation basing on an ecological perspective. The framework raises the continuity of co-production value. The service exchange between providers and consumers are classified as commensalism, collaboration and mutualism of partnership. The mutualism relationship is mutually beneficial interactions between individuals of providers and customers. This is characterised by “the destined partnership deriving direct effect in value co-production”. Collaboration is different from mutualism by not taking on the destined partnership that directly contributes to value co-production. Commensalism is only one sided beneficial
interactions. In addition, the framework presents criteria of performance as well as how to measure performance of mutualism in e-services.

Axelsson and Wynstra (2000) investigate interaction patterns in service exchanges of various services, suggesting business exchange includes exchange processes. The elements of exchange processes include the product exchanged, the information exchanged and the financial exchanged. The interaction patterns are identified by three elements: people involved (representative involved for example purchasers, marketing representatives, product and market specialist, etc.), the priority issues in the dialogues (e.g., how the service supports various core processes, how the service will be transformed, etc.) and the factor on which trust is based. He and Yang (2007) classify collaboration patterns based on interaction topology between participants and their role in collaboration. These patterns include simple access, service propagation, joined service without an agent, and composite service with and agent. According to the authors, “the different collaboration patterns result in different consistency requirements on the policies of the prospective collaboration partners”.

Later, He et al. (2009) proposed a control access model for service propagation, which helps detect inconsistencies between policies that result from collaboration being accepted, rejected or required during negotiations to remove inconsistencies. The approach describes policies with logical statements, and using description reasoning to generate correlation between policies.

These approaches of the second stream as mentioned above describe the nature of collaboration at a highly abstract or management level, but do not focus on information exchanged during collaboration. For service-oriented application development, the nature of collaboration should be reflected concretely at the business process modelling level to support the implementation of business services.

Regarding the first research stream, BPMN, UML and Petri Nets are popular business process modelling methods/languages used for business service modelling. BPMN 2.0 (BPMN, 2009) introduced collaboration and choreography mechanisms for collaborative business process modelling. BPMN has also been adapted for collaborative business process modelling for service-oriented architecture design (Touzi et al., 2009). In BPMN, a collaboration contains two or more pools representing participants in the collaboration. The message exchange between participants is described as message flow. Activities represent tasks or sub-processes within business process. Activities flow may be categorised as sequential, conditional and exceptional within gateway types such as OR, XOR, parallel, fork or join. Data represents input/output information for activities. The message flow concept represents data exchanges or data flows between actors or participants (i.e., business units or organisations). Message flow and activities flow may be handled with start, intermediate, or end events.

Choreography defines the sequence of interactions between participants during collaboration. Choreography modelling is similar to process modelling within constructs such as gateway, event and sequence flow. Each choreography activity corresponds to an interaction/message exchange between participants. Grossmann et al. (2008) propose extensions to business process modelling languages such as Petri Nets and UML that describe various types of inter-process dependencies across organisations including triggering, enabling, cancelling and disabling dependencies. The state activity diagram UML 2.0 in this approach has been extended to enable the descriptions of some dependency situations between business processes. Nevertheless, the extended model is quite abstract; there is an equivalent presentation of Petri Nets complex for each
dependency situation, and the model does not describe privileges concerning shared data and processes.

Intentional approach to service engineering (Roland and Souveyet, 2010) tries to help business users in service composition, service publication and service discovering by adding business descriptions of software services (i.e., web services) with intentions and strategies to achieve the goals of the services. This work in-progress is interesting but it stays at a high level of collaboration description of composite service in term of intentions and goals, the details of information exchange and constraints emerged between services are not focused.

Ryu and Yucesan (2007) propose the CPM model for collaborative process modelling in cooperative manufacturing. This model is similar to the UML activity model with an enhancement of the multi-actor concept and a transformation technique from this model to Petri Nets format. This model focuses on common processes across organisations; it does not present the data concepts. It concentrates on modelling collaborative processes, omitting data sharing. An advantage of this approach is the multi-actor concept that indicates who is responsible for performing processes.

