A Survey on Approaches to Modeling Artifact-centric Business Processes

Jyothi Kunchala [1], Jian Yu[1], and Sira Yongchareon[2]

[1]School of Computer and Mathematical Sciences
Auckland University of Technology, Auckland, New Zealand
{kjyothi, jian.yu}@aut.ac.nz

[2]Department of Computing
Unitec Institute of Technology, Auckland, New Zealand
sira@maxsira.com

Abstract: Business Process Modeling using artifact-centric approach has gained increasing interest over the past few years. The ability to put data and process aspects on an equal footing has made it a powerful tool for efficient business process modeling. The artifact-centric approach is based on key business-relevant entities called business artifacts, which are central for guiding business operations as they navigate through the business operations. The artifact-centric modeling approach can be laid in a four dimensional framework called BALSA for defining business processes, where the four dimensions include business artifacts, lifecycles, services and associations. Based on this data-centric paradigm, several artifact-centric meta-models have been emerged in the recent years. Although all the proposed models claim to support the artifact-centric approach, their support in specifying the BALSA elements of artifacts was not clearly described in the existing literature. This paper reviews all existing approaches to artifact-centric modeling and also discuss to what extent they align with the BALSA framework.

Keywords. Artifact-Centric process modeling, Business Artifacts, BALSA

1 Introduction

The business managers and analysts in organizations increasingly rely on business process modeling to document, understand and improve their business processes. Business Process Modeling [1] refers to the act of representing business operations with an objective to improve organizations current business processes. A business process model describes how the business operates to accomplish its objectives. Traditional activity-centric business process modeling is based on a task and control-flow constructs, which only define how business processes operate, without revealing details about the data resulted from the business process execution. An "impedance mismatch problem" [2] arises with the separation of application, process, and control data by activity-centric process aware information systems while providing support to imperative procedural models, which eventually affect the flexibility of activity-centric business process modeling approaches.

As opposed to the traditional activity-centric approaches to business process modeling, whose emphasis is completely on tasks and their control flows, a new data-centric approach for

modeling business processes has been emerged namely, *artifact-centric* approach [3], which takes into account data and process aspects in a more comprehensive manner. The modeling approach is centrally based on business artifacts [4] i.e., core business-relevant entities that manage operations of the business, whose content changes in response to the business actions. The artifact-centric modeling becomes popular with its unique advantages such as: (1) it enables business managers to better understand and specify their business operations by providing a more intuitive framework [3]; (2) it provides more flexible and robust structure for business process specification [3]; and it has the potential to improve flexibility, compliance and reduce the complexity of traditional activity-centric business process approaches [5]; (4) it has the ability to help cut down the costs of business transformations [6].

The artifact-centric modeling approach can be laid in a four dimensional framework called BALSA- *Business Artifacts*, *Lifecycle*s, *Services*, and *Associations* [1, 3]. "*By varying the model and constructs used in each of the four dimensions one can obtain different artifact-centric business process models with differing characteristics*" [3]. Currently there are many concrete artifact-centric modeling approaches such as GSM [7], ArtiNet [8], AXML [9], BPMN Extensions [10], and ACP-i [5]. An existing question is how to compare them and which approach to use when considering a real modeling problem.

In this paper, we aim to use BALSA as a reference framework or yardstick and see how each approach can be fit into this framework. To give the reader a concrete feel of each approach, we also use a common motiving scenario and demonstrate how to implement this scenario using each approach. This paper also help researchers and practitioners who have interests in the area of BPM to gain better understanding and knowledge of artifact-centric process modeling.

The remainder of this paper is organized as follows: Section 2 provides an introduction to the BALSA Framework with an example. Section 3 discusses artifact-centric modeling approaches and related work. Section 4 discusses and briefly evaluates all the modeling approaches studied against the framework. Finally, the conclusion and future work are given in Section 5.

2 BALSA Framework

2.1 BALSA ELEMENTS

With the focus on data aspects as its first-class citizens, the artifact-centric approach provides a four explicit, inter-related but separable "dimensions" in the specification of business processes [1, 3], where this four-dimensional framework is named as BALSA- Business Artifacts, Lifecycles, Services, and Associations. Each of these dimensions can be described as follows:

***Business Artifacts*:** The term "artifact" has its own roots in the business domain. In general, we can describe an artifact as, a means to record business information needed to perform business operations. And in the business terminology, an artifact can be better described as a key business-relevant entity responsible for driving overall business operations to achieve business objectives [4]. An artifact serves as a basic building block for business process modeling by clustering both the information aspects and process aspects in a more comprehensive way. An artifact contains two parts: an identity and content, where an identity differentiates it from other artifacts and its content can be represented with a set of attributes which hold data about those artifacts. Here the attributes and their values may be created, updated or deleted by the business tasks in the workflow. An important aspect of artifact is its type, which can be characterized by its *data/ information model* and *lifecycle model*, where the data model describes the business data that an artifact captures, and its lifecycle model specifies the possible stages that an artifact navigates through by responding to events and services that act on it. The data model can be specified in many forms, e.g., a name-value notation [4], an XML or ER model [1].

***Lifecycle*:** The lifecycle of artifact can be described as key business-relevant stages, through which an artifact navigates from its initiation to the completion. Different artifacts may differ with their "life expectancies". In general, the lifecycle may be specified using flow charts, finite state machines, state charts or using declarative mechanisms [1].

