A Scenario-based Modeling Method for Crossover Services

Meng Xi* , **#Jianwei Yin*** , **Yongna Wei*** , **Maolin Zhang*** , **Shuiguang Deng*** , **Ying Li***
*Zhejiang University, Hangzhou, China
corresponding author
{ximeng, zjuyjw, weiyongna, dengsg, cnliying}@zju.edu.cn, maolin.zml@alibaba-inc.com

***Abstract*—With the continuous emergence of large companies, businesses in different domains have been amalgamated, which leads to a prevalence of crossover services. Different from service choreographies and collaborations, the crossover services involve multiple domains and are highly integrated in terms of processes, entities, abilities, and resources. In a crossover service, the elements of independent service are disrupted and reorganized, which leads to the increase of the complexity of traditional modeling methods. This new form of service brings challenges to business analysis and modeling, such as semantic ambiguity, business coupling, and test explosion. In this work, a crossover service-oriented modeling method, namely CoSM, is proposed to help with the construction, reuse, and analysis of crossover services. CoSM is designed based on scenarios, which can be used to slice the processes and route to each other through the triggers. Besides, a prototype platform is built and concepts and processes can be constructed through that. We verify CoSM in a real case and design experiments to compare with pertinent models. Experimental results show that our model can express the same process with fewer connectors.**

***Keywords*-service model; crossover service; concept modeling; business process model;**

I. Introduction

Crossover service is a kind of services, which can deeply converge services from different industries, different organizations and different value chains [1]. Different from choreography, which defines business interaction by describing the public message exchange between services, crossover services get through the service processes in different domains and make them cooperate under the environment of unified data, resources and quality. And if we use collaboration to describe the interaction between participants in crossover services, it will lead to complex business processes and unclear business logic.

With the prevalence of crossover services, the modern service industry (MSI) is facing the challenges of semantic ambiguity, business coupling, and test explosion. In order to keep competition, the MSI companies have to invest significant time, resources and manpower to improve their business processes. Effective modeling and management of business processes has become an important part of business competition.

Existing business process models are mainly of two categories: activity-centric and data-centric. Data-centric models have better control of data, while activity-centric modeling methods can describe the logic of business processes more intuitively. Representative activity-centric business process models are BPEL and BPMN. The artifact-centric approach is one of the most well-known data-centric models. However, because a crossover service usually amalgamates business processes in multiple fields, its process is usually complex and contain a large number of data entities. This leads to the challenge of semantic ambiguity, business coupling, and test explosion in traditional modeling methods.

In this work, we perform detailed research on the transaction business of Taobao (TB), a typical crossover service that involve activities from e-commerce, finance, and delivery and both online and offline scenarios. And then we propose a crossover-oriented service model (CoSM). The CoSM consists of two sub-models: CoSM-concept model (CoSM-CM) and CoSM-process model (CoSM-PM). CoSM-CM is designed to effectively manage entities, abilities and attributes in crossover complex scenarios. CoSM-PM is proposed to cope with increasingly complex crossover service processes. CoSM-PM slices processes into scenarios and decomposes one complex process into several simple ones. The route among scenarios are completed through the triggers. A CoSM based prototype system with graphical interface is constructed as well. Our contributions are as follows:

- A concept model (CoSM-CM) that can manage and maintain mass entities, data and abilities in large service systems is proposed. CoSM-CM can express the constraints among the constituents and avoid the misuse of abilities and data at the model level.
- A scenario-based process model (CoSM-PM) is proposed. CoSM-PM is able to slice processes by scenarios and reduce the complexity of a single process. It can improve the readability of the processes and make it easier to analyze and reuse them.
- A detailed case study on the Taobao transaction service are given. Based on that, the possible challenges and problems in the development of crossover services are summarized.

The structure of this paper is as follows. Section II is the motivation case, which illustrates the challenges in crossover service modeling. The detailed introduction and formalized definition of CoSM are illustrated in section III. Section IV details the TB transaction case to illustrate the

2474-2473/20/$31.00 ©2020 IEEE
DOI 10.1109/SCC49832.2020.00010

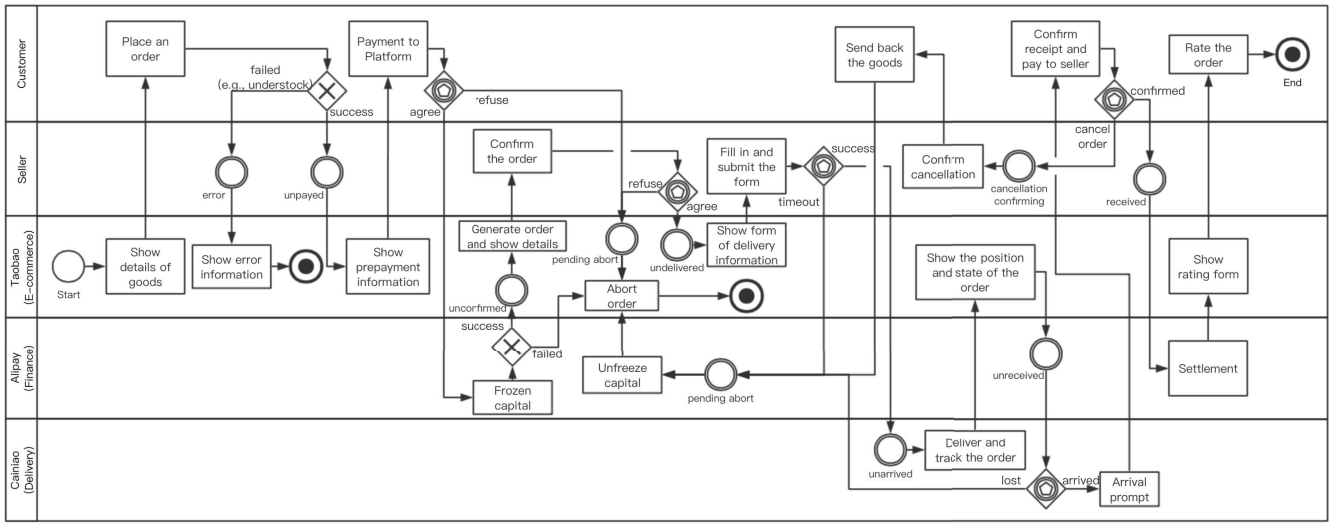


Figure 1: The Tansaction Process of Taobao

modeling approach based on CoSM. Section V discusses advantages of CoSM qualitatively and quantitatively through both theoretical and experimental methods. Related works and conclusions are given in the last two sections.