Some approaches to model service composition and orchestration for web services are based on coloured Petri Nets (Gehlot and Edupuganti, 2009) or Petri Nets (Yoo et al., 2009). Petri Nets is a bipartite graph with two nodes connected, place and transition. Place represents a resource and transition represents a process. A transition is fired if there are tokens in input locations for that transition. Firing of a transition removes tokens from input locations and adds tokens to output locations. Coloured Petri Nets allow assigning timestamps to tokens that denote the earliest times at which a token is available. These approaches describe data (as resources) sharing across organisations with various process flow controls such as sequential, parallel, choice, fork and join activities.

Based on Petri Nets, Gou et al. (2000) propose an approach for business process modelling within virtual enterprises. The authors chose three models of interoperability identified by workflow management coalition, and modelled them with Petri Nets. They include the chained model, the nested sub-process model and the parallel-synchronised model. Introducing a particular node and a transition, the approach allows modelling resource assignments and sharing between organisations, and allows modelling activity sharing and assignments, though the model is complicated.

These approaches focus only on message exchanges and do not consider the whole business context, which includes business objects, business activities and constraints. Furthermore, no study examines the connection between the first and second research streams, and any impact between them.

Recently, Kaner et al. (2011) propose a new approach for engineering of service processes based on designing simulation experiments. Performance measures categories are identified for suitable elements in a service process. The simulation experiments allow to design (choose or improve from experiments) a desirable service process based on their performance evaluation. This approach sounds interesting for collaborative service modelling; however, it mainly focuses on non-functional criteria of service process.
3 Service collaboration framework

This section presents a collaboration framework with various collaboration levels between organisations. The difference between this framework and those of the second stream is that it is based entirely on degree of business exchange between organisations, including exchanges of resources and business activities. This principle allows the framework to be connected easily at the business process modelling level. This framework is based on the interdependencies between organisations realising a collaborative service. The context of business exchange relates to resources and business activities. Resources are categorised as operant (static and embedded value) and operand resources (acting on operant resources within business activities to generate value) (Madhavaram and Hunt, 2008).

Interdependency is represented by the needs of sourcing (or sharing) resources within (or without) business activities, or accessing resources within (or without) business activities toward another. A business object belongs to resources and holds its data value (static part) and its behaviour (dynamic part). The behaviour of a business object is represented by business activities available to that object. Each business object in a business context has a specific behaviour. When a business object is shared with another business partner, its behaviour can be adapted to the new business context. An object status describes a significant condition or a state of the object in the business context. The internal behaviour of a business object includes business activities that may change data values, but does not change object status. For example, let us consider a hotel booking as a business object. Modifying the check-in date does not change booking status. The external behaviour concerns business activities that may change the object’s status. For example, cancelling a booking changes a booking to cancellation status. Figure 1 shows internal and external behaviours of a business object; S1 and S2 are object statuses.