***Services*:** A service can be described as a business task or an action performed on the artifact to progress towards business objectives. Service invocation on artifact may result in a state change of the artifact, and/or update artifact's content. Services can be specified with pre-conditions and post-conditions [11, 12, 13].

Associations: Associations specify the association among services and artifacts and their constraints. Here the constraints correspond to the conditions under which services can be executed. The constraints may be specified procedurally [4] or declaratively [11, 12, 13], for e.g., using flowcharts or ECA (Event-Condition-Action) rules, respectively.

2.2 Running Example

In this section, we describe a running example which is used throughout this paper for illustrating the BALSA aspects of each concrete modeling approach.

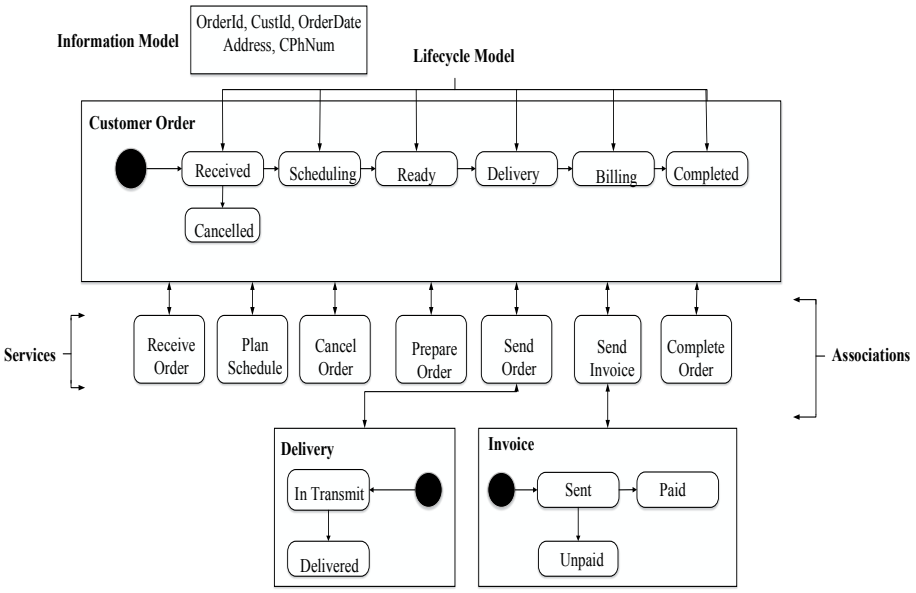


Fig. 1. Customer Order processing scenario

The figure shown above, which is adopted from [11], presents a Customer Order processing scenario, where the process starts by receiving an order from the customer and ends with successful delivery. Here the Customer Order forms one of the key entities of the business and interacts with other entities of the business such as Delivery and Invoice to complete its processing.

The information model of Customer Order artifact includes attributes such as *OrderID*, *OrderDate*, *CustID*, *CustAddr*, *CPhNum*. In the same way, the information model of Delivery artifact may include attributes such as *DeliveryID*, *DDate*, and *DStatus* and the information model of Invoice artifact may include *IVDate*, Total and *IVStatus* attributes. The lifecycle model of Customer Order artifact, specify the states as *Received, Scheduling, Ready, Delivery, Billing,* and *Completed*. In the same way other artifacts such as Delivery, Invoice also have their lifecycle states, where In *Transit* and *Shipped* states form lifecycle model of Delivery artifact and *Sent, Unpaid, Cleared* states form the lifecycle model of Invoice artifact. Services in the above example include *Receive Order*, *Plan Schedule*, *Cancel Order*, *Prepare Order*, *Schedule Shipping*, *Send Invoice* and *Complete Order* are the services that act on Customer Order artifact to change its state. The Receive Order service instantiates a Customer Order instance and puts it in the *Received* state. Similarly, Prepare Order service puts the artifact in *Ready* state. Similarly the Schedule Shipping and Send Invoice services act on multiple artifacts such as Customer Order and Delivery and Customer Order and Invoice. Associations between artifacts and services are represented through arrows that specify which services are acting on which artifacts and when.

Though the example described above is simple, it does not lack of generality and can be demonstrated by different modeling approaches. The following sections discuss each of the approaches studied in the paper in detail.

3 Artifact-centric Modeling Approaches

With the emergence of artifact-centric approach, several artifact-centric modeling approaches have been proposed, of which has its own characteristics and provides different ways to specify business operations. Some of the promising modeling approaches are GSM [7], ArtiNets [8], AXML [9], BPMN Extensions [10], and ACP-i [5]. The following section provides details and discussions to those modeling approaches with our running example to demonstrate how each approach can be applied on.

3.1 Guard-Stage-Milestone

In recent years, a declarative style of meta-model for specifying artifact lifecycles, Guard-Stage-Milestone (abbreviated as GSM) [7] has been introduced by IBM. The key constructs of the GSM meta-model used in specifying artifact lifecycles include: (1) ***information model,*** that holds all the data about an artifact instance; (2) ***milestones***, specify business objectives that might be achieved by an artifact instance; (3) ***Stages*** represents a set of activities performed by an artifact instance to achieve business objectives; and (4) ***Guards,*** simply act as sentries and represent a condition or triggering event to control stages and milestones.

