Towards an Integration System for Artifact-centric Processes

Maroun Abi Assaf
University of Lyon
CNRS, INSA-Lyon, LIRIS,
UMR5205, F-69621, France
maroun.abi-assaf@insa-lyon.fr

Supervised by Prof. Youakim Badr,
Prof. Kablan Barbar,
and Prof. Youssef Amghar
Expected Graduate Date: June 2017

ABSTRACT
The last few years have seen a growing interest in the artifact-centric process modeling approach across database and business process management communities. Artifact-centric processes not only unify databases and how data tuples are processed through their lifecycles but they also provide business people with a paradigm to easily express the way business activities should evolve towards achieving business goals. This PhD thesis focuses on integrating heterogeneous artifacts from different sources in order to provide unified views for managing them. Since artifacts are complex entities composed of information models, state-based lifecycles, tasks and business rules, their integration poses a challenge and requires combining several approaches from different domains like data integration and business process merging. In this paper, we propose a design for an artifact-centric process integration system, in addition to a graphical artifact modeling notation, and an artifact query language that support the artifact integration.

Categories and Subject Descriptors
D.2.12 [**Software Engineering**]: Interoperability – *Data mapping, Distributed objects*; H.2.3 [**Database Management**]: Languages – *Query languages*, *Data description languages (DDL)*, *Data manipulation languages (DML)*; H.2.4 [**Database Management**]: Systems – *Distributed databases;* H.2.5 [**Database Management**]: Heterogeneous Databases.

General Terms
Algorithms, Management, Design, Languages, Theory.

Keywords
Business Artifact; Business Process Models; Data Integration; Query Languages; Query Reformulation; Modeling Notations.

SIGMOD'16 PhD Symp, June 25-July 01 2016, San Francisco, CA, USA

ACM 978-1-4503-4192-9/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2926693.2929904

1. INTRODUCTION
Many of today's businesses are formed by mergers and acquisitions with other businesses and, as a result, require us to deal with the problem of integrating heterogeneous business processes performing similar and semantically equivalent functionality (e.g., manufacturing processes, sales processes). A convenient approach to manage heterogeneous processes consists of a unified view to centralize the access to information and tasks manipulated by these processes through global and unified processes. The unified process of distributed processes facilitates their manipulation and increases productivity by relieving users from dealing with differences of heterogeneous processes.

Business artifact, or artifact for short, is a business process modeling approach that seeks to explicitly unify data and their processing, and consequently eliminates the dichotomy that has separated the database community and the business process management community [13]. Business artifact consists of an informational model, comprising a set of attribute-value pairs, a set of tasks that manipulate the informational model, a set of states, and a transition diagram between states, called lifecycle [8]. The lifecycle describes how artifacts evolve by specifying possible execution orders and timing by which tasks should be invoked [4]. Artifact-based processes have proliferated at a phenomenal pace over the last few years with the wide range of promising applications including finance, monitoring, and virtual organization [8, 13]. The subject of this PhD thesis is the integration of heterogeneous artifact-centric processes

Integrating artifact-centric processes raises an acute problem because of the complexity of matching and mapping two or more artifact-centric processes at the level of their components (e.g., information models, lifecycles, tasks, and association rules between tasks and information models). Traditional data integration solutions and techniques [7, 10, 14] fail to address the complexity of artifact-centric process integration. The challenges facing artifact-centric process integration can be classified into three different levels that must be dealt with in order to define a complete and working integration system. The three levels are: *Modeling level, Implementation level,* and *Integration Level.*

First, at the *modeling level*, due to the nature of business artifacts that treats data and process as a single unit, artifact-centric processes must be modeled in specialized *conceptual models* using specialized graphical notations. These conceptual models must include all the necessary information describing the artifact's components; information model, tasks, states and lifecycles. Additionally, these *conceptual models* must be defined in a way that supports the integration process since they will be integrated into a global artifact-centric process.

Second, at the *implementation level*, the conceptual models defined in the previous level must be implemented in a software system that can be used to execute, manipulate, and interrogate the heterogeneous artifact-centric processes. Since business artifacts are unusual data types, their execution, manipulation and interrogation must be conducted using a specialized query language similar to SQL. This query language is not only locally used at the heterogeneous artifact-centric processes but also used at the integrated artifact-centric process in order to dispatch and map queries to the heterogeneous processes.