Figure 1 Internal and external behaviour of a business object

The internal behaviour may or may not be specified, but the external behaviour must be modelled explicitly. Based on this characteristic of a business object, there are different situations for sharing resources between organisations. Systematically, combinations of sharing data and processes produce the following sharing situations, illustrated in Table 1.
<table>
<thead>
<tr>
<th>Sharing the process</th>
<th>Sharing business object data values</th>
<th>Sharing internal behaviours of business object (update attribute value)</th>
<th>Sharing external behaviours of business object</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>• The hotel shares booking process and cancel process with the agency; agency shares booking object with the hotel, and the hotel can change the status of booking</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>• The hotel shares customer data with the data quality management service to verify and standardise customer details</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>• Airline company shares flight object with travel agency for booking a flight; travel agency can change the flight status to full.</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>• Alliance of airlines shares bonus calculation process, customer data, taken flight of customers with the members. However, a member cannot modify or update the status of customer and taken flight shared with other members.</td>
</tr>
<tr>
<td>Not sharing the process</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>• Travel agency requests sightseeing service, sharing customer information and tour with the tour agency; tour agency may lightly change the tour information depending on weather, subjective conditions, etc., realising the tour and changing the tour status to completed tour</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>• The hotel requests cleaning service and shares occupied room with the cleaning service, which cleans the room and changes the room status to cleaned room</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>• The hotel requests business analytics services and shares customer information, booking information and cancellation information with an it company.</td>
</tr>
</tbody>
</table>
Four components of the business discussed above (business object data value, internal behaviour, external behaviour and business activities) are observed and analysed in various collaborative situations. Meaningful combinations of those elements are marked and form collaborative situations with concrete examples illustrated. These situations of sharing business resources can be described within authorisation constraint, depicting privileges granted to roles within participant resources during collaboration. A role may be a person, an organisational unit or a group of people or organisational units who have the same responsibilities and privileges. Resource represents data, a business activity or a process. Shared data may be business data, an event or a request that triggers business activities. A privilege describes a permission of producing or accessing resources, or performing an activity granted to a role.

Permissions defined for data include creation, modification, reference and suppression of a participant’s own data (internal data). Participants/roles can request data reference and modification permissions from other participants/roles (external data). Indication of a role attached to a permission avoids the ambiguous situation in case of n-airy data sharing (i.e., a role receives the same data class from several senders). Process permissions deal with authorisations for process performance. An owner of a process can share permissions for process performance with other participants; the latter has the responsibility for process performance, and the owner can redesign the process. Consistency control emerges due to data and process sharing between service stakeholders, ensuring correct performance of business activities across organisations; constraints enforce consistency depending on collaboration levels between service stakeholders.

From the sharing-resources situations described in Table 1, we develop corresponding collaboration levels within their driver constraints (Table 2).

<table>
<thead>
<tr>
<th>Business object behaviours</th>
<th>Business object data value</th>
<th>Business activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very tight collaboration (level 1)</td>
<td>Tight collaboration (level 2)</td>
<td>Loose collaboration (level 3)</td>
</tr>
<tr>
<td>Very loose collaboration (level 4)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 3.1 Level 1 – very tight collaboration

Level 1 corresponds to sharing the whole business object and sharing processes corresponding to object behaviours. This level allows the receiver to update shared data and change object statuses. For example, a hotel collaborates with a travel agency while providing booking services to customers. The hotel shares booking processes and the cancel process to the agency. Customers can contact the travel agency for both booking and cancellation, or they can contact the hotel directly to make or cancel a booking. The hotel can update or cancel a booking made with the hotel, but a booking made with the agency can be updated or cancelled by the agency or the hotel. The hotel shares information on the number of vacant room with the agency, and the agency shares whole business objects concerning bookings made with the agency to the hotel such as customers, booking, and cancelled booking business objects. Figure 2 depicts the business processes and information from each partner related to this example. The hotel
is the receiver of customers, booking and cancelled booking business objects, shared by the agency and described with data flow from the agency to the hotel. The behaviours of customer and booking objects are also shared with the hotel (the flow with update allowed). The hotel has its own customer, booking and cancelled booking business objects. Meanwhile, the number of vacant room information is owned solely by the hotel and shared with the agency. The flow of shared information comes from number of vacant rooms in the hotel and goes to the process book in the agency.

Figure 2 An example of collaborative level 1 services (see online version for colours)

At this level, constraints to the process defined at the sender’s site are reproduced at the receiver’s site. Constraints on input/output data applied to the business process as its pre/post-condition have to be considered; constraints to object statuses across organisations can also occur. For example, the business logic at the hotel site requires exclusive constraint between the check-in process and the cancel process; if a booking is cancelled, the customer cannot check-in to the hotel with the booking. When a booking is cancelled, it cannot participate in the check-in process, and becomes inactivate. An inactivate data object cannot participate in the process to change status and behaviour. Since the cancel process is shared with the agency, this constraint is reproduced in the cancel process at the agency. If a cancelled booking occurs at the agency, the hotel must be informed. This collaboration is presented by a sharing of data flow. When the cancelled booking is received, the correspondent booking becomes inactive by business logic; thus, the check-in process cannot be realised.