II. Motivation Case

In this section, we take Alibaba's Taobao transaction as example to illustrate the characteristics and challenges of crossover services (see Figure 1). The transaction process runs like this. The order becomes unpaid as the customer places the order successfully. Then the financial platform will freeze part of the consumer's money and give the order to the seller for confirmation. And the seller needs to deliver the goods to the consumer through the logistics company. After the order arrives and the consumer signs in, the financial platform performs the settlement. In any order status, the customer may request and trigger a return process.

As shown in Figure 1, activities from different domains are involved in the transaction, like Taobao (e-commerce), Alipay (finance), and Cainiao (delivery). Concepts and processes of different domains are brought together and integrated into one process. Those make the process as complex and confusing as a labyrinth, and brings great difficulties to the design, analysis and test of crossover services. In this example, two aspects of characteristics in crossover services are depicted:

Aspect 1: Multi domain entities are amalgamated together. Data entities and participants in crossover services are usually a combination of multiple domain ones. For example, the order in Taobao transaction is a combination of goods transaction order, logistics order, and financial payment order. The customer is also the user of three platforms at the same time.

Aspect 2: Business processes tend to be complex. The data, activities, and participants from different domains are integrated into one process. There are always more activities and events involved in the process of crossover services.

Those characteristics also bring challenges to MSI:

Challenge 1: Semantic Ambiguity. Entities, data, and abilities in different domains are integrated into one business, and problems in semantic understanding arise. E.g., address represents the specific address of the customer on e-commerce platform, while for logistics company it indicates the data packet containing the customer's name, phone number, zip code, and specific address.

Challenge 2: Business Coupling. Due to the amalgamation of different domain business, multi processes are coupled together. The analysis and reuse of existing processes becomes rather challenging.

Challenge 3: Test Explosion. Crossover service convergence has lengthened business processes and increased service complexity. It also leads to an explosion in the number of service test cases. E.g., in the e-commerce transaction process, the consumer can choose to cancel the order at any stage before confirming the receipt, i.e. unpaid, unshipped, undelivered, delivered and unconfirmed. This multiplies the number of paths that need to be tested.

In this work, we proposed a crossover-oriented service model(CoSM). It combines the ontology-related theory and introduce scenarios to process modeling, and is able to manage the entities and concepts in crossover services efficiently and reduce the process complexity.

III. CoSM

We will illustrate the notations in detail through a top-down way in this section. The top-level structure will be introduced first to provide a holistic view, and the notations

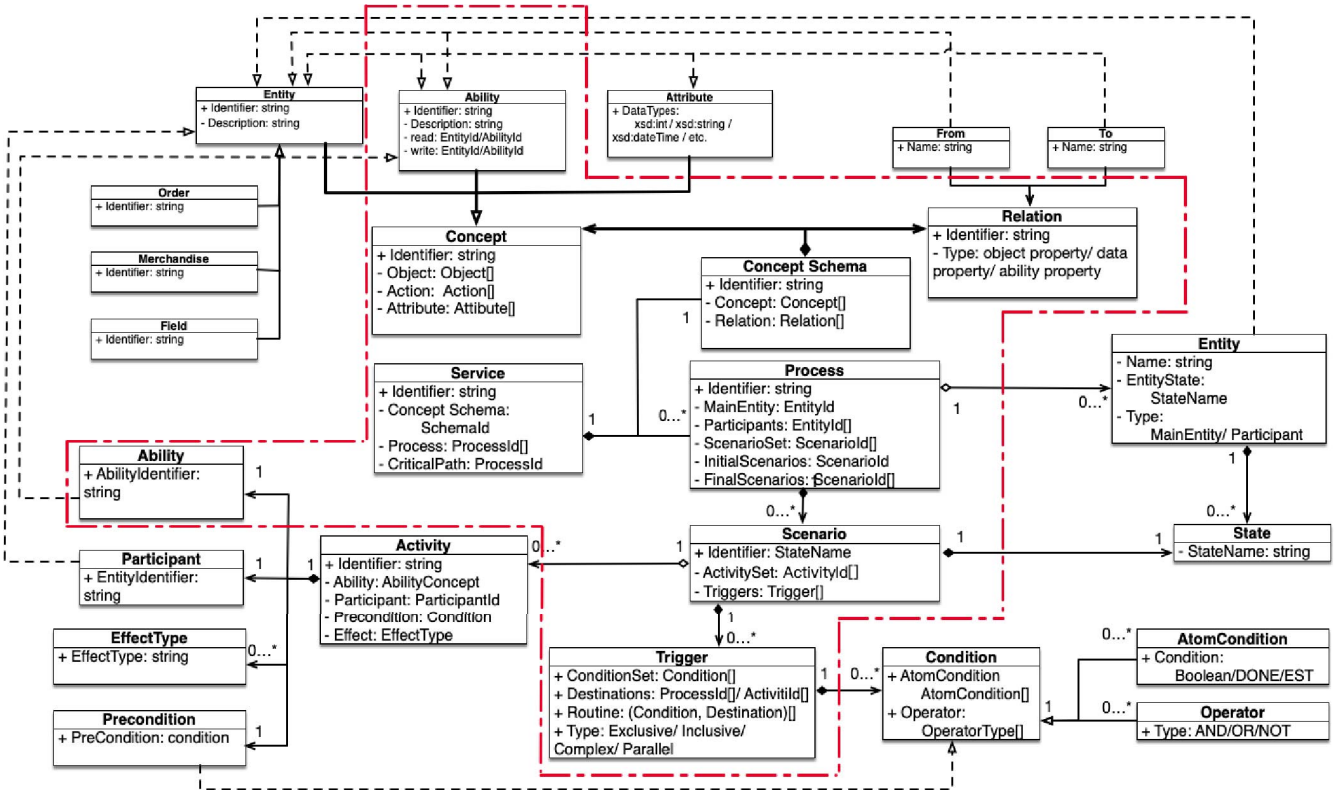


Figure 2: The UML diagram of basic notions in CoSM.

will be elaborated afterwards. In CoSM, an independent service should be a set of connected processes based on a determined limited set of concepts. A process is formed by splicing multiple scenarios. And the processes and scenarios are routed through triggers. The UML diagram of basic notions in CoSM is shown in Figure 2, where the innovative elements are highlighted with red dash-dot lines.