GSM Modeling of BALSA Elements

GSM supports specification of BALSA elements. The following section describes how GSM meta-model represents each element of the BALSA framework.

Business Artifacts (Information Model): GSM specifies information model of artifact with attributes, where the artifact type can be a scalar, or a record type or a collection type. These attributes are categorized into 3 different types such as *Data attributes, Event attributes and Status attributes.* The Data attributes contain information about the artifacts. The Event attributes represent information about the triggering events. And, the Status attributes are intended to hold status information about the stages and milestones i.e., about the values associated to stages and milestones that change over time and are of type Boolean. The possible

values of status attribute may be open/close (active/inactive) for stages and true/false for the milestones.

***Lifecycle**:* The lifecycle of an artifact can be modeled using the constructs stages, milestones, guards. Here stages correspond to the states of the BALSA lifecycle. The stage structures the activities of an artifact instance and becomes true or said to be open when its associated guard becomes true. And the stage will be closed when its milestone is achieved or becomes true. The stage contains a stage body, guards, one or more milestones, and may contain multiple sub stages where stages at the same level can be executed in parallel.

***Services**:* In GSM, the services are specified in the form of events, where event occurrence may result in a state change or modify the value of milestone. There are two types of event types such as external event types and status-change event types. GSM supports 4 kinds of status-change event types where first two are denoted in GSM-L syntax as S.opened() and S.closed() and the other two are denoted as m.achieved() and m.invalidated().

***Associations**:* GSM uses Event-Condition-Action (ECA) rules in the specification of associations. The ECA rules take the form "take an action, when the event occurs under the specified condition". And in GSM, these ECA rules are formed from the sentries. The sentry here is expressed in the form '**on**<event> **if**<condition>**then**<action>.

3.1.1 GSM Representation of the Running Example

The following figure illustrates the key constructs of the GSM meta-model by using the sketch of Customer Order artifact type, described in the running example section.

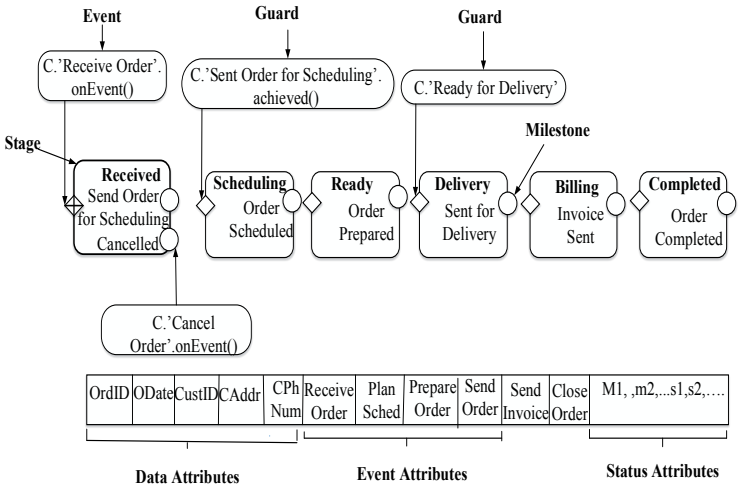


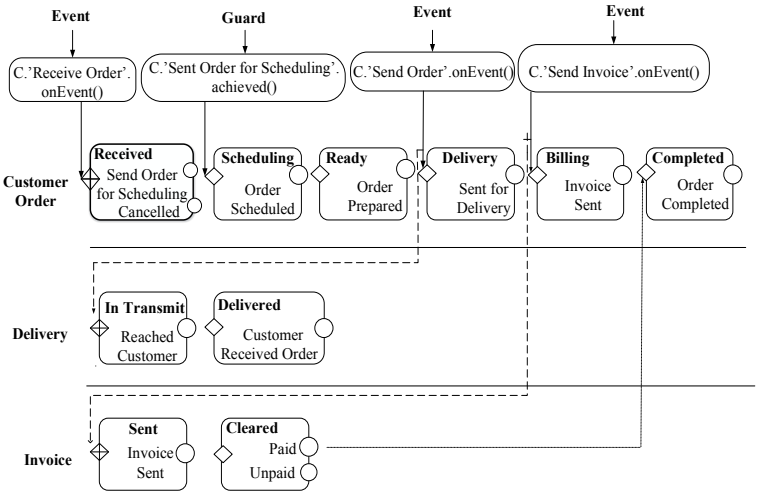
Fig. 2. GSM Representation for Customer artifact