3.2 Level 2 – tight collaboration

This level corresponds to sharing only the data part of a business object and the business processes that refer to this data value. The receiver can refer to the data but cannot update data or change object status.
For example, an airline alliance shares the process of *bonus calculation* for loyal clients (Figure 3). Shared data includes information on flights taken, miles flown and customer demographics. An airline cannot modify the shared information owned (or created) by other airlines.

**Figure 3** Example of collaborative level 2 services (see online version for colours)

This level is concerned with sharing the data part of business objects and sharing business activities. Due to shared business activities, constraints on pre/post-conditions of the process are also considered. For example, in cases where airline companies share common *bonus calculation* processes, the post-condition of the *bonus calculation* process related to accumulative miles of the current client (e.g., set to zero after redeeming a free ticket) is reproduced for each partner.

### 3.3 Level 3 – loose collaboration

This level corresponds to sharing whole business objects but not sharing business processes involved in the sender’s business context. The receiver can update object statuses with business activities in the receiver’s business context. For example, a hotel collaborates with a cleaning agency to request daily cleaning service (Figure 4). The hotel provides information of *occupied room* to the cleaning agency. The *daily clean* process is performed directly on the shared information *occupied room* and changes the status to *cleaned room*.

**Figure 4** Example of collaborative level 3 services (see online version for colours)
This level deals with sharing business object behaviour and data. An object’s behaviour at the sender’s site must adapt to the object’s behaviour at the receiver’s site. Consequently, constraints on object status across organisations must be considered, illustrated in Figure 4. At the business logic level, if the check-out process at the hotel’s site occurs, the daily clean process at the cleaning service site is no longer available. This does not represent an exclusive constraint on processes within two organisations; there is a constraint: “For every room after the check-out process, the daily clean room process is disabled, but it is not the case to the contrary”.

3.4 Level 4 – very loose collaboration

This level is concerned with sharing data values only. The receiver can refer to shared data for business activities, with the sharing representing a trigger for processes at the receiver’s site. For example, the hotel conducts a customer service survey to improve service (Figure 5). The hotel sends this request to the travel agency and shares check-out customer information with the travel agency. The travel agency refers to this information and sends an invitation to customers. This level only allows receivers to refer to shared data; no constraint applies across organisations because there is no process performed and no object status changed.

Figure 5 Example of collaborative level 4 services (see online version for colours)

4 The COSEMO model

In the previous sections, we present different collaboration levels between business partners. Consequently, diverse constraints across organisations emerge. In this section, we present a model for collaborative business service modelling that describes thoroughly all collaboration levels and constraints as mentioned above.

Collaborative service conceptual modelling (COSEMO) uses some concepts from the IASDO model (Pham Thi et al., 2006; Pham Thi and Helfert, 2007) with extensions on authorisation constraints. A process can transform information from a state (e.g., input information) to another state (e.g., output information). There are two possibilities: an
exclusion of these two states in the business context (e.g., occupied room and vacant room) or there is no exclusion (e.g., occupied room and cleaned room). If there is no exclusion, information validity remains in both states; otherwise input information becomes invalid when output information is created. These situations can be described with tokens as in the Petri Nets. A token represents an object. A token is removed from occupied room, and a new token is added to vacant room after initiating the check-out process. In this case, a token in occupied room remains and a token is added to cleaned room after initiating the daily clean process, which cannot be described naturally as business logic with the Petri Nets.