Definition 1. *A service is a quad $\sigma = (S, \Gamma, \mathbb{P}, p)$, where S is the identifier, Γ is a concept schema, $\mathbb{P}$ is a family of processes. $p \in \mathbb{P}$ is the indicator of critical path and from which the service should run.*

As shown in Definition 1, there are multiple processes in a service in CoSM. The critical path is the execution path where the business value can be reflected most. Because there are multiple processes in a CoSM service, the critical path is designed to determine where to initiate the service. There should be one and only one critical path in a service. And other processes require a trigger to execute.

A. CoSM Concept Model

In this work, we extended OWL to enable the expression of ability in knowledge graph. The focus of CoSM-CM is modeling methods for domain knowledge including domain abilities. Therefore, the construction methods of knowledge graphs such as knowledge extraction and knowledge fusion will not be discussed here.

Definition 2. *A concept schema is a 6-tuple $\Gamma = (C, \mathcal{E},$ **ATTR**$, \mathcal{I}, \Phi, \Psi)$, where C is the identifier, $\mathcal{E}$/**ATTR**/$\mathcal{I}$ are names of entity/attribute/ability (resp.). Φ is a set of predicates that can joint each entity/ability to the vertices except attributes in concept schema C. Ψ is a set of predicates that can joint each $\mathcal{E}$ to a the vertices of* **ATTR**.

Intuitively, in Definition 2, the extensions we made are mainly reflected in two points: a) Added vertex type ability $\mathcal{I}$. b) Expanded the original object property Φ in OWL.

An entity $\mathcal{E}$ is a collection of data for a business role in a business. CoSM is a data-centric process modeling approach where each service has its main entity. If entity e is the main entity of process p, and process p is the critical path of service σ, then e is the main entity of service σ.

Ability $\mathcal{I}$ is an abstraction of services, interfaces, and functions in the system. In CoSM-CM, input/output of abilities are reflected through the relationships with entity.

Regarding the object property Φ, data property Ψ, and other literal part (**ATTR**) of CoSM-CM, we import the corpus of XSD, RDF, RDFS, and OWL. The extensions of object property in our model are shown in Table I.

Definition 3. *An concept instance is a triple $c = (s, o, \mu)$,*

Table I: Extension of Object Properties in CoSM

CoSM Object Properties(Φ)	subject(s)	object(o)	paraphrase
sl:write	entity	ability	entity s has the right to invoke ability o
	entity	entity	entity s has the right to modify the information of entity o
	ability	ability	the output of ability s will trigger the execution of ability o as input, such as a notification will be sent to the seller by default after someone placing an order
	ability	entity	the output of ability s would write into entity o
sl:read	entity	ability	entity s has the right to obtain the output of ability o
	entity	entity	entity s has the right to obtain the information of entity o
	ability	ability	the input of ability s rely on out put of ability o
	ability	entity	the input of ability s rely on entity o
sl:own	entity	ability	entity s is the provider of ability o
	entity	entity	the existence of entity o rely on entity s, such as the existence of sellers rely on Taobao
	ability	ability	ability o is a component of ability s, this property is used to achieve ability composition
	ability	entity	the execution of ability s could generate entity o
owl:topObjectProperty	omit	omit	omit

Table II: Example of CoSM-CM Triples

subject	predicate	object
entity:TB	field_id	xsd:string
	sl:read	entity:order
	sl:write	entity:order
	sl:own	ability:create_order
	...	...
entity:customer	customer_id	xsd:integer
	capital	xsd:float
	address	xsd:string
	sl:write	ability:create_order
	...	...
entity:order	order_id	xsd:integer
	customer_name	xsd:string
	customer_phone	xsd:integer
	customer_address	xsd:string
	amount	xsd:integer
	price	xsd:float
	coupon	xsd:float
	...	...
ability:frozen_capital	sl:read	entity:order
	sl:read	ability:payment_to_platform
	sl:write	entity:customer
	sl:own	entity:Alipay
	sl:read	entity:seller
...	...	...

*where $s \in \mathcal{E} \cup \mathcal{I}$ is a subject, $o \in \mathcal{E} \cup$ **ATTR** $\cup \mathcal{I}$ is an object, and $\mu \in \Phi \cup \Psi$ is a property of s.*

Example 1. In TB transaction case, entities include order, customer, seller, TB, Alipay, Cainiao. The order is the main entity, which has order_id, price, coupon, and other data properties. Input and output of an ability can be determined through the sl:write, sl:read relationship with it as subject. The relationship of sl:own, sl:read, sl:write with the entity as subject can help further business reasoning and verification. Triples are shown in Table II, and corresponding graphic interface are shown as Figure 4a.

In Definition 3, we define the triple of the CoSM concept. The subject can be entity or ability. Object is from entity, ability or attribute. The property is selected from the object property and data property based on the subject and object. In Example 1, we show the concepts instances based on the TB transaction case.

To support automatic reasoning about concepts, relationships, and the processes involved, rules of subsumption, synonymous, and satisfaction are proposed and defined as below.

Definition 4. ***Subsumption Rule:*** *An entity $\mathcal{E}_a$ is subsumed by an entity $\mathcal{E}_b$, written $\mathcal{E}_a \sqsubseteq \mathcal{E}_b$, iff all subjects with $\mathcal{E}_a$ as the object are related to $\mathcal{E}_b$ with the same property, and all objects with $\mathcal{E}_a$ as the subject are related to $\mathcal{E}_b$ with the same property.*

The subsumption rule is mainly used to judge the relationship between entities in multiple domains and restrict the permissions of entities. It is reflexive, antisymmetric, and transitive. For example, since $consumer \sqsubseteq user$ and $(consumer, sl : write, create\_order)$, it can be inferred that $(user, sl : write, create\_order)$ exists as well. The $user$ specifies the upper bound of $consumer$, i.e., $consumer$ cannot access the entity, ability or attribtues that $user$ has not been given.