Finally, at the *integration level*, the conceptual models referred to as *local models* must be integrated into a unified model referred to as *global model* that will be used as a centralized access point to the heterogeneous artifact-centric processes. Correspondences between elements of the conceptual models must first be discovered. This correspondences discovery process must support the different components of business artifacts; Information model, tasks, states, and lifecycles. Second, the global model must be generated according to the discovered correspondences. Finally, mapping specifications are generated in order to translate queries between the global model and local models.

This PhD thesis is at the start of its third year. We have extensively reviewed the literature of related research areas. We also proposed and are implementing an artifact-centric process integration system. This integration system includes a proposal for a graphical *Business Artifact Modeling Notation* (***BAMN***) used to model conceptual *Business Artifact Models* (***BAM***). It also includes a proposition for an *Artifact Query Language* (***AQL***) similar to SQL used to define, manipulate and query business artifacts. Finally, it includes integration semantics used to collect correspondences between several *Business Artifact Models*, generate a global *Business Artifact Model* and mapping specifications.

The remainder of this paper is organized as follows. Section 2 provides an overview of related work. We report our results and current progress in Section 3. Finally, Section 4 provides some concluding remarks, in addition to future and remaining work.

2. RELATED WORKS

To the best of our knowledge, the problem of artifact integration is not fully addressed in the literature [3]. Since business artifact combines data models, data processing and state-based lifecycles, their integration becomes a complex problem. Finding solutions leads to analyze integration problems in databases and business process management fields.

From the database perspective, data integration involves the integration of local schemas into a global schema and mappings which are used to translate data between local and global schemas [10, 14]. The global schema is generated based on identified correspondences between elements of local schemas. This process of identifying correspondences is known as schema matching [7, 15]. Similarly, we integrate several local business artifact models in order to build a global model and then generate transformation mappings. Since business artifact models include control elements like tasks and flow connectors, artifact integration requires different integration semantics that takes into consideration these control elements and their execution flows.

On the other hand, from the business process perspective, process integration is extensively covered to build business processes by merging other business processes [9, 16]. Nevertheless, the business process merging is primarily based on control-flows, which are not directly modeled in artifact models.

Recent work in the field of artifact-based business process modeling seeks to inter-operate heterogeneous artifacts [12]. In sum, these representative works provide a diversification in solutions that are convenient for solving the integration at the data level or at the process level without taking into account the complexity of artifact integration that has to jointly deal with data, their processing and evolution (e.g., state-based lifecycles).

3. RESULTS

This section presents our progress in this thesis and current results.

Figure 1 illustrates the artifact-centric process integration system we have designed. The system is composed of four phases: *Modeling Phase*, *Implementation Phase*, *Execution Phase*, and *Integration Phase*.

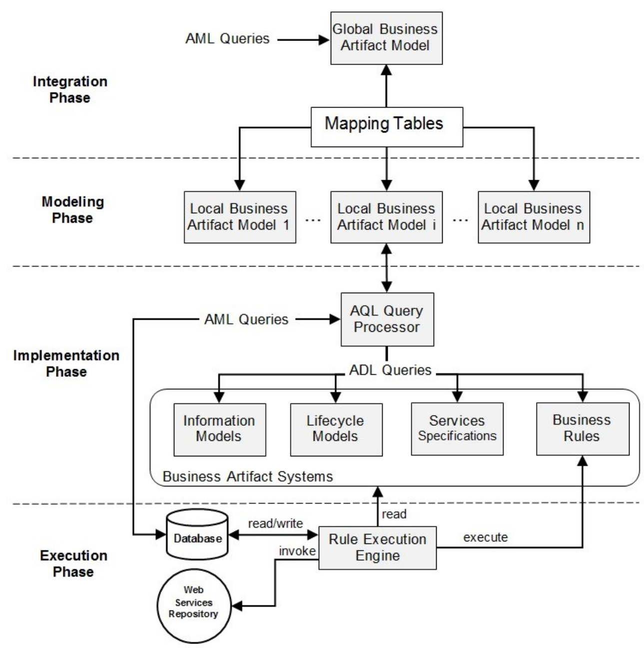


Figure 1. Artifact-centric Process Integration System

First, in the *Modeling Phase*, conceptual *Business Artifact Models* are modeled using the *Business Artifact Modeling Notation* (*BAMN*) in a graphical editor called *BA Modeler*.