Different kind of nodes in the figure are used to designate GSM constructs like, the rounded-corner rectangles are used to represent stages, the guards are designated using diamonds and the small circles associated to each stage represent a milestone. For Customer Order artifact, the data attributes hold the values of attributes that include *customerId*, *orderId*, *orderDate*, *customerAddress*, *phNumber*. For the Delivery artifact the data attributes include deliveryDate, Status and invoiceId, Total, Status are the data attributes of Invoice artifact. In the above figure,

the upper portion represents a lifecycle model of Customer Order artifact that includes a set of stages with milestones, which the Customer Order artifact might achieve during its lifetime. The Customer Order artifact moves through its lifecycle with the result of event occurrences. The c.'ReceiveOrder'.onEvent() event initiates the Customer Order artifact instance and puts it in *Received* stage. And in the same way the 'Cancel Order' event triggers *Cancelled* milestone. The Customer Order artifact enters Scheduling stage and achieves Scheduled milestone with the occurrence of 'Plan Schedule' event. In the same way the result of event occurrences lead to the completion of Customer Order artifact. The status attribute of milestone, for example 'm' is initially initialized to FALSE and can become TRUE if the milestone is achieved. The status of stage such as 's' becomes open, if its associated guard becomes TRUE and will be closed if its milestone becomes TRUE. The Guard is simply a sentry. When the guard condition such as c.'Send Order for Scheduling' is achieved, the milestone becomes true, and the artifact instance enters next stage by triggering its guard value to true. Then the corresponding event can be invoked on the artifact instance. Here the variable 'c' is used to denote the artifact instance currently under consideration.

3.1.2 Interaction between artifacts

The GSM supports interaction among artifacts through conditions and events [14, 15]. The figure below illustrates the interaction between the artifacts. The dashed lines denote B-steps that correspond to the incorporation of event into the GSM system, for example the Send Order event allows interaction between Customer Order and Delivery artifacts, where its result changes the state of Customer Order artifact from *Ready* to *Delivery* and also initiates the Delivery artifact instance. Similarly the Send Invoice event allows interaction between Customer Order and Invoice artifact.



3.2 ArtiNets:

ArtiNets workflow model [8], a variant of artifact-centric workflow models introduced to support the specification of artifact lifecycles and their constraints. Inspired by DecSerFlow [16], a Declarative Service Flow Language, ArtiNets also allows declarative style in specifying constraints on artifact lifecycles. The key components of ArtiNet model are: *artifacts, services,*

places, and transitions. ArtiNet framework is closely related to Petri nets [17], but only differs with two aspects: where artifacts form the key constructs of ArtiNet model instead of tokens, and the difference lies in the transition firing rule.

3.2.1 ArtiNet Modeling of BALSA Elements

Business Artifacts (Information Model): ArtiNet model primarily focuses on modeling the lifecycle aspects of artifacts, and their constraints. ArtiNet model may specify artefact's information model, but these details are not much addressed in its current literature.

Lifecycle*:* ArtiNet model uses three key constructs in the representation of BALSA Lifecycle such as *artifacts, places, services and transitions.* A place is a repository that stores artifacts, and transition correspond to the actions/ events invoked on artifacts, that may change the location of artifact from one place to another. The transition firing may consume/ generate only one artefact though it has multiple input/output places.

Services*:* A service in the ArtiNet model corresponds to a task performed on the artifacts, where the execution sequence of these tasks is defined by lifecycle constraints.

Associations*:* In ArtiNet model, the invocation of services on artifacts is limited by constraints (conditions). The constraints here may be the regular constraints, which are expressed using regular expressions [8] or counting constraints, which are expressed using semi-linear sets of Parikh map [18, 8]. Regular constraints specify a condition, under which a service can be invoked on the artifact, and the counting constraints specify how many number of times, the services should be executed.

3.2.2 ArtiNet representation of running example

The figure below presents an ArtiNet workflow representation of Customer Order artefact, where the rounded rectangles represent the services (tasks) that act on Customer Order artifact and the circles represent the places (states) that the artifact exists. The arrows correspond to the transition of artifact from one place to the other place. The service invocation on artifact leads to a transition of artifact from one state to other state. In the above figure, 'Receive Order' service creates a CO instance and puts it in the '*Received*' state. Then with the invocation of 'Prepare Schedule' service the artifact enters into the '*Scheduling*' state. In similar manner the artifact exist various other stages with response to service invocations, and can be archived when it finishes its completion.

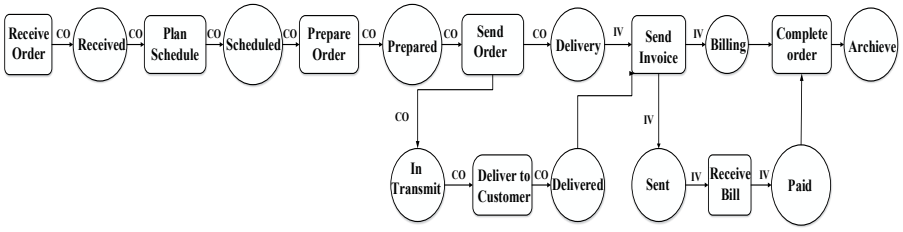


Fig. 4. ArtiNet representation for Custer Order Scenario

The service invocation on artifact leads to a transition of artifact from one state to other state. In the above figure, 'Receive Order' service creates a CO instance and puts it in the '*Received*' state. Then with the invocation of 'Prepare Schedule' service the artifact enters into the '*Scheduling*' state. In similar manner the artifact exist various other stages with response to service invocations, and can be archived when it finishes its completion.

The above model allows integration of all the three artifacts. Associations among the three artifacts is clearly illustrated in the above figure. And CO here is the Customer Order artifact instance which flows through the lifecycle, where the other artifacts may acquire for processing, when required.