Definition 5. ***Synonymous Rule:*** *An entity $\mathcal{E}_a$ and an entity $\mathcal{E}_b$ are synonymous, written $\mathcal{E}_a \doteq \mathcal{E}_b$, iff $\mathcal{E}_a \sqsubseteq \mathcal{E}_b$ and $\mathcal{E}_b \sqsubseteq \mathcal{E}_a$.*

The synonymous rule can detect whether there is polysemy or structural redundancy in a crossover service. It is reflexive, symmetric, and transitive. For example, if exists $seller \doteq merchant$ in a crossover service, it can be inferred that $seller$ and $merchant$ have exactly the same access rights in the service and either needs to be optimized.

Definition 6. ***Satisfaction Rule:*** *Within a service σ, a process $p \in \mathbb{P}$ is satisfied with the concept schema Γ iff any instance of entity, ability, or attribute of p also belongs to Γ, and all the triples in which Γ instances can be involved are applicable to p.*

The satisfaction rule is used to judge whether the concept schema is complete, and to avoid the abuse of data and abilities in the process. Entities, abilities, attributes, and relationships that are not defined in the concept schema should not appear in the process of a crossover service.

B. CoSM Process Model

In CoSM-PM, we extend the traditional state from a static data state of the process or process milestone to a state space and name it as scenario. CoSM processes are defined to be consist of several scenarios, and each scenario contains a specific activity execution sequence. At the same time, we replace the traditional gateway by defining the triggers, which would bind to the scenario, to complete the routing between the processes.

Definition 7. *A process is a 6-tuple p = (P, E, $\mathcal{J}$, $\mathbb{Q}$, b, F), $P \in \mathbb{P}$ is the identifier of the process, $E \in \mathcal{E}$ is the main entity of the process and $\mathcal{J} \subseteq \mathcal{E}$ are participants. $\mathbb{Q}$ is a ordered finite set of scenarios and $b \in \mathbb{Q}$, $F \subseteq \mathbb{Q}$ are initial, final scenarios (resp.).*

Example 2. In the Taobao transaction process, the main entity is order. Under the operation of the customer, seller, etc., it will experience uncreated, unpayed and other states. The order should start from uncreated state and end with received (see Table III).

As shown in Definition 7, CoSM-PM is designed to describe the business logic and the lifecycle of the main entity. Here, state is wrapped as an identifier into the scenario. A process example based on Taobao transaction is given in Example 2.

Table III: Example of Taobao Transaction Process

process: Taobao Transaction	
main entity	Order
participants	Customer, Seller, Taobao, Alipay, Cainiao
scenarios	uncreated, unpayed, unconfirmed, undelivered, unarrived, unreceived, received
start	uncreated
end	received

Definition 8. *A scenario is a triple q = (Q, $\mathcal{A}$, $\mathcal{T}$), Q is the state of the scenario and also its identifier, $\mathcal{A}$ is a ordered finite set of activity names. $\mathcal{T}$ is a set of business rules, which are defined as triggers below, that assign conditions to $\mathcal{P} \in \mathbb{P}$.*

Example 3. There is a scenario called unpayed in TB transaction process. At the same time, unpayed is also the state of the order in the scenario. Under this scenario, the order will go through three activities in sequence. Within the scenario, order abort process will be invoked as long as customer refuses to pay or Alipay deduction fails (see Table IV).

In CoSM-PM, we extend the traditional state point from the milestone to the state space and name it as scenario. In the same scenario, the state of the main entity is the same. Each scenario has a fixed activity execution sequence, and gateways are replaced by the triggers bound to the scenario. An example of scenario is shown in Example 3.

Table IV: Example of Unpayed Scenario

scenario: unpayed	
state	unpayed
activity	Payment to Platform, Show prepayment information, Frozen capital
trigger	if DONE(Payment to Platform, Cutomer, refuse) OR DONE(Frozen capital, Alipay, failed), invoke process::abort_before_prepay

Definition 9. *An activity is a 5-tuple α = (A, I, J, θ, ϵ), where $A \in \mathcal{A}$ is the identifier, $I \in \mathcal{I}$ is the ability involved in the activity, $J \in \mathcal{J}$ is the participant involved in α, θ is the pre_condition of the activity, and ϵ is the effect of the activity.*

Example 4. Frozen capital activity can freeze part of the customer's capital in a third-party platform (Alipay) to ensure the smooth progress of the transaction. The ability used is $ability : Frozen\_capital$(see Table II). The participant who performs this operation is Alipay. And it can only be done if the customer agrees to pay. After execution, the customer's capital will be deducted and the status of the order will be changed to unconfirmed (see Table V).

The activity of CoSM-PM consists of ability, participant, precondition and effect. We integrate the input and output of the activity into the ability, which decouple the business ability for reuse. In addition, the modeling of the ability and corresponding relations in the CoSM-CM can further enhance the reasoning of the process. Activity example is shown in Example 4.

