International Journal of u- and e- Service, Science and Technology
Vol.9, No. 11 (2016), pp.55-70
http://dx.doi.org/10.14257/ijunesst.2016.9.11.06

Rediscovering Impacts of Collaboration through Workflow Analysis to Value Service Development

Chengxiang Ren[1], Yucong Duan[2*], Zhangbing Zhou[3], Guohua Fu[4], Zhen Guo[5] and Honghao Gao[6]

[1, 2,5]College of Information Science and Technology, Hainan University, China
[3]School of Information Engineering, China University of Geosciences, China
[4]School of Economics and Management, Hainan University, China
[6]College of Computer Science and Technology,Zhejiang University,Hangzhou,China
[1,2]irhawks,yucongduan@hotmail.com, [3]zhangbing.zhou@gmail.com
[4]fghfz328@163.com,[5]46444972@qq.com, [6]gaohonghao@zju.edu.cn

Abstract

Information and service economies are developing rapidly in recent years with the development of information technology and new applications. Software and service development are becoming more and more complex relating to different participants, organizations and alliances. This phenomenon has caused urgent requirements on economic and technical fusion through coordination and collaboration among stakeholders. This paper addresses the gap between business layer and technical development in roles and activities. We then propose a combination of economic value and profits and technical service development processes. We also consider value exchange under profit sharing contracts in participants' workflows. A simulation and an experiment are performed to show related impacts.

Keywords*: Workflow, Collaboration, Profit Sharing, Service Development*

1. Introduction

In current world's economy, micro, small and medium-sized enterprises (SMEs) need to satisfy depth requirements, participate in external collaborations and further join external collaborative networks to be more innovative and competitive [1]. Since the implementation of Service-Oriented Architecture (SOA) may be complex, costly and risky [2], collaboration that enables service providers to work in an open network becomes attractive. Software is a key factor to increase competitiveness. But software and service are different from manufacturing in areas such as development stages and methodologies, physical deployment, software/service quality, access and usage [2-3]. Traditional economic and management techniques such as product supply chain cannot easily be applied [4]. Since services can be purchased from infrastructure providers or be outsourced to other entities, techniques in service development in SMEs may not be of the first magnitude. Instead of that, development under a more open environments with characters such as complex cross-organization collaboration brings extra challenges to SMEs. During service development progress, a lot domain knowledge and decisions are involved. Feedbacks, reworks and schedules are quite often. Knowledge accumulation and reuse of design are considered to achieve added-value for subsequent use. Since for a service system configurations that affect performance may be determined in the execution phase. The development phase may leave some factors unsolved, which causes uncertainties in functionalities and/or qualities.

ISSN: 2005-4246 IJUNESST

While gaining most added-value is still important, development in SME-like environment is sensitive to cross-organization collaboration. Many development process models stress on the transformation of resources and are not intended to describe value exchange. Development process require negotiations for service contracts as well as the coordination of development activities. Phenomena that are viewed as business issues like information dissymmetry may as well act on service development. This paper firstly tries to identify the gap between business perspective and technical perspective [5] to meet SME-like entities' demands of enhancing their competitiveness. We then dedicate ourselves to value service development. Researchers from economic and management view profit sharing in a social and organizational way. However, the realization of profit sharing is concrete for technical process. We try to introduce profit sharing to service development process in a workflow manner.