3.3 The AXML Artifact model

The AXML artifact model [9] is a data-centric workflow approach that has been introduced to encapsulate data and workflow activities in a distributed environment. The AXML Artifact model is built based on Active XML [19], which is a declarative framework developed to tackle web services for distributed data management, and is designed to work in a peer-to-peer architecture. The AXML artifact model is designed to capture various aspects of artifacts such as their states, evolutions, interactions, and history [9, 20]. An AXML artifact specified in AXML artifact model captures data, tasks, actors, where the actors can be humans, processes or systems. And the state of an artifact is represented with an AXML document, represented in a tree structure with XML data and some function calls.

3.3.1 AXML Modeling to BALSA Elements

Business Artifact (Information Model)*:* AXML represents the artifact's data in the form of nodes in an artifact tree (AXML document). These nodes may be element nodes, content nodes. *Peer* act as a repository to store artifacts, supports interaction and provides computing resources for the artifacts they hold. The artifacts of one peer may exchange data (in the form of strings) with the artifacts of other peers, which can be reconstructed at the receiving peers.

Lifecycle*:* AXML supports modeling of artifact lifecycle, where the state of an artifact is represented with an AXM document that contains some nodes. These nodes may be element nodes, content nodes, function calls and sub artifacts.

Services*:* A service is a function call in AXML artifact model, which is specified with the function nodes in the AXML document. Here the services may be internal services or external services. Similar to GAXML, the function call in AXML has 4 components: *call guard*, which controls the activation of a function call, *argument query*, which computes call argument, *return guard*, that controls the result of the call, and *result query*, that computes the result of the call. The guards, arguments and return queries are specified using Boolean combinations of tree-patterns (BTPQ) queries over the documents.

Associations*:* The association among services (function calls) and artifacts is restricted by constraints, which are expressed in the form (*event, precond, postcond*). The event here may correspond to a function call activated on the peer, the call received at some other peer, result sending and the reception of that result. Other events such as the artifact creation, state change or archiving may correspond to function calls. The precond and postcond are the conditions specified as formulas (BTPQ) over the artifacts states.

3.3.2 AXML representation for Customer Order artifact

An AXML document, that represents Customer Order artifact tree is given below, that contains different kinds of nodes such as element nodes, content nodes, function nodes and some sub artifacts. The nodes such as cname, id, address, date are the element nodes, and the content nodes include Sam, 1001, 3214. The "creditCheck" element denotes a sub artifact, created by Customer Order artifact in order to check the details of the customer. And a function "warehouseCheck" is activated to check the availability of the order in the warehouse.

```
<customerorder artID="CO1">
<customer>
<cname> Sam </cname>
<id> 1001</id>
<address> Auckland</address>
<phno>2127378999</phno>
</customer>
<order>
<id>3214</id>
<date>20/03/2014</date>
<item1> LG Television</item1>
<item2>Samsung Mobile</item2>
</order>
<creditCheck artID="CO1-cc">
<number>xxxxxx1234</number>
<pin>xxxxxx</pin>
</creditCheck>
<fun funID="warehouseCheck"/>
</customerordery>
```

Fig 5. AXML representation for Customer Order Artifact

3.4 BPMN Extensions

The BPMN (Business Process Modeling Notation) standard has been extended to support artifact-centric business process modeling. Various BPMN Extensions [10] proposed to model different aspects of artifact-centric approach include as artifacts, object lifecycles, location information, access control, goal states, and policies [21]. Artifacts form the key constructs of the model, lifecycle specifies the possible states of artifacts, location information specifies how the location of artifacts is changed, and access control specifies that the artifacts are accessed remotely. These extensions are suitable for modeling single artifact. The remaining extensions such as goal states and policies are used in removing undesired behavior of artifact.

3.4.1 BPMN modeling to BALSA elements

Business Artifact (Information Model)*:* BPMN uses a data objects representation for artifacts, where a placeholder symbol can be used to hold the data object and its name appears in the upper left corner of the place holder symbol. BPMN may also specify the information model of the artifact, but it is out of scope of the current literature.

Lifecycle*:* The lifecycle of each artifact is modeled using the constructs like tasks, events and gateways, where the initial (start event) and final (end event) states of the artefact is denoted by standard BPMN symbols. The task here is simply a service, which changes the state of the artifact. And an event represents current state of the artifact. Goal states define desired final states of the data objects, and represented using parallel gateways which connect these states.

***Services**:* The service corresponds to a task in BPMN, which is triggered by an agent. Here the agent can be a role or organization or location. Before executing a task on the artefact, the agents need to acquire it. The agent can acquire an artefact by knowing its location (e.g., URL) or by using any other addressing mechanisms.

***Associations**:* BPMN supports associations among services and artifacts through policies, which restrict the execution sequence of tasks in different artifacts. Policies are also used in the specification of dependencies among tasks of one or two artifacts and are modeled using the constructs tasks and gateways.

3.4.2 BPMN representation of Customer Order Scenario

The figure below illustrates an artifact-centric BPMN model of the Customer Order processing scenario, where all the three artifacts are depicted using placeholder symbols with their names appear in the top left corner. The lifecycle of each artefact is modeled using the constructs like tasks, events and gateways, where the task is represented by rounded rectangle, which contain name of the task and the agent who triggers the tasks on artifacts, for e.g., Receive Order, Plan Schedule are the tasks with agents Customer and Employee. Events are represented in the model with double rounded circles, and denote current state of the artifacts like *Received*, *Scheduling* and *Completed* etc. The initial and final state of an artifact is represented by using BPMN symbols. Similar to the lifecycle representation, the location information is also included in the upper portion of the artefact with a stick figure, where each agent is assumed as a location, which specifies the current location of the artifact. Arrows between the agents represent message channels through which they exchange artifact instances.