Second, in the *Implementation Phase*, business artifact system specifications are generated from the conceptual models of the *Modeling Phase*. These system specifications are specified as *Artifact Definition Language* queries (*ADL* queries) of the *Artifact Query Language* (*AQL*). The *AQL Query Processor* will then execute *ADL* queries and generate corresponding *Business Artifact Systems* as *Information Models*, *Lifecycle Models*, *Services Specifications*, and *Business Rules*.

Third, in the *Execution Phase*, a *Rule Execution Engine* based on *Java Rule Engine Drools* will execute *Business Rules* of *Business Artifact Systems*, invoke services, and perform modifications to a database holding business artifact instances, thus executing the corresponding artifact-centric processes.

Finally, in the *Integration Phase*, conceptual local models of the *Modeling Phase* are integrated into a global model and mapping tables that translate between them are generated. *Artifact Manipulation Language* queries (*AML* queries) of the *Artifact Query Language* (*AQL*) can be written on the global model, then translated using the mapping tables into the local models, then finally executed using the *AQL Query Processor* in order to manipulate and interrogate distributed business artifact instances.

In the rest of this section, we discuss more the three primary phases of the artifact-centric process integration system; *Modeling Phase*, *Implementation Phase*, and *Integration Phase*.

3.1 Modeling Phase

In the *Modeling Phase*, conceptual *Business Artifact Models* (*BAMs*) are modeled in a graphical editor called *BA Modeler*. To this end, we define the *Business Artifact Modeling Notation* (*BAMN*) that the *BA Modeler* is based on.

BAMN is a minimalistic and simple set of graphical constructs that provide all necessary information required prior to generating business artifact systems. The advantages of *BAMN* over traditional activity-centric modeling notations like *UML Activity Diagrams* and *BPMN* are that it not only incorporates the data aspects of a business from the beginning, but it also relieves business people from dealing with complex Control-Flow logic and patterns like *Parallel Split*, *Synchronization*, *Exclusive Choice*, *Discriminator*… [17]. *BAMN* is based on the notation described in [11, 13] but introduces additional constructs and variations. A conceptual *Business Artifact Model* (*BAM*) is thus built by modeling interacting lifecycles of several business artifacts. To this end, the *BAMN* covers business artifact lifecycles, expressed in terms of: *Task*, *Repository*, *Flow Connector*, *Data Attributes List*, *Condition*, and *Event* constructs as summarized in Table 1.

Table 1. Business Artifact Modeling Notation BAMN

Modeling Construct	Graphical Notation	Description
Task	Task	Units of work to be performed on artifacts
Repository	Repository	Places where artifacts await future processing, if any
Read/write Flow Connector	Artifact ⟶	Transit artifacts between tasks and repositories
Read-only Flow Connector	Artifact - - - -▸	Read artifact content from a repository
Data Attribute List	Att_1 : $type_1$ Att_2 : $type_2$ … Att_n : $type_n$	Attribute-type pairs that characterize a repository
Condition	Condition	Conditions associated to flow connectors
Event	Event	Event associated to flow connectors

Tasks refer to places where change can be made to business artifacts. They correspond to units of work to be performed in order to manipulate business artifacts. *Repositories* denote places in which artifacts can be placed awaiting future processing, if any. *Repositories* represent different stages or states of business artifact lifecycles. *Flow Connectors* associate tasks and repositories; read/write *Flow Connectors* indicate that artifacts are transferable between *tasks* and *repositories* where they are manipulated and evolved with respect to their lifecycles. Read-only *Flow Connectors* indicate that business artifact content is only readable and cannot be modified, thus it remains in the same *repository*. *Data Attribute Lists* are associated to *repositories*. They allow the description of an information model by annotating the *Business Artifact Model* when business artifacts reach a certain state or stage. *Event* and *Condition* constructs are attached to *Flow Connectors* and specify situations under which *Flow Connectors* will be activated. The *Event* construct specifies an external event that is received. The *Condition* construct specifies constraints that should be satisfied in order to activate a Flow Connector.

In Figure 2, a conceptual *Business Artifact Model* illustrates part of the lifecycle of a *Candidate Application Artifact* that deals with admitting a candidate into a master program in a university.