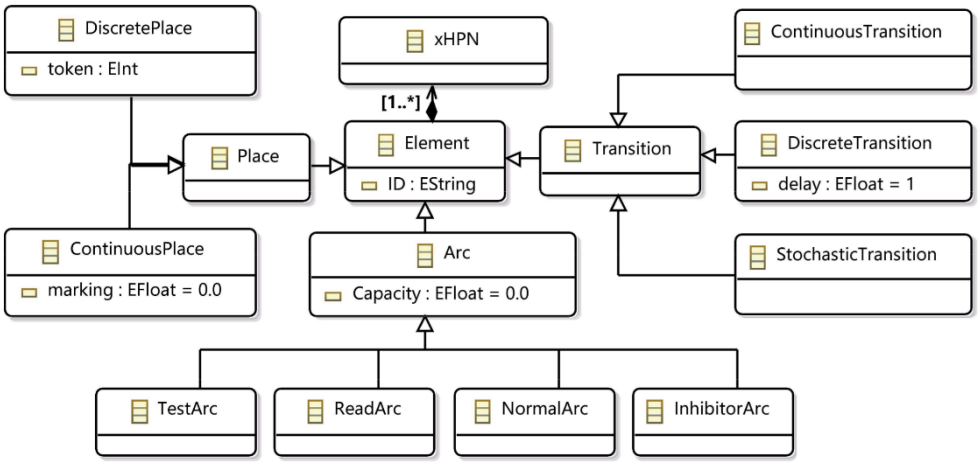
The rest of this paper is organized as follows: Section 2 brings up a convenient and universal process metamodel, the extended Hybrid Petri Net. Section 3 addresses issues in the combination of multi-level process modeling. Section 4 describes a value transferring mechanism combined with existing value frameworks presented in Section 3. Section 5 presents a simulation of value accumulation in service development to explore our modeling and then analyzes a profit sharing scenario of the interaction of an Application Service Provider (ASP) and an Application Infrastructure Provider (AIP) in service lifecycle. Section 6 introduces related areas and literatures. Section 7 summarizes our works and gives our future research directions.

2. Process Modeling Using the Extended Hybrid Petri Net

Process modeling techniques, which are widely used in both the software process and business process, give a symbolic representation of activities, resources and their consumptions. Among them, petri nets and their various extensions are universal modeling methods for representing processes in various application fields in nearly all levels of abstraction. Petri net models can be qualitative or quantitative, discrete or continuous. Many types of petri nets are simple, flexible and expressive to describe various systems. While specific process metamodels are more accurate and consistent for specific fields, we still use an extended petri net to express overall business and technical process to keep simplicity and universality.

A *net* typically contains $\langle P,T;A\rangle$ two kinds of elements namely places P and transitions T and one kind of relations called arcs A. Based on net $\langle P,T;A\rangle$, a *petri net* is equipped with a function $M:P\to$ □ to represent a state (identified by a place's marking or token) of a certain place and a function $W:A\to$ □ to indicate the weight of arcs. Typically, when a transition fires, tokens in output places are generated and tokens from input places are consumed at a certain amount. Additionally, a function named capacity function $C:P\to$ □ is used to indicate the capacity of places. E**x**tended **H**ybrid **P**etri **N**et (xHPN) [6] can be taken as an extension of a petri net with a capacity function and other structures. Figure 1 shows a simple concept model for xHPN. Prob, Bachmann, Janowski, *et. al.*, [6] gave a detailed specification on xHPN. Compared with normal petri nets, an xHPN comprises of

(1) Two kinds of places, namely discrete places P_D and continuous places P_C. Each discrete place has tokens measured by a non-negative integer and each continuous place has marks measured by a non-negative real number.



(2) Three kinds of transitions, namely discrete transitions T_D, stochastic transitions T_S and continuous transitions T_C. Discrete transitions are specified with *delays*. Stochastic transitions replace the fixed delay with a random variable with an exponential distribution. Both discrete and stochastic transitions fire by removing the arc weight from all input places and adding the arc weight to all output places. A continuous transition fires as a continuous flow in a speed depending on arbitrary variables.

(3) Four kinds of arcs, namely normal arcs A_N, test arcs A_T, inhibitor arcs A_I and read arcs A_R. An arc is weighted by a non-negative integer or a real number. Moreover, the weight of an arc can depend on time and/or current markings of a place.

Places connected to transitions using test or inhibitor arcs do not change their markings during firing. In this case, the markings only indicate a usage. If a place is connected to a transition by a test (inhibitor) arc, the marking of the place must be greater (less) than the arc weight to enable firing. Related symbols of xHPN are listed in Table 1. The simplicity in modeling time factors, stochastic transition, hybrid process and transitions without reducing resources makes xHPN more conformable to meet requirements from both the business layer and technical layer.