Table V: Example of Frozen Capital Activity

activity: Frozen Capital	
ability	ability:frozen_capital
participant	Alipay
pre_condition	DONE(Payment to Platform, Cutomer, agree)
effect	customer.capital = customer.capital - order.price, order.state = unconfirmed

Definition 10. *A trigger is a quad t = (Θ, $\mathbb{L}$, κ,* **TYPE***), Θ is a set of conditions, $\mathbb{L} \subseteq \mathbb{P} \bigcup \mathbb{Q}$ is the set of the process*

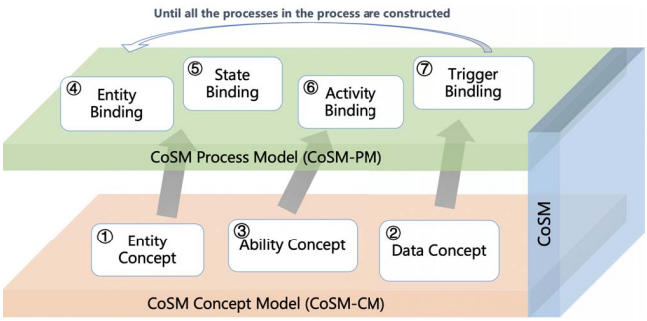


Figure 3: Guideline for Constructing a New Service

and scenario names, κ is a mapping that assign each $\theta \in \Theta$ to $l \in \mathbb{L}$. And the **TYPE** *could be Exclusive / Inclusive / Complex / Parallel.*

Definition 11. *A condition θ is one of the following:*

Boolean/ BEFORE*(α)/* AFTER*(α)/* AT*(α)/* DONE*(α, J,* RES*)/* EST*(a_1, a_2,* RELA*)/ θ_1 AND θ_2/ θ_1 OR θ_2/ NOT θ.*

BEFORE(α)/ AFTER(α)/ AT(α) means the condition is true iff the current activity precedes/succeeds/is activity α. DONE(α, J, RES) means the condition is true iff the activity α is executed by participant J and got result RES. In EST(c_1, c_2, RELA), c_1, c_2 are two concepts and RELA is a relation.

IV. Case Study

In this section, we will use the Taobao trasaction service as an example to show how our model works. We assume that it is the first service of Alibaba and construct it with no exist resource. The guideline for constructing a new service by CoSM is shown in Figure 3.

First, we need to constructe the CoSM concept module.

1) Determines the entities involved. Besides the entity itself, the relations among entities are constructed at the same time.
2) Bind entity attribute. In this step, the attributes of entities need to be determined. E.g., entity order has attributes such as order_id, amount, price, etc. And the properties of these attributes are also determined. In general, the main entity of the service owns more properties than the others.
3) Introduces the abilities used in the service. The abilities and their relations with entities are all constructed within this step.

After performing those three steps, the concept part of the service should be built (see Figure 4a). Then we can begin to construct the process part of the service.

4) Entity binding: Determine all entities involved in the current process, and determine the main entity. The main entity in TB Tansaction is order, and the entities involved in order operation include Cainiao, Alipay, Taobao, seller and customer.
5) State binding: All the states that the main entity has experienced in the business should be determined. The number and order of the scenarios are settled at the same time. Within the critical path of TB transaction, the states include uncreated, unpayed, unconfirmed, and etc.
6) Activity Binding: Determines the activities that the main entity needs to experience before reaching the next state in the corresponding scenario. Abilities and participants assembled with the activities should be confirmed as well.
7) Trigger Binding: Bind the exception handling processes to the scenario of the process via the triggers. Exception handling process is usually built by 4 to 7 steps after it is named and bound.

Based on the service concept schema constructed by 1 to 3 steps, we build CoSM processes through 4 to 7 steps until all the processes in the service σ are defined. And the CoSM processes constructed are shown in Figure 4b.

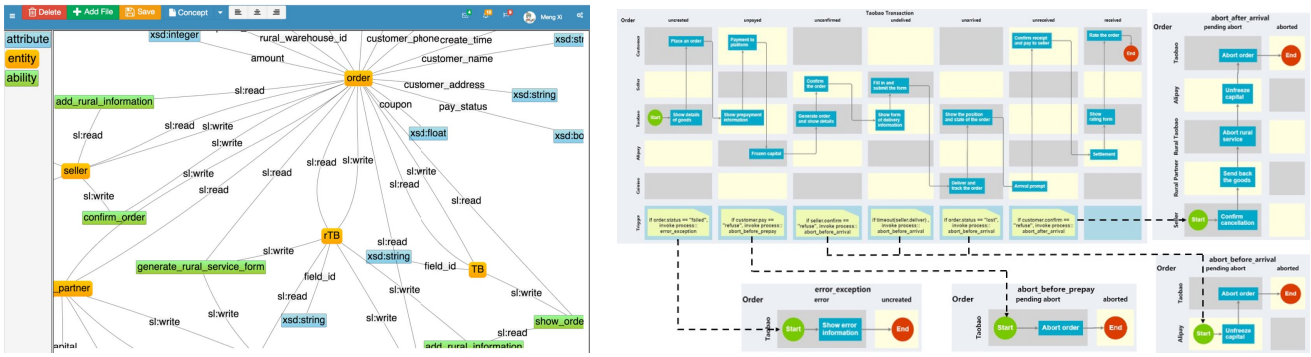
V. Discussion & Comparison

A. Qualitative Discussion

In this section, we compare CoSM with BPMN2.0 and Artifact BPM (see Table VI), and combine the three problems mentioned in Section II to illustrate CoSM's main innovations.

Firstly, to solve the semantic problem, we construct CoSM-CM to model the concepts and their relationships. The concept of ability is introduced to manage services, interfaces, and functions from different domains. The relationships of sl:write, sl:read, sl:own are introduced to facilitate the management of invocation, acquisition, and ownership between entities and abilities. To support automatic reasoning about the entities, data, abilities, and their relationships, subsumption rule, synonymous rule and satisfaction rule are introduced. On the one hand, through the subsumption rule and the synonymous rule, the affiliation between entities in different fields can be determined during crossover service construction, and the problem of polysemy or structural redundancy can be solved. On the other, by the satisfaction rule, the entity and ability boundaries in the crossover service process become clear, and the ability abuse caused by the process integration of different domains can be avoided. Beyond that, with the existing knowledge fusion method, we can complete coreference resolution and entity disambiguation to overcome most of the semantic problems.

Secondly, we solve reuse problem by introducing triggers and scenarios into CoSM-PM. The main cause for the reuse problem is the process complexity caused by process superimposition. It makes the original process difficult to analyze and makes the business logic of the superimposed process even more confusing. CoSM-PM, on the one hand, decomposes a complex process into several simple processes



(a) Graphic Interface of CoSM-CM

(b) Graphic Interfaces of CoSM-PM

Figure 4: Taobao Transaction Constructed by CoSM (part).