The policy 1 in figure specifies the dependencies between the tasks of two artifacts Customer Order and Delivery, which can also be modeled using the constructs tasks and gateways. Here this policy describes that the "Prepare Order" task of Customer Order artefact should be executed before the "Send to Customer" task of Delivery artifact. We assume that an Invoice artefact has goal states as "*Receive Pay by Cheque/Card*" and "*Receive Pay by Cash*" which are connected through a parallel gateway, to present two payment options for the customer. The customer can pay through any of the two modes, which completes processing of Invoice artifact.

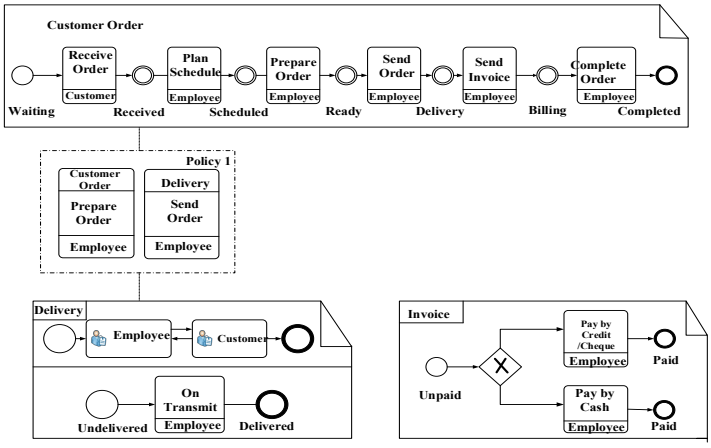


Fig. 6. Artifact-centric BPMN model of the Customer Order scenario

3.5 ACP-i Model

The ACP-i model [5], an artifact-centric business process model is an extended version of ACP model presented in [22, 23], has been proposed to support inter organizational business process modeling. The core components of this approach include: *roles*, *artifacts*, *tasks* and *business rules*. The roles are the organization roles participate in the collaboration, an artifact here is a business entity or object that exists in the collaboration, a task is an operation (read/update) on artifacts performed by the organizations in the collaboration, and a business rule specifies a set of constraints on tasks in a Condition-Action style. The ACP-i model distinguishes artifacts into two types such as local artifacts, and shared artifacts. The local artifacts are the artifacts owned by the organizations and shared artifacts, correspond to the commonly agreed artifacts used for coordination among parties in the collaboration.

3.5.1 ACP-i modeling to BALSA Elements

Business Artifacts (Information Model)*:* The information model of an artifact is represented by using a name-value pair notation, where each attribute is of type scalar or by using an array list of nested attributes.

Lifecycle*:* Lifecycle of an artifact is represented using a state machine with set of states, where state transitions of the artifact is based on business rules. Label Transition System (LTS) is used in ACP-i model to capture these lifecycles.

Services*:* A service in ACP-i model is specified as a task (action) that performs read/update operations on the artifacts, which is constrained by the conditions, defined in the business rules. The organizations involved in the collaboration perform these tasks according to the defined business rules.

Associations*:* Associations are specified through business rules, which specify conditions, under which the task should be invoked on artifacts. The conditions here are the constraints expressed in Condition-Action style as pre-conditions and post-conditions.
Here the business rules are classified into two types, where the first type of business rules are used to change the state of single artifacts, whereas the other type of business rules called synchronization rules can be used for expressing synchronization dependencies among artifacts and are used to change the states of multiple artifacts.

3.5.2 ACP-i representation of running example

The following figure illustrates the ACP-i representation for Customer Order processing scenario. To demonstrate ACP-i model let us assume the artifacts described so far are the shared artifacts such as Customer Order (CO), Delivery (D), Invoice (IV). We consider the other artifacts such as Stock Check (SC), Schedule Delivery (SD), Bill (B) in the figure as local artifacts whose details are kept private by the organizations, which own these artifacts and not revealed to the external parties in the collaboration.

