Service Language Model: New Ecology for Service Development

Ying Li
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
cnliying@zju.edu.cn

Meng Xi
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
ximeng@zju.edu.cn

Hui Chen
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
chchenhui@zju.edu.cn

*Jianwei Yin
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
zjuyjw@zju.edu.cn

Abstract—**With rapid development of the Internet, all walks of life are engaged in the tide of Internet. In the era of "Internet+", traditional industries are widely developed and expand plenty of emerging business, such as online transactions, Internet finance and so on. However, various problems arise at the same time in this revolution. On one hand, business processes become increasingly intricate, and different fields may have difficulty in communication. On the other hand, developers are hard to understand the real demands from users and the rate of code reuse is not high. In order to solve these problems, we propose a middle-end and project manager (PM) oriented service language model, which could help decouple software development and user requirements, improve work efficiency and reduce development costs.**

Keywords-service language model, ontology, business process

I. INTRODUCTION

Nowadays, modern service industry has grown up rapidly and play an important role in our daily life. As service scenarios become increasingly complicated and user demands change quickly, traditional business process modeling methods can hardly meet the current business requirements.

The problem could be serious in the field of E-commerce. Different from offline sales, business processes in E-commerce develop rapidly. To cope with this situation, enterprises need to invest a lot of resources to improve their original business processes. Therefore, it is necessary to design a process modeling method that can quickly and accurately respond to requirements. And in recent years, data-centric Artifact model has emerged. Different from traditional activity-centric modeling approaches, the construction of Artifact model begins with data, and then business processes are built around the data entities. By using data-centric approach, the business process could be flexible and adaptable. However, the data-centric life cycle could be difficult to reuse and easily lead to code redundancy. On the other hand, the existing demand analysis method is usually activity-centered, which makes it difficult to put Artifact modeling into practice.

In this paper, we design a service language model (SLM). SLM consists of Service Concept Model (SCM) that can describe the requirements of services and Service Design Model (SDM) that can construct and manage service processes. Within SDM, there is an entity feature model for data, an ability feature model for functions and interfaces and a refined life cycle model for business processes.

The main contributions of SLM are as follows. First, a concept model of service is designed to conceptualize requirements and describe them in a formal way. Secondly, a method was provided to manage and deploy data and data entities in services. Thirdly, Business abilities can be reused and their range can be cleared easily by using SLM. Finally, a process model which could help stratify the business and clarify the business structure was proposed.

The rest of this article is organized as follows. Section Two is the motivation case. Section Three introduces SLM systematically. Section Four illustrates an application instance where the model is verified. Finally, the related works and conclusions are given.

II. MOTIVATION CASE

In order to have more in-depth research and better understanding of the problem, we have worked with Alibaba company. Here, we take buying basketball shoes and virtual coupons as an example. One is physical goods trading which is one of the earliest services of Alibaba, and the other is virtual goods trading which is emerging these years. The former one involves goods delivery, while the latter only needs to send some verification codes. Since the physical properties of these two commodities are different, their service processes should be differentiated. Moreover, when modeling, we may encounter the following questions.

Problem 1. Business logic is coupled with platform logic, and process activity code is difficult to strip and reuse. This makes the platform code difficult to analyze and organize. With the business becoming more complex, platform codes may have redundancy;

Problem 2. Long implementation cycle. A simple need also requires steps such as program evaluation, requirement analysis, process design, process development, and regression testing;

Problem 3. Internal business units lack knowledge of business uniformity, communication between departments can cost a lot of time.

*: corresponding author

DOI reference number: 10.18293/SEKE2018-214

III. Service Language Model

In this chapter, we will introduce our service language model systematically. The main concepts in service concept model include entity rules and routing rules, and the main concepts in service design model include entity, ability, essence process, and service process. For a better understanding of the definitions below, we expect the existence of the following pairwise disjoint countably infinite sets: Tp of primitive types, C of (artifact) classes (names), A of attributes (names), S of artifact states, and ID_C of (artifact) identifiers for each class C. A type is an element in the union T = Tp ∪ C.

A. Service Concept Model

As we all know, requirement analysis has always been extremely important in software engineering, as its quality directly affects the effectiveness of the later system design. In order to make requirement analysis more concise and efficient, we build the concept of service model.