Table 1. Symbolic Representation for xHPN

Places	Transitions	Arcs
Arbitrary	Arbitrary	Arbitrary
Discrete Continuous	Discrete Continuous Stochastic	Normal Test Inhibitor Read

3. Development Process Under Organizations

3.1. Various Views on Development Process

Table 2, shows levels of an organization and their characters in modeling and controlling an organization's complexity. For a specified organization, the operational layer has more detailed information such as which one, which team or which department will conduct the execution. Other business concerns like supply chain lay between high abstraction and medium abstraction, while business process usually lies in the medium

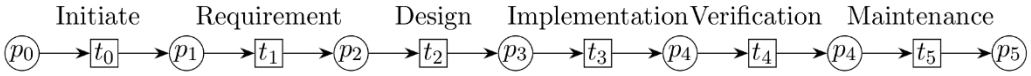
abstraction. The arrangement of development activities is affected by factors coming from different levels.

Development process itself is also hierarchical. In large areas, we have lifecycle models such as the waterfall model, V-model, architecture based development, model-driven development and software product based development. These models are usually considered as reference models for software and service development. Figure 2 separately shows phases and their order in the waterfall model and a service development model. OMG's Software Process Engineering Metamodel (SPEM) 2.0 specification, which is based on UML, describes related elements in a software process. In SPEM's view, a software process is a set of activities, each of which is comprised of one or more tasks performed by an abstract entity called a *role*. Roles are used to describe the responsibilities and a set of skills to facilitate their assignment to proper people.

Table 2. Characters of Different Levels

Abstraction	View	Detail	Level
High	Strategic	Minimum	Macro
Medium	Tactical	Medium	Meso
Low	Operational	Maximum	micro

There are also various lifecycles for service-oriented development. One of them is the Service-Oriented Modeling Framework (SOMF). The framework defines activities of discovering and analyzing, business integration, logic design, conceptual architecting, logical architecting consequently. Daniel and Matera [7] proposed a lightweight development process model specially designed for mashups, a new service development style. The model contains several major activities namely discovery and selection, mashup composition and integration, usage (deployment) and maintenance. We simply use a service lifecycle containing service discovery, selection, composition, deployment and maintenance in consequence. SPEM typically supports activity-oriented modeling. During development more attentions are not paid on which kind of resources to exchange, but how to use relevant resources to accomplish intended activities. This section outlines the difference between above two views and depict several structural aspects on



development activities.

(a) Waterfall Lifecycle for Software Development

(b) Mashup Lifecycle for Service Development

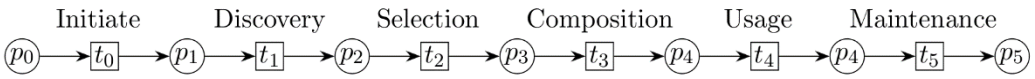


Figure 2. Lifecycle Models

3.2. Phenomena in Service Development

1) Interactions during development. Roles can interact with each other in business networks, communities or organization units, which adds complexities to analyze collaboration. Different roles in development have different interests. For example, designers and programmers work together to implement a service system, but designers value models while programmers value codes. It's applicable to refine activities to ensure that each activity is done by one and only one participant and then combined each participant's workflow to form a larger process.

2) Scheduling factors of software process. By comparing scheduled time with actual time, decision-makers may decide to do some rework, or to cut off some functions. During a deadline-driven software process, scheduling happens at the end of every stage. During a software product line lifecycle, decision making is the main source of design value. It's applicable to introduce stochastic factors in xHPN to model phenomena such as a component costs more or less time than expected. Feedbacks and iterations are in companion with decision-making. The integrated test phase in V-model may cause feedbacks. Code reviews may cause a small-scale feedback.

3) Assets and Knowledge. Software product line engineers use the term ``assets" to describe software production/services. Assets have different usages and evaluation principles. Assets are used in different workflows and by different participants. Classical assets in software development include various documents, codes, test results, software models, investigated knowledge and economic factors such as cost and time.

Software process is knowledge-sensitive [8]. However, current process modeling techniques lack support of knowledge creation, representation and sharing. Knowledge is ``referred to" instead of ``be made up". Typical knowledge in software process includes techniques and languages for modeling, programming and testing. The changes of knowledge in a software process have typical topics such as knowledge applications, training and sharing techniques. The xHPN is applicable to illustrate various usages of knowledge. A usage is shown in Figure 3. Since knowledge management is more and more important to software process, knowledge becomes another view of the software process that all stakeholders must be concerned with. Requirement engineers concern about market trends. Design staffs concern about frameworks and architectures. Programmers concern about programming languages.