The COSEMO model allows users to describe control flows based on business logic using AND, OR, and XOR gates. This control is, as business rules define, at the process level in the form of process pre- and post-conditions, and defined at the data object level in the form of remaining state/leaving object states. The COSEMO model includes a set of concepts, functions and privileges. These include the following concepts: classes, business activities or processes, and organisational roles:

- C: represents a set of classes C of information objects created, transformed during the service realisation. C may also be a significant dynamic state of an information object, transformed from another state by performing a process.
- P: a set of business activities or processes P involved in the service. A process creates output information and/or transforms input information to its new state as output information. Every process P has its pre- and post-conditions that defines rules on input/output data, particularly in cases of multi-input classes and/or multi-output classes.
- R: a set of organisational roles or organisations (service stakeholders) involved in the collaborative service.

To work with these concepts, there is a need for the following functions related to input information, output information and remaining information:

- \( f_i: (C, P) \rightarrow \{0, 1\} \), function of input information, if \( f_i(c, p) = 1 \), \( c \in C, p \in P \), then \( c \) is an input class of \( p \)
- \( f_o: (P, C) \rightarrow \{0, 1\} \), function of output information, if \( f_o(p, c) = 1 \), \( c \in C, p \in P \), then \( c \) is an output class of \( p \)
- \( f_{\text{remain}}: (C, P, C) \rightarrow \{0, 1\} \), if \( f_{\text{remain}}(c_1, p, c_2) = 1 \), \( f_i(c_1, p) = 1 \), \( f_o(p, c_2) = 1 \) then the \( c_1 \) state is remained when \( c_1 \) is transformed to \( c_2 \) state through \( p \) realisation, otherwise if \( f_{\text{remain}}(c_1, p, c_2) = 0 \), \( f_i(c_1, p) = 1 \), \( f_o(p, c_2) = 1 \) then the \( c_1 \) state transforms to the \( c_2 \) state.

Finally, there are privileges such as owner privileges for the process and data:

- \( p_{\text{privilege}}: (R, P) \rightarrow \{\text{owner, responsibility}\} \), defining roles as owner of a process or in charge of a process. If \( (r, p) = \text{owner} \), then \( r \) is an owner of the \( p \) process. Normally, an organisation owns a process. In a collaborative context, several organisations may own the process. The owner is the one who develops the process, while responsibility describes a delegate or sharing situation from the owner to other roles.
c_privilege: (R, C) \rightarrow \{creation, modification, reference, suppression, modification'_{(Role)}, reference'_{(Role)}\}, assigning data privileges to roles. 
Creation (C), modification (M), reference (R), and suppression (S) are internal privileges applied to information created by a role. Modification'_{(Role)}, reference'_{(Role)} \text{ and } suppression'_{(Role)} \text{ are external privileges applied to information created by other roles. For example, if } c\text{ _privilege}(r_1, c) = \{C, M, M'_{(r_2)}\}, \text{ then role } r_1 \text{ can be created and modified instances of class } c \text{ create themselves; } r_1 \text{ can modify instances of } c \text{ created by role } r_2.

At the graphical level, there is a replication of common business objects and processes between organisations such as customer, booking, and book process. The COSEMO model interprets them as one business object, and one business process encompasses privileges granted to multiple participants. Table 3 presents the specifications of four collaboration levels in the COSEMO model:

**Table 3** Specifications of four collaboration levels in the COSEMO model

<table>
<thead>
<tr>
<th>Collaboration level</th>
<th>Specification with the COSEMO model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Level 1 – very tight collaboration</strong></td>
<td>p_privilege(r1, p) = owner, p_privilege(r2, p) = responsibility, c_privilege(r1, c) = {C, M, R, S, M'_{(r_2)} }, c_privilege(r2, c) = {C, M, R, S}</td>
</tr>
<tr>
<td></td>
<td>p_privilege(r1, p) = owner, p_privilege(r2, p) = owner, c_privilege(r1, c) = {C, M, R, S, R'_{(r_2)} }, c_privilege(r2, c) = {C, M, R, S}</td>
</tr>
<tr>
<td><strong>Level 2 – tight collaboration</strong></td>
<td>c_privilege (r1, c) = {C, M, S, R}, c_privilege (r2, c) = {R'<em>{(r_1)}, M'</em>{(r_1)}}</td>
</tr>
<tr>
<td><strong>Level 3 – loose collaboration</strong></td>
<td>c_privilege(r1, c) = {C, M, S, R}, c_privilege(r2, c) = {R'<em>{(r_1)}, M'</em>{(r_1)}}</td>
</tr>
<tr>
<td><strong>Level 4 – very loose collaboration</strong></td>
<td>c_privilege(r1, c) = {C, M, S, R}, c_privilege(r2, c) = {R'_{(r_1)}}</td>
</tr>
</tbody>
</table>