Table VI: Comparison between CoSM and other modeling methods

	BPMN2.0	Artifact BPM	CoSM
Data	basic data object for processes	integrated into entity	integrated into entity
Entity	none	pairwise data name and attribute	relational inferential data modeling
Participant	partner object	none	entities that can manipulate process data
Ability	basic functions and services	integrated into activity	relational inferential ability modeling
Constraint	none	none	constraints among entities, attributes, and abilities
Activity	work performed within process	actions between states	container of ability
Process	business process	lifecycle of entity	process graph linked by the triggers and composed of scenarios
Business rules	implicit in the gateway	implicit in the gateway	extracted separately as triggers

by trigger, on the other hand, increases the flexibility of process reuse through scenario.

Thirdly, CoSM solves the hidden conflict problem by formal verification and combining process reasoning with CoSM-CM. In traditional process models, business rules are hidden in process logic. CoSM enumerates business rules by trigger, which makes business analysis more intuitive and avoids business logic conflicts to some extent. At the same time, combined with the CoSM-CM, CoSM-PM can perform process reasoning, with the help of relationships among concepts, to avoid data conflict and ability conflict.

Moreover, CoSM show advantages in the operation, maintenance and upgrade of crossover services. Crossover services need significant logs to help with maintenance. When the exception occurs, an artificial log analysis is needed to find the broken point of the process and its responsible department. In CoSM, exceptions can be more accurately recorded in which service, sub-process and scenario. Besides,through the relationship between sl:write and sl:own, the user who caused the exception and the provider of the business ability can be quickly and automatically located.

B. Quantitative Analysis

In this part, we analyse the quantitative complexity of a process by comparing the number of transitions needed for the classical modeling method and CoSM-PM method. Transition refers to lines among activities, gateways, states, start and end points, which are represented by T. And $|T|$ indicates its number. Here we assume that all activities and states are single-in single-out, all gateways are single-in multi-out.

Suppose there is a process p of service s, the number of activities is $|\alpha|$, the number of gateways is $|g|$, and the number of states is $|q|$. $|exc_{g_i}|$ indicates the exception number introduced by gateway i. To calculate the number of transitions $|T|$, we need to calculate $|T_{activity}|$, $|T_{gateway}|$, $|T_{state}|$, $|T_{start}|$ starting from activity, gateway, state, and start point respectively. The number of transitions can be calculated as Equation 1.

$$|T| = |T_{activity}| + |T_{gateway}| + |T_{state}| + |T_{start}| \quad (1)$$

Constructed by classical process modeling method, $|T_{activity}| = |\alpha|$, $|T_{gateway}| = |g| + \Sigma_{i=1}^{|g|}|exc_{g_i}|$, $|T_{state}| = |q|$, $|T_{start}| = 1$. Equation 1 can be transformed into Equation 2.

$$|T_{classical}| = |\alpha| + |g| + \Sigma_{i=1}^{|g|}|exc_{g_i}| + |q| + 1 \quad (2)$$

Built by CoSM, the number of trigger is $|t|$, $|T_{activity}| = |\alpha|$, $|T_{gateway}| = 0$, $|T_{state}| = 0$, $|T_{start}| = 1 + \Sigma_{i=1}^{|t|}|exc_{t_i}|$. And the $|exc_{t_i}|$ indicates the exception number introduced by trigger i. The transaction number should be calculated as Equation 3.

$$|T_{CoSM-PM}| = |\alpha| + \Sigma_{i=1}^{|t|}|exc_{t_i}| + 1 \quad (3)$$

Since the trigger optimizes the connection of multiple activities to the same gateway, $|t| \leq |g|$ and $\Sigma_{i=1}^{|t|} t_i^{exception} \leq \Sigma_{i=1}^{|g|}|exc_{g_i}|$. Therefore, as shown in Equation 4, CoSM-PM uses less transition when expressing the same process in most cases. And Equation 4 equals to 0 only when there is no gateway and state point in the process.

$$\begin{aligned} &|T_{classical}| - |T_{CoSM-PM}| \\ =&|g| + |q| + (\Sigma_{i=1}^{|g|}|exc_{g_i}| - \Sigma_{i=1}^{|t|}|exc_{t_i}|) \\ \geq&|g| + |q| \geq 0 \end{aligned} \quad (4)$$

In practice, activity, gateway, and state are multi-in, which is an optimization mechanism, to merge the same points. The actual number of transition is less than that in theory. In the case of Section II, $|T_{classical}|$ and $|T_{CoSM-PM}|$ are 46 and 29 for TB transaction respectively.

C. Experimental Results

In this section, we investigate the connector numbers of CoSM and BPMN based on an existing dataset of IBM [2]. The process model of CoSM is a BPMN extended variant, so the BPMN processes can be converted into incomplete CoSM processes. And that is enough for us to compare connector numbers of the two models. We filter the processes more than 75 nodes as there are not enough processes in this range for statistics. In the experiment, we transform the process written in BPMN standard into CoSM process through automatic script. Then compare the number of connectors used by the same process under different modeling methods. All experiments were evaluated on a machine with Intel(R) Core(TM) i5-4200M CPU @ 2.50GHz 2.50GHz, running JDK 1.8 and Windows 10.

As shown in Figure 5, for the experiment dataset, when describing the same process, the number of connectors required by CoSM is less than BPMN, even if the triggers are also calculated as connectors. In the case of trigger calculated, the number of connectors used by CoSM is 14.11% less than that of BPMN on average. The percentage goes up to 23.97% when only links between activities in CoSM are calculated. This shows that our model can greatly reduce the complexity of the process in practical application.

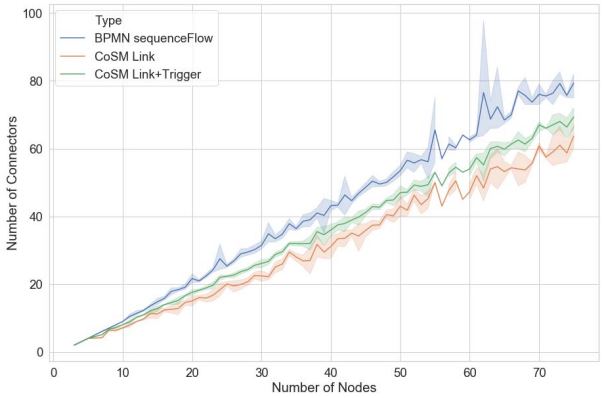


Figure 5: Comparison of the number of connectors in different modeling methods.