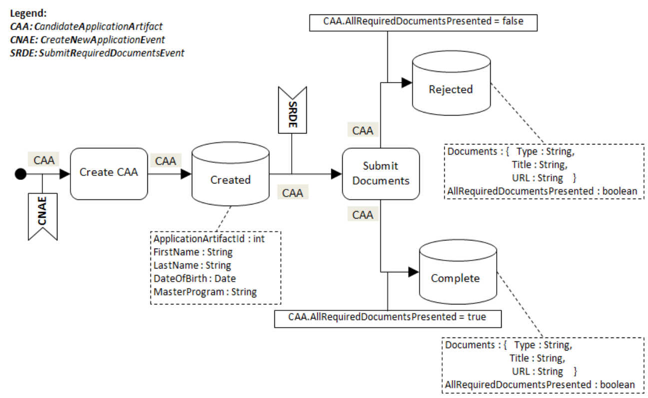


Figure 2. Conceptual Model of Candidate Application Business Artifact

3.2 Implementation Phase

In the *Implementation Phase*, conceptual models constructed in the *Modeling Phase* are translated into actual *Business Artifact Systems* in terms of: *Information Models*, *Lifecycle Models*, *Services Specifications*, and *Business Rules*. The *Business Artifact System* we employ is based on the systems proposed in [2, 5, 6].

The translation is performed according to a semantic transformation that is expressed in terms of first-order logic rules. To this end, we define the *Artifact Query Language* (*AQL*) which is a high-level query language intended for writing queries at the business logic level. And as a result, *AQL* is an abstraction layer over *SQL* that relieves its users from writing SQL queries that involve technical elements like multiple tables, primary/foreign keys constraints, and joins. *AQL* is composed of an *Artifact Definition Language* (*ADL*) and an *Artifact Manipulation Language* (*AML*). The generated system specifications are generated as *ADL Queries*. Additionally, we define an *AQL Query Processor* that is used to execute *ADL* and *AML* queries.

First, *ADL* includes four statements to create business artifacts (information model), services, business rules and lifecycles. These four statements are translated and executed by the *AQL Query Processor*. 1) *"create artifact statement"* supports the definition of simple, complex, and reference type attributes, in addition to the states of lifecycles. *"create artifact statement"* is translated into SQL Create Table and Insert queries. 2) *"create service statement"* specifies the input, output, precondition attributes, and effect attributes of services. *"create service statement"* is translated into semantic web service specifications. 3) *"create rule statement"* specifies the event, condition, and action of business rules. *"create rule statement"* is translated into Java Rule Engine Drools rule. 4) *"create lifecycle statement"* specifies possible state transitions and is translated into *Finite-State Machines* (*FSM*). Figure 3 illustrates a *"create artifact query"* for a *Candidate Application Artifact* where *ApplicationArtifactId*, *FirstName*, and *Age* are simple type attributes, *Documents* is a complex type attribute, *Interviews* is a reference type attribute, and *Initialized*, *Created*, *Rejected*, *Complete*, and *Archived* are states of its lifecycle.

```
Create Artifact CandidateApplicationArtifact With
Attributes (
          ApplicationArtifactId : Integer,
          FirstName : String,
          Age : Integer,
          Documents : ( Type : String,
                        Title : String,
                        URL : String       ),
          Interviews : CandidateInterviewArtifact
)
States (
          Initialized as initial state,
          Created,
          Rejected as final state,
          Complete,
          Archived as final state
)
```

Figure 3. Candidate Application Artifact Create Query

Second, *AML* includes statements to manipulate and interrogate business artifact instances. 1) The *Instantiate* statement is used to instantiate new business artifact instances. 2) The *update* statement is used to update simple type attributes. 3) The *Insert* and *Remove* statements are used to insert and remove complex and reference type attribute values into business artifacts. 4) The *Delete* statement is used to delete business artifact instances. Finally, 5) the *Retrieve* statement is used to retrieve business artifact instances that meet certain conditions. Figure 4 illustrates examples of some *AML* queries written on the *Candidate Application Artifact* with the id 100543. All of the *AML* queries are translated by the *AQL Query Processor* into *SQL* queries according to defined translation semantics.

```
1) Instantiate New CandidateApplicationArtifact With
   Values(100543, "John", 23)
   State Set To Created

2) Insert Documents Into CandidateApplicationArtifact
   {
     ("Diploma", "Bachelor in Computer Science", "http://..."),
     ("Letter", "Recommendation Letter", "http://...")
   }
   Where ApplicationArtifactId = 100543

3) Update CandidateApplicationArtifact
   Set State to Complete
   Where ApplicationArtifactId = 100543

4) Retrieve CandidateApplicationArtifact
   Where State Is Complete
   And age = 23
   And Documents Include {
         ( * , "Bachelor in Computer Science", * )
   }
   And Interviews In Ready State

5) Delete CandidateApplicationArtifact
   where ApplicationArtifactId = 100543
```