The meta-model of COSEMO with UML class diagram notations is presented in Figure 6. The dynamic state class describes information in a specific state, significant to the business process; the modeller decides this. Output data flow class describes an output class of a process, while the input data flow class describes an input class of a process.

In addition, COSEMO includes constraints in its meta-model. The modeller should respect these constraints to guarantee the coherence of specifications. These constraints are presented in Table 4.
Figure 6 Meta-model of the COSEMO model

Table 4 Constraints on modelling with COSEMO at the meta-model level

<table>
<thead>
<tr>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(C1)</td>
<td>Classes in input data flow and output data flow class involved in the remaining state class must belong to dynamic state class</td>
</tr>
<tr>
<td>(C2)</td>
<td>If role r has the creation privilege on class c, it also has reference privilege on c</td>
</tr>
<tr>
<td>(C3)</td>
<td>If role r has owner or responsibility privileges on process p, r has at least reference privilege on all input classes p</td>
</tr>
<tr>
<td>(C4)</td>
<td>If role r has owner or responsibility privileges on process p, r has creation privilege on all output classes p</td>
</tr>
<tr>
<td>(C5)</td>
<td>If role r has owner privilege on process p, r has reference+ privilege on all input and output classes for p</td>
</tr>
</tbody>
</table>

5 Evaluation

Development of the COSEMO framework was based on the observations of various data sharing and process situations between organisations. Four levels present a systemic combination of sharing business object components and sharing processes that have significance for situations that exist in the real world (Table 1). A mix of collaborations
between two organisations may be required due to various importance levels placed on business processes. The COSEMO model is a means of implementing this framework. Our work follows the design science methodology in which “evaluation requires the development of metrics and the measurement of artefacts according to those metrics. Metrics define what we are trying to accomplish. They are used to assess the performance of an artefact. Lack of metrics and failure to measure artefact performance according to established criteria result in an inability to effectively judge research efforts” (March and Smith, 1995).

Validation of the framework is based on the logic acquired when sharing business objects and business activities through observations and real-world examples. In the future, validation will be made with a survey. Regarding the evaluation of the COSEMO model, we use the following metrics: description power, consistency and utility. These metrics reflect the requirements raised in collaborative business service modelling as argued earlier in this paper. Measurement of these metrics is subjective evaluation based on case study. One of the methods includes comparing proposed with existing artefacts based on the metrics. Below we compare our model with current approaches to collaborative service modelling.

5.1 Business process modelling notation

BPMN allows users to describe collaborative business processes graphically with collaboration and choreography concepts. The main collaboration technique is message exchange between organisations. Our approach allows describing business object sharing, including data and its behaviours, leading to complete recognition of constraints across organisations. Our modelling of object remaining/leaving states and authorisation constraints allows us to describe multiple collaboration levels and constraints across organisation not available with BPMN.

5.2 Petri Nets

Petri Nets approaches do not present authorisation on data and processes for modelling like in our collaboration framework. The Petri Nets model proposes process flow controls with AND/XOR-split/join but not an OR-split/join. Therefore, the specifications described in Figure 4 need additional transformations if modelled with Petri Nets. The additional information may not be necessary at the business level. Also with Petri Nets modelling, after initiating a process, tokens in input locations are removed.