Definition 1. *An service concept model instance is a triple (sc, Rk, Rr), where sc is the identifier, Rk is the entity concept model involved and Rr is the routing concept model.*

The entity concept model is mainly used to specify the terms involved in the service. It can express the domain terminology, attributes of each term, and links among different terms. In addition, the entity concept model can generate the entity feature model by semi-automatic methods.

Definition 2. *An entity concept class is a four-tuple (rkc, k, d, re), where rkc is the identifier, k is the set of term names, d is the set of term descriptions and re is the set of relationships among terms.*

Definition 3. *An instance of an entity concept is a 3-tuple (rk, kd, krk) where rk is an identifier, kd is a partial mapping from k to d, and krk is a set of relations mapping between terms by means of "k-re-k".*

Example 1. *In the motivation case, business side needs to provide the requirement of data that may be used in their service, like receiving information which consists of name, phone and address. They could describe this requirement in the format of service language model as shown in Table I.*

TABLE I. Entity Concept Example: Receiving Information Entity Concept

ID	Entity & Description	Relation

721019	receiving information: data set
	name: string, length<10
	phone: number, length=11
	address: string
		basic(receiving information, name)
		basic(receiving information, phone)
		basic(receiving information, address)

On the other hand, routing rules are mainly used to express the system through a simple logic involved in the activities of the process. The conceptual model of routing mainly guides the construction of the refinement process model, and gives guidance on the construction of the life cycle model.

Definition 4. *A routing concept class is a four-tuple (rrc, S, c, E), where rrc is the identifier, S is the state in the route, C is the set of judgement conditions, and E is the set of posterior effects.*

Definition 5. *A routing concept instance is a pair (rr, sce), where rr is the identifier, and sce indicates that the entity e satisfies the condition c under state s.*

Example 2. *In the motivation case, there are requirements about process itself like how to place an order. The business side needs to arrange the process and describe it in the format of routing concept like Table II.*

TABLE II. Routing Concept Example: Place an Order Routing Concept

ID	sce
217210	(no_order, place an order success, state=to_pay) (to_pay, payoff, balance reduce & state=to_deliver) ...

B. Entity Feature Model

The entity feature model is mainly used to model the data entities in the service. A feature model is composed of a set of characteristics and their relationships, and it also has certain constraints.

Definition 6. *An entity feature class is a 7-tuple (Cε, AT, τ, Q, s, F, Opt), where Cε is the identifier, AT is the attribute in the entity or the characteristic in the entity, τ is the type of the attribute or the constraint relation of the class or feature, Q is the set of states of the entity, s is the initial state of the entity, F is the end state, and Opt is the optionality of the state.*

Definition 7. *An entity feature instance is a triple (e, μ, q) where e is the identifier, μ is partial mapping that assigns each AT in T (mentioned at the begging of section three), q is current state when e is an full entity or Opt when e is a feature of an entity.*

Example 3. *In the motivation case, order is an important entity. The process of a transaction is the lifecycle of an order which is from generation to extinction. Its instance is presented in Table III.*

TABLE III. Entity Concept Example: Receiving Information Entity Concept

ID	μ	state

171701	receiving information: data set
	name: string, length<10
	phone: number, length=11
	address: string
		basic(receiving information, name)
		basic(receiving information, phone)
		basic(receiving information, address)

When analyzing the transaction process, we find that there are entities involve in a process with no state, which are called participants.

Definition 8. *A participant class is a triplet (Cp, RE, τ) where Cp is the identifier, RE is the resource of the participant, and τ is the main type of the resource.*

Definition 9. *A participant instance is a binary (p, μ), P is the identifier, μ is the partial mapping that assigns each RE in T.*

C. *Ability Feature Model*

The modeling method of ability feature model is the same as that of the entity model, which is used to describe the configurable services and functions of Artifact.

Through the construction of the ability model, we can use functional units in the system as configurable atomic services. Through the call to the service, data in entities can be processed and their state may transfer as well.