VI. Related Works

A. Ontology & Knowledge Graph

To our knowledge, the research related to knowledge graph can be traced back to the semantic network studied by Richard et al. in 1956, which is a data structure based on graph structure storage knowledge [3]. The node and edge of the graph structure represent concepts and their relationships, respectively. Ontology and RDF, RDFs, OWL, etc., which we are now familiar with, are all developed on the basis of the semantic network. Conceptual graph is another import branch in this domain that are proposed in IBM and used for for reasoning, knowledge representation, and natural language semantics [4], [5].

In 2012, in order to improve the performance of its search engine, Google proposed the concept of Knowledge Graph(KG). Based on the knowledge graph, through semantic understanding, it is possible to integrate massive amounts of knowledge on the web and optimize the final search results [6]. Formal definition and analysis are studied and proposed as a basis for further discussion [6], [7]. Besides, there are also some other open KGs such as DBpedia, YAGO, Freebase, and etc [8]. In order to find the most suitable knowledge graph under the given setting, Färber et al. proposed a framework to deal with the issue [9].

In addition, in order to improve the interoperability between heterogeneous systems, knowledge graph and its reasoning ability are also widely used in the development of mediators. For example, Bennaceur et al. proposed an automatic reasoning method of component interface mapping based on ontology reasoning and constraint programming [10]. On the basis of ensuring the semantic compatibility between operations and interface data, these mappings synthesize a mediation to coordinate the calculation mapping, so as to make the components interact correctly. Autili et al. Also proposed a framework for the synthesis of mediators, which is used to describe the existence and correct conditions of mediators and establish the applicability boundary of the synthesis method [11]. Besides, bouloukakis et al. Also proposed an automatic integrated solution of protocol intermediaries supporting interconnection of heterogeneous things based on data exchange (DEX) connector model [12]. These methods usually transfer and negotiate data between different domains through mediators. Although they can solve the interaction problems between heterogeneous systems, they can not meet the needs of crossover services for high integration of multi-domain processes, entities,

abilities and resources.

These works focus more on KG's application scenarios in the Internet such as search and semantic reasoning and have achieved remarkable results. In this work, we focus on combining KG with BPM. By extending and customizing the generic KG, it can better help business staff to conduct business process modeling, process reasoning, and correlation detection.

B. Business Process Modeling

At present, the most popular business process modeling methods are mainly divided into two categories: activity-centric process modeling and data-centric process modeling. In activity-centric BPM, the most well-known one should be BPMN. BPMN was first proposed in 2004 as an abbreviation of Business Process Modeling Notation [13]. In 2010, its official version BPMN2.0 came into being, renamed Business Process Model and Notation, to solve the problems of storage, exchange and execution [14]. In addition, researchers have also extended the different aspects of BPMN's suitability, security, data quality and so on [15]. At the same time, BPEL, which can describe the information exchange protocols between the functions, services and activities of each web in the business process, has quickly become another important standard for acticity-centric BPM [16]. A variety of variants, such as WS-BPEL2.0, BPEL4Chor, etc., have also been proposed and researched to further improve and enhance BPEL's web service interaction and choreography [17], [18]. Ferreira et al. provide a dynamic approach for capturing, processing, and storing event-handling environments and implement business processes as WED-flows [19].

One of the representatives of data-centric BPM is artifact-centric BPM. In the BPM field, artifacts are defined as a concrete, identifiable, self-describing block of information. On this basis, researchers have comprehensively analyzed its concept and constructed the corresponding language model [20]–[22]. Service models, like data-centric web service model, artifact model for agile methods, and MeCo-TSM, are also proposed to improve certain aspects [23]–[25]. Further studies on model verification, incentive modeling, and collaboration support are studied and presented as well [26]–[30].

The choreography in BPMN2.0 standard can represent the interaction between system components and related global constraints. In order to provide the formal guarantee of correctness and termination, the technology of identifying synchronous services, and the technology of automatically repairing non realizable choreography have been proposed [31], [32]. To represent a possible implementation of the distributed system, design methods and formal frameworks to check the consistency of choreography and collaboration have also been proposed [33]. In 2018, Autili et al. proposed a formal method to realize the realizability of choreography by automatically synthesizing [34]. Besides, a new method of service choreography synthesis was defined and implemented to automatically generate code template blanks without considering all message flows [35]. Although these methods can solve the practical problems of choreography, due to its own limitations, choreography can only express the message flow interaction between different systems, and can not meet the deep integration needs of crossover services.

VII. Conclusion

In this work, we proposed a crossover-oriented service model (CoSM), which is composed of concept model and process model, to solve the semantic ambiguity, business coupling, and test explosion problem appeared in crossover services. We also developed a prototype system with GUI to help PM work with CoSM.

At present, although the CoSM process can be automatically converted through scripts, the modeling of CoSM-CM needs to be completed manually, especially the concept of business ability. In the future, we plan to further study the CoSM concept model, equip it with methods including concept acquisition, concept fusion, concept processing, etc.

Acknowledgment

This work is supported by the National Key Research and Development Program of China(No.2017YFB1400601), National Natural Science Foundation of China under Grant (No.61825205, No.61772459), National Science and Technology Major Project of China(No.50-D36B02-9002-16/19).

References

[1] J. Yin, B. Zheng, S. Deng, Y. Wen, M. Xi, Z. Luo, and Y. Li, "Crossover service: Deep convergence for pattern, ecosystem, environment, quality and value," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 1250–1257.

[2] D. Fahland, C. Favre, B. Jobstmann, J. Koehler, N. Lohmann, H. Völzer, and K. Wolf, "Instantaneous soundness checking of industrial business process models," in *International Conference on Business Process Management*. Springer, 2009, pp. 278–293.

[3] D. Spinellis and K. Raptis, "Component mining: a process and its pattern language," *Information & Software Technology*, vol. 42, no. 9, pp. 609–617, 2000.