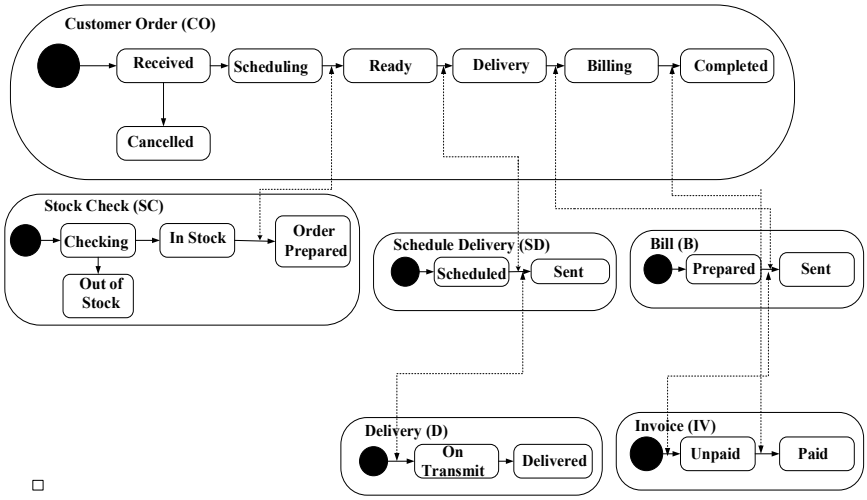


Fig. 7. ACP-i representation for Customer Order Processing scenario

Once the order is received from the customer by executing the task receiveOrder(co), the employee interacts with the SC artifact to find the availability of the stock. If the stock is available, the employee prepares the order, where the SC artifact enters the *Order prepared* state to complete its processing. When the SC artifact enters the O*rder prepared* state with the result of task prepareOrder(sc) task, which automatically triggers the state transition of CO from *Scheduling* to *Ready* state. The employee cancels the order if the stock is unavailable. When the order is ready, the employee interacts with Schedule Delivery (SD) artifact to schedule the delivery, this process ends when the DS artifact enters the state *Sent*, which consequently triggers the delivery state of CO artifact and the initiates the D artifact. For this scenario, the business rule can be expressed as (pre-condition : *instate(co, ready)^instate(sd,prepared)*; Task: *sendOrder(sd,co)* ; post-condition: *instate(sd,sent) ^instate(co,delivery)^instate(d,init))*. Similarly the employee interacts with Bill artifact to prepare bill for the order, this process ends when the B artifact enters the state *Sent*, which consequently triggers the *Billing* state of CO artifact. The entire CO process ends when it enters the *completed* state. The dashed lines represent the interactions among artifacts, which are specified through synchronization dependencies.

4 Discussion

The concept of business artifacts and the notion of modeling business processes in terms of artifact lifecycles were introduced in the literature by Nigam and Caswell [4]. This data-centric approach formed foundation for various artifact-centric meta-models in recent years.

GSM [7], a declarative approach for lifecycle specification, has been introduced by IBM team. The GSM operational semantics [14] are based on Event-Condition-Action (ECA) rules, and can be used in controlling activities, and recording status of the milestones. These operational semantics are related to the GSM B-steps (or Business steps) [14], which correspond to the business-relevant changes, the system undergoes with the occurrence of an incoming event. The GSM provides three equivalent formulations [14, 15] to support interaction between artifact instances. A prototype engine, called Barcelona [24], has been introduced to support design-time and run-time environments of GSM. Barcelona supports GSM schemas with multiple artifact types and also a huge number of artifact instances.

Barcelona provides a graphical design editor and allows mapping of GSM Business Object Models directly into an XML format.

Table 1. Comparison framework for all the artifact-centric modeling approaches studied in this paper

Approach	Information Model	Lifecycle	Services	Associations
GSM	Programming data types	Declarative	Declarative (Events)	Declarative (ECA-rules)
ArtiNet	Procedural	Procedural	Procedural (Tasks)	Declarative (ECA-rules)
AXML	XML elements	Declarative	Declarative (Function calls)	Declarative (ECA-rules)
BPMN	Procedural	Procedural	Procedural (Tasks)	Procedural (Policies)
ACP-i	Name-Value pair notation	Declarative	Declarative (Actions)	Declarative (Condition Action rules)

Other approach called ArtiNets [8], also enables the declarative specification of constraints on artifact lifecycles in the spirit of DecSerFlow [16] language. The AXML artifact model [9] also supports declarative lifecycles based on Active XML [19], but takes the hierarchical structure in data representation and supports implementation of artifacts, which can be accessed among organizations in the collaboration.

The standard BPMN [10] approach has also been extended to support artifact-centric modeling, and provides additional constructs that help business people to model various other aspects of their business processes.

The other approach, ACP-i [5] also supports artifact-centric modeling, and mainly focuses on modeling inter-organization business processes. In [25], a framework for realizing artifact-centric process models in SOA has been proposed and implemented based on the ACP-i approach proposed in [23].

All the above discussed approaches support modeling of BALSA elements. The following table presents the details on, how each approach represents the four dimensions of the artifact-centric approach.

The key motivations in the development of GSM are to find a meta-model to aid business stakeholders in specifying and managing their business operations by using intuitive natural constructs that closely resembles the ideology of business stakeholders about their business operations [7]. The GSM contrasts with procedural approaches such as BPMN by following a declarative approach, and supports parallelism within artifact instances and modularity through hierarchical constructs [15]. GSM closely corresponds to Case Management [14] and served as a foundation for CMMN core model [26]. But the GSM does not support that level of declarative-ness, when compared to DecSerFlow [7].

When compared to the AXML artifact model which focuses on structural aspects, the GSM mainly focuses on managing data aspects. Artinets, allow integration of lifecycles in one model, where the coordination is acquired through transitions on multiple artifacts.

Similar to GSM and ArtiNets, the ACP-i model also focuses on behavior aspects, where Label Transition System (LTS) is used to capture these behavior aspects. An ACP system

prototype shows that the ACP-i model can be directly executed without model transformation [25], i.e., no need to convert an artifact-centric model to activity-centric model such as BPEL.