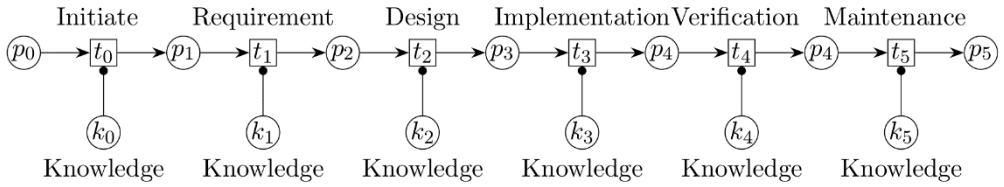


Figure 3. An Illustration on Knowledge Introduction

3.3. Comprehensive Description of Development

Workflow technologies heavily support the automation of today's complex business process. A workflow describes a process where documents, information and tasks are passed from one participant to another according to a set of procedural rules. Workflow model, the basis of workflow execution, can be constructed in a process-oriented or role-oriented manner. Based on the relation among activities and their occurring order, six basic workflow patterns shown in Figure 4 are identified in the petri net. Persons are usually bound to activities rather than to resources in development. A function $Assign: Persons \rightarrow Transitions$ is added to a petri net to describe task allocation.

Top-down, bottom-up and mixed strategy are used to describe a detailed software process. In the top-down manner, we firstly refer to an overall lifecycle model (V-model, product line lifecycle model or model-driven process model), then refine activities and finally assign atomic tasks to relevant roles. In the bottom-up manner, we firstly summarize regular activities to each role, then combine them together and eventually get cross-department or cross-organization activities. The mixed manner combines the two

strategies above. The top-down strategy allows us to conform to proper theoretic lifecycle model and development methodology and to decompose large activities to many small ones. The bottom-up strategy has advantage in describing the capabilities of each role and analyzing interactions and impacts on interactions among participants. The bottom-up strategy is also helpful for modeling decision paths. A participant may take part in many different activities, which makes relevant optimization of a participant more complex. The six patterns listed in Figure 4 are more suitable for the bottom-up method. They are helpful to locate relations, identify basic decision-making patterns and evaluate values and costs according to participants' behaviors.

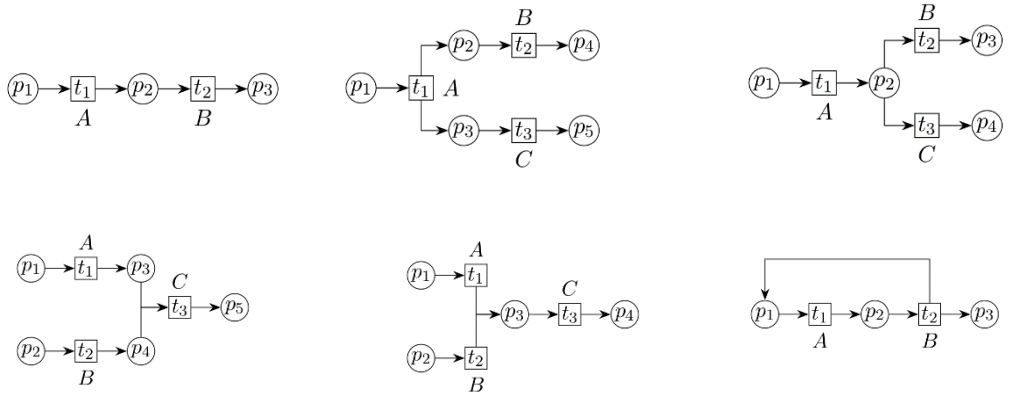


Figure 4. Basic Patterns for Activities

4. Economic and Management Views on Development

We start with the concept of service contracts to cut into the business layer. A service contract is basically a service delivery agreement between a service provider and a service consumer. Different service contracts have different contents. Service contracts between an ASP and a customer specify requirements, solutions to meet requirements and the cost to meet requirements, while service contracts between an ASP and an AIP may additionally specify a collaboration model. A service contract may contain economic information in value distribution and risk evaluation, participant's information in their roles, collaboration and workflows. A service contract may also give technical information in functionalities, quality attributes and execution process.