Figure 4. AML Query Examples

3.3 Integration Phase

In the *integration phase*, several local conceptual models from the *Modeling Phase* are integrated into one global model that acts as a centralized access point. The *integration phase* is based on our preliminary results on artifact lifecycle integration [1] and composed of three sub-phases: *Matching Sub-Phase*, *Generation Sub-Phase*, and *Mapping Sub-Phase*. First, the *Matching Phase* deals with identifying correspondences between different elements of the local *Business Artifact Models*. Second, the *Generation Phase* deals with generating the unified view or global *Business Artifact Model* based on the identified correspondences of the *Matching Phase*. Finally, the *Mapping Phase* defines mappings between local *Business Artifact Models* and global *Business Artifact Model* so that *AML* query transformation between them can be achieved. Furthermore, each of the three sub-phases is composed of several operations.

3.3.1 The Matching Sub-Phase

The Matching Sub-Phase consists of three incremental *match operations*. The first *match operation* identifies correspondences between business artifacts, tasks and repositories. This *match operation* is manually performed by the user using a specialized graphical interface. *Zero-to-one*, *one-to-zero*, and *one-to-one* correspondences between *business artifacts*, *tasks* and *repositories* are allowed in addition to *one-to-many* and *many-to-one* correspondences defining compositions between *tasks*. Figure 5 illustrates the different correspondences relationships of this *match operation* and their graphical representation.

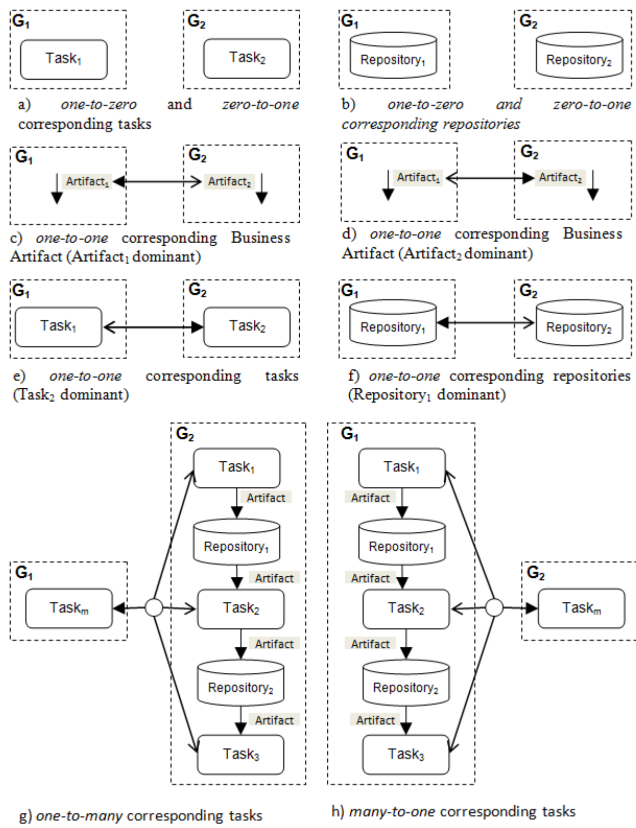
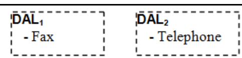
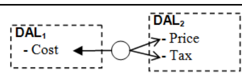
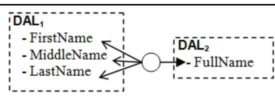


Figure 5. Artifact, Task and Repository Correspondence Relationships

Based on the identified correspondences, the second *match operation* is automatically performed and identifies correspondences of *flow connectors*. Finally, the third *match operation* is manually performed by the user using a graphical interface and identifies correspondences between *data attributes* of the models. In this *match operation*, the user specifies *matching expressions* for corresponding *data attributes* using a set of defined functions that allows correspondences of the type *one-to-one* and *one-to-many* between others. Table 2 illustrates some examples of the *data attribute match operation*.

The result of the *matching phase* is a set of correspondences between the different elements of the local *Business Artifact Models* and is used in the remaining sub-phases of *the Integration Phase*.

Table 2. Data Attribute Correspondences Examples

Match Relationship	Graphical Notation	Matching Expression
One-to-zero, zero-to-one		-
One-to-one		Amount = Price * 1000
One-to-many		Cost = Price * (1 + Tax/100)
Many-to-one		FullName = concat(FirstName, " ", MiddleName, " ", LastName)

3.3.2 The Generation Sub-Phase

The *Generation Sub-Phase* deals with generating the global Business Artifact Model based on the collected correspondences of the *Matching Sub-Phase*. The *Generation Sub-Phase* consists of five incremental steps. First, *business artifacts* are generated, and then *tasks*, *repositories*, *data attribute lists*, and finally *flow connectors*. Each step is based on the result of the previous steps.