5 Conclusion and Future work

In this paper, some key artifact-centric modeling approaches have been reviewed and discusses by using a four-dimensional framework called BALSA as a reference framework. A running example has been used to help illustrating each approach. We also present an initial evaluation and comparison of the approaches discussed in this paper. In the future, we plan to do more thorough evaluation of the approaches through both real-life case study and in-lab experiments.

References

1. Bhattacharya, Kamal, Richard Hull, and Jianwen Su. "A data-centric design methodology for business processes." *Handbook of Research on Business Process Modeling* (2009): 503-531.
2. Russo, Alessandro, et al. "Implementing and Running Data-Centric Dynamic Systems." *Service-Oriented Computing and Applications (SOCA), 2013 IEEE 6th International Conference on*. IEEE, 2013.
3. Hull, Richard. "Artifact-centric business process models: Brief survey of research results and challenges." *On the Move to Meaningful Internet Systems: OTM 2008*. Springer Berlin Heidelberg, 2008. 1152-1163.
4. Nigam, Anil, and Nathan S. Caswell. "Business artifacts: An approach to operational specification." *IBM Systems Journal* 42.3 (2003): 428-445.
5. Yongchareon, S., Liu, C., Zhao, X.: An Artifact-centric View-based Approach to Modeling Inter-organizational Business Processes. In: WISE 2011, LNCS 6997, pp. 273-281.
6. Bhattacharya, Kamal, et al. "Artifact-centered operational modeling: Lessons from customer engagements." *IBM Systems Journal* 46.4 (2007): 703-721.
7. Hull, Richard, et al. "Introducing the guard-stage-milestone approach for specifying business entity lifecycles." *Web Services and Formal Methods*. Springer Berlin Heidelberg, 2011. 1-24.
8. Kucukoguz, Esra, and Jianwen Su. "On lifecycle constraints of artifact-centric workflows." *Web Services and Formal Methods*. Springer Berlin Heidelberg, 2011. 71-85.
9. Abiteboul, Serge, et al. "The AXML artifact model." *Temporal Representation and Reasoning, 2009. TIME 2009. 16th International Symposium on*. IEEE, 2009.
10. Lohmann, Niels, and Martin Nyolt. "Artifact-centric modeling using BPMN."*Service-Oriented Computing-ICSOC 2011 Workshops*. Springer Berlin Heidelberg, 2012.
11. Bhattacharya, Kamal, et al. "Towards formal analysis of artifact-centric business process models." *Business Process Management*. Springer Berlin Heidelberg, 2007. 288-304.
12. Deutsch, Alin, et al. "Automatic verification of data-centric business processes." *Proceedings of the 12th International Conference on Database Theory*. ACM, 2009.
13. Fritz, Christian, Richard Hull, and Jianwen Su. "Automatic construction of simple artifact-based business processes." *Proceedings of the 12th International Conference on Database Theory*. ACM, 2009.
14. Hull, Richard, et al. "Business artifacts with guard-stage-milestone lifecycles: managing artifact interactions with conditions and events." *Proceedings of the 5th ACM international conference on Distributed event-based system*. ACM, 2011.

15. Damaggio, Elio, Richard Hull, and Roman Vaculín. "On the equivalence of incremental and fixpoint semantics for business artifacts with Guard–Stage–Milestone lifecycles." *Information Systems* 38.4 (2013): 561-584.
16. Van Der Aalst, Wil MP, and Maja Pesic. *DecSerFlow: Towards a truly declarative service flow language*. Springer Berlin Heidelberg, 2006.
17. Murata, T. (1989). Petri nets: Properties, analysis and applications.*Proceedings of the IEEE*, *77*(4), 541-580.
18. Parikh, Rohit J. "On context-free languages." *Journal of the ACM (JACM)* 13.4 (1966): 570-581.
19. Abiteboul, Serge, Omar Benjelloun, and Tova Milo. "The Active XML project: an overview." *The VLDB Journal* 17.5 (2008): 1019-1040.
20. Abiteboul, Serge, Luc Segoufin, and Victor Vianu. "Modeling and verifying active xml artifacts." *IEEE Data Eng. Bull.* (2009).
21. Lohmann, Niels, and Karsten Wolf. "Artifact-centric choreographies." *Service-Oriented Computing*. Springer Berlin Heidelberg, 2010. 32-46.
22. Yongchareon, Sira, and Chengfei Liu. "A process view framework for artifact-centric business processes." *On the Move to Meaningful Internet Systems: OTM 2010*. Springer Berlin Heidelberg, 2010. 26-43.
23. Yongchareon, Sira, Chengfei Liu, and Xiaohui Zhao. "An artifact-centric view-based approach to modeling inter-organizational business processes." *Web Information System Engineering–WISE 2011*. Springer Berlin Heidelberg, 2011. 273-281.
24. Fenno Terry Heath, I. I. I., et al. "Barcelona: A design & runtime environment for declarative artifact-centric BPM." *Service-Oriented Computing*: 705.
25. Ngamakeur, Kan, Sira Yongchareon, and Chengfei Liu. "A framework for realizing artifact-centric business processes in service-oriented architecture."*Database Systems for Advanced Applications*. Springer Berlin Heidelberg, 2012.
26. Marin, Mike, Richard Hull, and Roman Vaculín. "Data centric BPM and the emerging case management standard: A short survey." *Business Process Management Workshops*. Springer Berlin Heidelberg, 2013.