It's known that a contract may be incomplete for each participant's private information. A service contract clearly specifies roles and their activities, benefits if they obey the rules and punishments if they break the rules. Moreover, profit distribution, and possible risks are specified. In supply chain related contracts, we usually deal with concepts such as roles, resources, profit model and profit sharing. For purely technical aspects on software development process, roles and activities are different. A service contract usually refers to value patterns in creation, exchange, collaboration and distribution. As with the relation between service contract and development process, the latter may not only refer supply chains in the former, but also information on lifecycle models, structure models and decision models. Dynamic value and static value are identified according to their calculation. Static values are values that have no relation with the workflow process, and can be depicted as attributes of places, transitions, or arcs in petri nets. Dynamic values reflect the impact of change in petri net structures. In the value-related development process, a stakeholder in development may also act an economic stakeholder. Resources are related to attributes and profit sharing contracts are related to decision models.

4.1. Value Exchange and Value Evaluation

Evaluating resources through exchange is common to see. Value exchange is a process that involves at least two participants and at least two forms of resources. Resources and events form the basis of a value chain in the Resource-Event-Agent (REA) framework [9]. The REA model includes agents, their resources and economic events through which agents exchange resources. Economic exchanges are limited to instances between just two trading partners in REA [10]. Products, services, financials and contents are identified as four types of resources in [11]. However, the concept of resources can be extended to other objects such as requirements, infrastructures, platforms, hardwares, designs, requests and database to enrich the value [12]. Exchange may happen within or beyond organizations or departments. If exchange happens between individual economic subjects, the condition holds as

$$V_i(\sum Resource_{in}) < V_i(\sum Resource_{out}) \quad (1)$$

for each stakeholder i, which means each stakeholder gets more (expected) value through exchange.

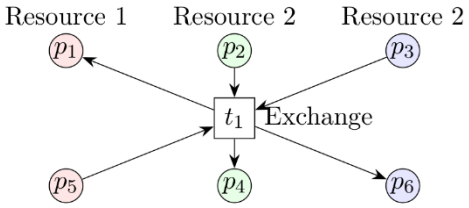


Figure 5. Resource Exchange

The e^3-value framework [13] depicts a networked business model where business partners exchange things of tangible and intangible economic value. Valuable objects to a participant are called value objects. Value objects refer to things including products, services, financial or consumer experiences that each participant exchanges. In e^3-value framework, the exchanged value has a certain monetary value for the business partner who is requesting the value. Value Delivery Modeling Language (VDML) is another value modeling facility aiming at value network modeling. In both VDML and e^3-value, roles perform activities which create and/or consume value through participating in a collaboration. An activity may internally be accomplished by a series of steps. Activities are defined in service collaboration. Roles can be assigned to many VDML participants. VDML assigns a role to participants through *role binding*. A collaboration comes with a service architecture. A simple or compound service contract is created to specify collaborations.

These value exchange and delivery models are mapped into petri net representations as follows: resources are mapped into places, events are mapped into transitions and roles are bound to places. Petri net is able to fit the multi-party business collaboration case, through proper partitions. The REA framework has shortcomings in describing some value types such as added-value. Added-value is also indirect in petri nets. Value changes with time and is influenced by many factors. In a timed petri net, the evaluation is a variable with respect to time. Value exchange is viewed as a kind of collaboration where economic agents such as buyers and sellers are involved in the collaboration. During exchange, economic resources like money, goods are being exchanged in the collaboration. To the participants, they are willing to join the collaboration because they benefit from the flow of resources.

4.2. Value Collaboration and Distribution

Organizations cooperate with each other in the long-term or in the short-term. The collaboration may be sponsored by one role or have equal status. Collaboration manners include but not be limited to sharing various assets including ideas, knowledge, work, computing and service assets, economic issues including costs, risks and benefits, and organizational relationships like trust. Collisions and collaboration ubiquitously exist among organizations.