Definition 10. *The instance of the ability is a six-tuple (ab, D, VEr, VEw, VPr, VPw), ab is the identifier, d is description of this ability, VEr is variable of entity to read, VEw is variable of entity to write, VPr is variable of participant to read, VPw is variable of participant to write.*

Example 4. *In the motivation case, we need an ability of creating an order. This ability need to generate an order and transfer the customer's money to third party platform. We need the order's id and whether the transfer succeed. The ability is presented in Table IV.*

TABLE IV. ABILITY EXAMPLE: ABILITY OF CREATING AN ORDER

order_create	
id	344323
description	create an order and transfer the customer's money
entity	order
participant	customer
entity read	order.merchandise, order.destination, order.message, order.seller
entity write	order.id
participant read	customer.user_id, customer.cash, customer.method
participant write	customer.payment_result

D. *Refined Lifecycle Model*

The refined lifecycle model is mainly used to represent the changes in state and the activities used during the execution of the service. Refinement lifecycle model can stratify the processes in the lifecycle process, decouple the processes in different areas and reduce the complexity of each process while improving the process complexity.

Condition is essentially a discriminant expression, and finally return Boolean values, which could complete some simple calculations and judgments.

Definition 11. *A condition is a Boolean expression, which could be connected and calculated by logical operators.*

L1 process describes the core functions of the business and is used for service classification and location.

Definition 12. *An L1 process is a triple (m, d, RP2), m is the identifier, d is the model description and RP2 is the L2 process inherited from this L1.*

L2 inherits from the L1 process and represents all the states that an entity will experience in a life cycle.

Definition 13. *An L2 and a process is a five-tuple (tr, d, q, r, RP3), tr is identifier of transition, d is description, q is states and r is relation between states.*

The L3 process inherits from L2 and adds activities and gateway nodes on the basis of L2, which can represent all the activities and states that an entity experiences in a process. At the same time in the L3 process we will complete the ability to assemble. We specify each activity to assemble an ability.

Definition 14. *An L3 process is a six-tuple (re, d, q, a, g, r), re is identifier, d is description, q is the states, a is activities, g is gateways and r is relations connecting q, a and g.*

Definition 15. *Activity instance is a four-tuple (a, ab, P, E), a is identifier, ab is identifier of ability, P is pre-condition (atom), and E is effect.*

Definition 16. *Gateway instance is pair (g, type), g is identifier of gateway and type could be one of the four: Exclusive, Inclusive, Complex and Parallel.*

Definition 17. *relation is a four-tuple (r, from, to, c), r is identifier of relation, from / to is state / activity / gateway, and c is a condition.*

Then we can add the activities of the preconditions and configuration items to generate L4 process on the basic of L3 process. And a runnable process is complete.

IV. APPLICATION INSTANCE

Here we take the physical transaction service as an example to explain how to use the service model to construct it. Due to the space limitation, we only make a brief explanation.

A. *Constructing service concept model*

The physical transaction process is mainly around the order, which has a clear state in each stage of the process. The data in the entity can be described as "entity data name + relationship + limit value". For instance, if the entity "order" contains "delivery method", its value should be one of "general delivery", "free shipping", "cash on delivery" and "self-pickup". All the requirements for the entities in the process can be described by the entity concept model.

The process may involve other participating entities as well, like goods and members. In this process, the participating entities do not own state, but the data involved in the implementation of the process. Then we could describe the business process and routing rules through a set of combination of state, condition and effect, such as "unpaid, payment executed, state change to unshipped".

B. *Constructing service design model*

The service design model is a system implementation of the service concept model. From the service concept model to the service design model there is a mapping relationship, and these two models can manually transform into each other semi-automatically.

By reorganizing and rearranging entity concept models, we can construct entity feature models. That is, modeling the entity through feature modeling. We modeled three entities in the physical transaction process respectively including "Order", "Goods" and "Customer".

Then, we encapsulate functions and interfaces that may be used in the process and list the abilities. According to the classification criteria of the e-commerce domain characteristics, the classification of abilities could be performed. An ability feature model is similar to entity feature model. In this way, we can quickly query the ability to operate on the entity data involved in the process and assemble it.