In a business context, if a token in a location represents the status of data objects in a correspondent state, a token is not always removed after taking part in a transition. For example, an occupied room in a hotel after being cleaned by the cleaning service (i.e., takes part in the cleaning transition) is still occupied (remains in the occupied location), and can be cleaned again. However, an occupied room is not occupied any longer (token is removed) after taking part in the check-out transition. Otherwise, a modeller needs some transformations to conform to Petri Nets standards, impractical for service modelling at the business level.

Table 5 presents the comparison between different approaches for collaborative business service modelling.
Table 5  Comparisons between different approaches based on the proposed metrics

<table>
<thead>
<tr>
<th>Business service modelling used with</th>
<th>Description power</th>
<th>Consistency</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPMN</td>
<td>• Message exchange</td>
<td>• Do not control consistency across organisation</td>
<td>• Popular use</td>
</tr>
<tr>
<td></td>
<td>• Do not constrain exchanged data</td>
<td></td>
<td>• Do not facilitate service implementation as a consequence of limited description</td>
</tr>
<tr>
<td>Petri Net-based approaches</td>
<td>• Message exchange</td>
<td>• Part control consistency across organisation</td>
<td>• Do not facilitate service implementation as a consequence of limited description</td>
</tr>
<tr>
<td></td>
<td>• Do not constrain exchanged data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COSEMO</td>
<td>• Data and behaviours exchange</td>
<td>• Control consistency across organisation</td>
<td>• Allows consistent and precise modelling; facilitates technical service implementation</td>
</tr>
<tr>
<td></td>
<td>• Constrain exchanged data and behaviour</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network of services</td>
<td>Focuses on network topology or interaction patterns rather than exploring characteristics of business exchange</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This comparison suggests the effectiveness of the proposed model; it satisfies the requirements that are the limitations in the current approaches for collaborative business services modelling. COSEMO allows to implement entirely the proposed collaboration framework. Meanwhile, no studies suggest how the network topology and interaction patterns can be reflected at the business process modelling level.

6  Conclusions

Over the last few years, many aspects of service sciences – especially service-oriented computing (SOC) – have caught the attention of both researchers and industry. As a new interdisciplinary research domain, several research roadmaps were proposed. However, literature reviews demonstrate that there is more research on service sciences focusing on SOC paradigms than on business services. There is a need for a strong foundation on business service modelling and design, particularly in the network collaboration context.

To engage collaborative business services, legal documents such as SLAs and business contracts should focus on the interdependencies of information and business activities between participants. These agreements are the basis for workflow enactment or business process modelling and execution. Modelling should reflect these interdependencies precisely. Most approaches for business service modelling concentrate on data sharing with message exchange mechanisms, but do not investigate the entire scope of business object sharing. Considering business object sharing with its behaviours allows describing interdependencies more precisely regardless of collaboration patterns, verifying business performance consistencies across the organisation.

We develop a collaboration framework to address this issue that covers four collaboration levels based on degree of sharing data and processes. Consequently, constraints on data and processes emerge to ensure the consistency of data and processes.
between service providers. We propose the COSEMO model for modelling data and processes and controlling constraints across organisations.

The COSEMO model has some advantages compared to other approaches. First, it describes all collaboration levels thoroughly to ensure the consistent control of data and processes across organisations. Second, it is flexible since it is based on business logic. Third, specification of organisational role privileges for shared data and processes facilitate technical service design levels. Thus, this approach supports users during collaborative business service modelling for service-oriented application development. In the future, we deal with non-functional aspects of the collaborative business service analysis and a systematic technique for transforming business service specifications to technical service design based on this approach.

References


Modelling collaborative business services


Annex

Modelling the travelling service

This annex presents in detail the modelling of the travelling service example with the COSEMO model. Figure 7 demonstrates the travelling service.