[4] J. F. Sowa, "Conceptual graphs for a data base interface," *Ibm Journal of Research & Development*, vol. 20, no. 4, pp. 336–357, 1976.

[5] ——, "Conceptual graphs," *Foundations of Artificial Intelligence*, vol. 3, pp. 213–237, 2008.

[6] L. Ehrlinger and W. Wöß, "Towards a definition of knowledge graphs." in *SEMANTiCS (Posters, Demos, SuCCESS)*, 2016.

[7] S. Ferré, "A proposal for extending formal concept analysis to knowledge graphs," in *International Conference on Formal Concept Analysis*. Springer, 2015, pp. 271–286.

[8] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer *et al.*, "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[9] M. Färber, F. Bartscherer, C. Menne, and A. Rettinger, "Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago," *Semantic Web*, no. Preprint, pp. 1–53, 2016.

[10] A. Bennaceur and V. Issarny, "Automated synthesis of mediators to support component interoperability," *IEEE Transactions on Software Engineering*, vol. 41, no. 3, pp. 221–240, 2014.

[11] M. Autili, P. Inverardi, R. Spalazzese, M. Tivoli, and F. Mignosi, "Automated synthesis of application-layer connectors from automata-based specifications," *Journal of Computer and System Sciences*, vol. 104, pp. 17–40, 2019.

[12] G. Bouloukakis, N. Georgantas, P. Ntumba, and V. Issarny, "Automated synthesis of mediators for middleware-layer protocol interoperability in the iot," *Future Generation Computer Systems*, vol. 101, pp. 1271–1294, 2019.

[13] S. A. White, "Introduction to bpmn," *Ibm Cooperation*, 2004.

[14] R. Dijkman, J. Hofstetter, and J. Koehler, *Business Process Model and Notation*. Springer, 2011.

[15] A. Rodrguez, A. Caro, C. Cappiello, and I. Caballero, *A BPMN Extension for Including Data Quality Requirements in Business Process Modeling*. Springer Berlin Heidelberg, 2012.

[16] M. Mongiello and D. Castelluccia, "Modelling and verification of bpel business processes," in *Model-Based Development of Computer-Based Systems and Model-Based Methodologies for Pervasive and Embedded Software, 2006. MBD/MOMPES 2006. Fourth and Third International Workshop on*. IEEE, 2006, pp. 5–pp.

[17] B. Charlton, B. Vaughn, E. Thomas, E. John, J. Diane, K. Khanderao, K. Dieter, M. Simon, S. Ralph, and T. H. Ron, "Web services business process execution language version 2.0 (primer)," *Medicina*, vol. 44, no. 1, pp. 64–71, 2007.

[18] G. Decker, O. Kopp, F. Leymann, and M. Weske, "Bpel4chor: Extending bpel for modeling choreographies," in *IEEE International Conference on Web Services*, 2007, pp. 296–303.

[19] J. E. Ferreira, O. K. Takai, S. Malkowski, and C. Pu, "Reducing exception handling complexity in business process modeling and implementation: The WED-flow approach," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6426 LNCS, no. PART 1, pp. 150–167, 2010.

[20] A. Nigam and N. S. Caswell, "Business artifacts: An approach to operational specification," *Ibm Systems Journal*, vol. 42, no. 3, pp. 428–445, 2003.

[21] K. Bhattacharya, C. Gerede, R. Hull, R. Liu, and J. Su, "Towards formal analysis of artifact-centric business process models," in *International Conference on Business Process Management*, 2007, pp. 288–304.

[22] D. Cohn and R. Hull, "Business artifacts: A data-centric approach to modeling business operations and processes," *IEEE Data Eng Bull*, no. 3, pp. 3–9, 2009.

[23] R. Vaculn, T. Heath, and R. Hull, "Data-centric web services based on business artifacts," in *IEEE International Conference on Web Services*, 2012, pp. 42–49.

[24] M. Kuhrmann, D. M. Fernndez, and M. Grber, "Towards artifact models as process interfaces in distributed software projects," in *IEEE International Conference on Global Software Engineering*, 2013, pp. 11–20.

[25] Y. Li, M. Xi, Y. Yin, Z. Luo, H. Gao, and J. Yin, "Mecotsm: Multi-entity complex process-oriented service modeling method," in *2018 IEEE International Conference on Web Services (ICWS)*. IEEE, 2018, pp. 82–90.

[26] D. Calvanese, M. Montali, and E. Teniente, "Verifiable uml artifact-centric business process models," in *ACM International Conference on Conference on Information and Knowledge Management*, 2014, pp. 1289–1298.

[27] O. Scekic, H. L. Truong, and S. Dustdar, "Supporting multilevel incentive mechanisms in crowdsourcing systems: An artifact-centric view," 2015.

[28] M. Estaol, M. R. Sancho, and E. Teniente, *Verification and Validation of UML Artifact-Centric Business Process Models*. Springer International Publishing, 2015.

[29] Y. Sun, J. Su, and J. Yang, "Universal artifacts: A new approach to business process management (bpm) systems," *Acm Transactions on Management Information Systems*, vol. 7, no. 1, p. 3, 2016.

[30] M. L. V. Eck, N. Sidorova, and W. M. P. V. D. Aalst, "Guided interaction exploration in artifact-centric process models," in *Business Informatics*, 2017, pp. 109–118.

[31] S. Basu and T. Bultan, "Choreography conformance via synchronizability," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 795–804.

[32] ——, "Automated choreography repair," in *International Conference on Fundamental Approaches to Software Engineering*. Springer, 2016, pp. 13–30.

[33] F. Corradini, A. Morichetta, A. Polini, B. Re, and F. Tiezzi, "Collaboration vs. choreography conformance in bpmn," *arXiv preprint arXiv:2002.04396*, 2020.

[34] M. Autili, P. Inverardi, and M. Tivoli, "Choreography realizability enforcement through the automatic synthesis of distributed coordination delegates," *Science of Computer Programming*, vol. 160, pp. 3–29, 2018.

[35] M. Autili, A. Di Salle, F. Gallo, C. Pompilio, and M. Tivoli, "Aiding the realization of service-oriented distributed systems," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 1701–1710.