3.3.3 The Mapping Sub-Phase

The *Mapping Sub-Phase* deals with generating mappings that are used to translate *AML* queries between local *Business Artifact Models* and global *Business Artifact Model*. The generated mappings are based on the structure and relationship between the different *BAMN* constructs used to model business artifacts.

4. CONLUSION AND FUTURE WORK

The effective integration of artifact-centric processes poses a serious challenge since it requires dealing with the complexity of its components and their relationships. In this thesis, we propose a design for an artifact-centric process integration system that makes use of a graphical artifact modeling notation and an artifact query language that support the integration challenge.

Future work will seek to define and formalize the artifact query language and develop query reformulation algorithms suitable for this language and implementing it as the *Mapping Sub-Phase* of the integration system. Additionally, algorithms should be developed to support the *Generation Sub-Phase* of the integration system.

5. REFERENCES

[1] M. Abi Assaf, Y. Badr, K. Barbar, and Y. Amghar. On the Integration of Artifact Lifecycles. In *Proceedings of the 7th International Conference on Management of Computational and Collective Intelligence in Digital Ecosystems*. ACM, 59-63, 2015.

[2] K. Bhattacharya, C. Gerede, R. Hull, R. Liu, and J. Su. Towards Formal Analysis of Artifact-Centric Business Process Models. *Business Process Management*. Springer Berlin Heidelberg, 288-304, 2007.

[3] D. Calvanese, G. De Giacomo, R. Hull, and J. Su. Artifact-Centric Workflow Dominance. In *Service-Oriented Computing*, Springer Berlin Heidelberg, 130-143, 2009.

[4] D. Cohn, and R. Hull. Business Artifacts: A Data-Centric Approach to Modeling Business Operations and Processes. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering* 32(3):3-9, 2009.

[5] C. E. Gerede, and J. Su. *Specification and Verification of Artifact Behaviors in Business Process Models*. Springer Berlin Heidelberg, 181–192, 2007.

[6] C. E. Gerede, K. Bhattacharya, and J. Su. Static Analysis of Business Artifact-Centric Operational Models. In *IEEE International Conference on Service-Oriented Computing and Applications* IEEE, 133-140, 2007.

[7] D. Hai. *Schema Matching and Mapping-Based Data Integration: Architecture, Approaches and Evaluation*. VDM Verlag, 2007.

[8] R. Hull. Artifact-Centric Business Process Models: Brief Survey of Research Results and Challenges. In *On the Move to Meaningful Internet Systems: OTM 2008*. Springer Berlin Heidelberg, 1152-1163, 2008.

[9] M. La Rosa, M. Dumas, R. Uba, and R. Dijkman. Business Process Model Merging: An Approach to Business Process Consolidation. *ACM Transactions on Software Engineering and Methodology*. 22(2):11, 2013.

[10] M. Lenzerini. Data Integration: A Theoretical Perspective. In *the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, ACM, 233-246, 2002.

[11] R. Liu, K. Bhattacharya, and F. Y. Wu. Modeling Business Contexture and Behavior Using Business Artifacts. In *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, 324-339, 2007.

[12] N. Lohmann, and K. Wolf. Artifact-Centric Choreographies. In *International Conference on Service-Oriented Computing*. Springer Berlin Heidelberg, 32-46, 2010.

[13] A. Nigam, and N. S. Caswell. Business Artifacts: An approach to operational specification. *IBM Systems Journal* 42(3):428-445, 2003.

[14] C. Parent, and S. Spaccapietra. Database Integration: The Key to Data Interoperability. In *Advances in Object-Oriented Data Modeling*. MIT Press, 221-253, 2000.

[15] E. Rahm, and P. A. Bernstein. A Survey of Approaches to Automatic Schema Matching. *The VLDB Journal*, 10(4): 334-350, 2001.

[16] S. Sun, A. Kumar, and J. Yen. Merging Workflows: A New Perspective on Connecting Business Processes. *Decision Support Systems*, 42(2): 844-858, 2006.

[17] W.M.P. Van Der Aalst, A.H.M. Ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. *Distributed and Parallel Databases* 14(3):5-51, 2003.