The refined process model is used finally. Firstly, we need to determine which trading domain the business belongs to, and that is L1 process. Then we further determine the states of the order, which could be "unpaid", "unshipped", "assessed" and etc. L2 process is constructed by these states and their transition relationship. L3 need to add activities and gateways which may cause the transition of states. For instance, the transition from "unpaid" to "unshipped" needs to go through activity "payment". The L4 level process is the configuration of the preconditions, transition conditions between activities and other configuration items. Through these operations, a physical transaction business is completed.

V. Related Work

The service language model is based on previous research. We mainly refer to research results of ontology, domain-specific language (DSL), variability modeling and business process modeling (BPM).

The concept of ontology originated from the field of philosophy and later has been given a new definition with the development of computer science. Ontology web language (OWL) is a representative application of ontology in computer science. Different from traditional knowledge-based methods that are represented by formal normative language, OWL is more suitable for regulating demands and domain knowledge [1]. In addition, it has the ability of automatic verification and consistency checks [2]. Nevertheless, OWL only stays in the concept level, which means it is unable to implement a system.

DSL has been recognized as a normative executable language that is highly expressive in specific areas, where domain contents are abstracted [3]. Domain experts can easily design models with the help of DSL. Thus, it is applied to a number of fields. For example, Matlab [4] in the mathematical field can help people deal with vector, matrix and other mathematical operations conveniently. On the other hand, the IBM Sharable Code project also uses DSL to prepare services [5]. However, DSL is only applicable to certain areas, which means it has difficulty in solving cross-domain problems.

Variability modeling is the key technology for variability management in software product lines. In our project, we mainly refer to the feature-based variability modeling method to construct our entity feature model. This method was first introduced in 1990 by Kang KC et al. [6]. In recent years, since feature modeling technique can greatly improve the flexibility and reusability of system, it has been widely used to describe relevant characteristics of software product lines and to manage reusable assets in systems.

There are many business process modeling methods. Business process modeling notation (BPMN) is a basic and practical standard in web service field[7] since it can intuitively express business processes. It defines a variety of elements that may be used in business process modeling, including activities, gateways, events and so on. Business process execution language (BPEL) is an abstract high-level language that can clearly describe functions and services provided by each web, as well as protocols among different activities [8]. However, these methods cannot eliminate the redundancy of large complex systems, since they always focus on business process implementation level.

VI. Conclusion

On the basis of artifact centric process modeling, this research improves the modeling method of data entities to make it more adaptable to the operating environment of large and complex systems. We construct a hierarchical lifecycle model that can help organize and manage the process. Ability feature model is proposed to manage and reuse the code resources more effectively and efficiently. We also modeled requirements into SCM which could help build a domain knowledge and describe requirements in a formal way. At present, we have been able to realize semi-automatic conversion between requirements and design under artificial operation. As for future work, we will introduce natural language processing and machine learning methods to achieve automatic conversion from requirements to code.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant No.61772459, the National Key Research and Development Program of China under Grant No.2017YFB1401202 and the Key Research and Development Program of Zhejiang Province under Grant No.2017C01013.

References

[1] Wouters B, Deridder D, Van Paesschen E. The use of ontologies as a backbone for use case management[C]//European Conference on Object-Oriented Programming (ECOOP 2000), Workshop: Objects and Classifications, a natural convergence. 2000, 182.

[2] Happel H J, Seedorf S. Applications of ontologies in software engineering[C]//Proc. of Workshop on Sematic Web Enabled Software Engineering"(SWESE) on the ISWC. 2006: 5-9.

[3] Van Deursen A, Klint P, Visser J. Domain-specific languages: An annotated bibliography[J]. ACM Sigplan Notices, 2000, 35(6): 26-36.

[4] Hanselman D, Littlefield B. Mastering MATLAB 6: a comprehensive tutorial and reference[M]. Pearson, 2001.

[5] Maximilien E M, Ranabahu A, Gomadam K. An online platform for web apis and service mashups[J]. IEEE Internet Computing, 2008, 12(5).

[6] Kang K C, Cohen S G, Hess J A, et al. Feature-oriented domain analysis (FODA) feasibility study[R]. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 1990.

[7] White S A. Introduction to BPMN[J]. IBM Cooperation, 2004, 2(0): 0.

[8] Fu X, Bultan T, Su J. Analysis of interacting BPEL web services[C]//Proceedings of the 13th international conference on World Wide Web. ACM, 2004: 621-630.