Book process and cancel booking processes are common between the hotel and the agency; in the model, they are denoted with multi-organisational roles. Shared information between the hotel and the agency are customer, booking, cancelled booking, number of vacant rooms, and checked-out customer. Shared information between hotel and cleaning agency is occupied room. Different collaboration levels are denoted in the model.
Specification on remaining objects

<table>
<thead>
<tr>
<th>Function</th>
<th>Explication</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1 ( f_{\text{remain}}(\text{booking, cancel booking, cancelled booking}) = 0 )</td>
<td>A provision booking is not valid any more when cancelled</td>
</tr>
<tr>
<td>S2 ( f_{\text{remain}}(\text{occupied room, daily clean, cleaned room}) = 1 )</td>
<td>An occupied room remains occupied after it is cleaned with the daily clean process,</td>
</tr>
<tr>
<td>S3 ( f_{\text{remain}}(\text{vacant room, check-in, occupied room}) = 0 )</td>
<td>An available room is not available any more when it is checked-in</td>
</tr>
<tr>
<td>S4 ( f_{\text{remain}}(\text{occupied room, check-out, available room}) = 0 )</td>
<td>An occupied room is not occupied any more when it is checked-out.</td>
</tr>
</tbody>
</table>

Pre- and post-conditions of processes

| S5 | Book. Precondition = Customer AND (Number of vacant room > 0) |
| S6 | Cancel booking. Precondition = Booking AND Cancel request |
| S7 | Cancel booking. Postcondition = Cancelled booking AND Confirm of cancellation |
| S8 | Check-in. Precondition = Booking AND Available room |
| S9 | Check-in. Postcondition = Checked-in customer AND Occupied room |
| S10 | Daily-clean. Precondition = Occupied room AND Date time |

Figure 7  Modelling travelling services with the COSEMO model (see online version for colours)
Specifying authorisation constraints on business resources and remaining functions, this model covers constraints on shared data and processes across organisations mentioned in Section 3.2:

- **Constraint 1:** A booking cannot be checked-in if it is cancelled (at the hotel site or agency site) => specifications S1, S8, c_privilege (hotel, booking) = \{C, M, R, S, M^+_{(TA)}, R^+_{(TA)}\}, c_privilege (hotel, cancelled booking) = \{C, M, R, S, M^+_{(TA)}, R^+_{(TA)}\}

- **Constraint 2:** For every room, after the check-out process is performed, the daily clean room process is disabled, but not in the contrary case => S2, S4, S10, c_privilege (cleaning agency, occupied room) = \{C, M, R, S, M^+_{(Hotel)}, R^+_{(Hotel)}\}

### Specification on authorisation constraints

Details of authorisation constraint specifications are illustrated in Table 6.

#### Table 6  Authorisation constraint specification

<table>
<thead>
<tr>
<th>Data/process</th>
<th>C/owner</th>
<th>M</th>
<th>R</th>
<th>S</th>
<th>M^+</th>
<th>R^+</th>
<th>Resp.</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>M^+_{(TA)}</td>
<td>R^+_{(TA)}</td>
<td>N/A</td>
</tr>
<tr>
<td>Number of vacant rooms</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>N/A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Booking</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>M^+_{(TA)}</td>
<td>R^+_{(TA)}</td>
<td>N/A</td>
</tr>
<tr>
<td>Cancelled booking</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>Occupied room</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>R^+_{(TA)}</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Checked-out customer</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>Book</td>
<td></td>
<td>x</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Cancel booking</td>
<td></td>
<td>x</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Customer</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>M^+_{(Hotel)}</td>
<td>R^+_{(Hotel)}</td>
<td>N/A</td>
</tr>
<tr>
<td>Number of vacant rooms</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>Booking</td>
<td></td>
<td>x</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Cancelled booking</td>
<td></td>
<td>x</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Checked-out customer</td>
<td></td>
<td>x</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Occupied room</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>M^+_{(Hotel)}</td>
<td>R^+_{(Hotel)}</td>
<td>N/A</td>
<td>Cleaning service</td>
</tr>
</tbody>
</table>