Supply chain coordinates the collaboration among organizations. A supply chain can be viewed as activities one after another in service development. The supplier-customer relation in the supply chain can be taken as centralized or decentralized. The partnership relations are based on supply contracts. Contract favors the persistence of a relationship, and specifies non-cooperative behavior. Supply contracts can be evaluated by price, quantity, costs, time and quality, but in service scenario, supply contracts may be evaluated by capacity and continuity. Roles in supply chain include suppliers, producers, distributors, retailers and customers. In a multi-level supply chain, roles may have complex relationships. Roles are interconnected by factors such as material, financial, information and decision flow. The following assumptions are set for supply chain network:

(1) Each supply chain is composed of independent units with individual preferences;

(2) Each unit will attempt to optimize his individual preference.

Each unit's preferences in knowledge, information, performance, texts, security and other standards vary among different participants.

Supply chain can be described in a petri net as many processes. Compared with the service development process, roles are bound with resources (places in the petri net) in the supply chain rather than activities. However, supply process and development process share some common characters. Decision-makers in the supply chain optimize their preferences based on specific material, financial, information or decision flow factors. Their goals may be built on quality, time and/or cost. Decision-makers in service development choose options based on factors such as information, quality, service, time, attributes and functions. Functional property or non-functional property based decisions also exist in the supply chain. In supply chain, profits always come together with risks. In software development, functional and non-functional failures in design or degradation of components may happen. For the purpose of petri net modeling, adding extra decision paths to existing activities is one method to control risks or failures. Another method is to make an assumption that activities may fail, but the failure case leads to rework and finally succeeds. However, extra cost and/or time is paid.

Profit sharing contract in supply chain or supply network is used to stabilize inter-organization partnerships. To achieve the goal, profit sharing should ensure extra profit for each participant. Profits are not required to be in monetary or financial form, and information, skills, services, or potential benefits are also allowed. One profit sharing form for the supply chain is *options*, which includes real options and financial options. Profit sharing mechanisms designed for a service can also be accomplished through the priority of service usage or the extra time of service usage. Researchers in software economics have developed tradeoff analysis to guarantee functions and qualities, and the same time to satisfy each stakeholder's goals. Since tradeoffs exist for functional and qualitative properties, they are capable of balancing the increment of benefits and decrement of overall risks.

4.3. Several Quantitative Analysis on Value and Design Value

From a service provider's perspective, value means the satisfactory of related stakeholders. Each service provider joins the supply chain as an individual entity. Product value is the expected benefits from markets:

$$P = B - C \tag{2}$$

- P, the profit of the service
- B, the benefit of the service
- C, the cost of the service.

The profit problem is depicted as a mathematical programming of finding the maximization of the objective function

$$\max(B - C) \tag{3}$$

under various constraints. Investors of a project and researchers in value-based software engineering [14] also use the term Return-On-Investment (ROI) to evaluate the performance of investigations. ROI is calculated by formula

$$\text{ROI} = (P / C) \times 100\% = ((B - C) / C) \times 100\% \tag{4}$$

In both cases, value is considered as the potential of bringing benefits.

However, from the value exchange perspective, reversed resource flow is necessary. When considering customers' satisfactory, we have

$$B = \sum_{i=1}^{n} s_i \times PM_i, \tag{5}$$

- $s \in [0,1]$, the satisfactory of stakeholder i
- PM_i, the price when stakeholder i's expection is totally satisfied

Value evaluation may change from time to time. The calculation of value, cost and benefits may be misplaced during the development process: stakerholders pay at time t_j, while they evaluate the product at time i, which caused uncertainties. Stakeholders can also evaluate the design process, which means that stakeholders do not evaluate function or quality attributes by “have” or “have not”, but the difference between existing activities and ideal arrangements.

Service providers usually consider service capability, different customers and correlation matrix in their profit model. Different target markets may affect each other. One market target may disable other markets. In this case, we assume that a service is deployed with Cost c and to serve a specified kind of stakeholder i, then we have

$$B = \sum_{i=1}^{n} B_i = \sum_{i=1}^{n} r_i \cdot g_i \tag{6}$$

- g_i, value gained from stakeholder i
- r_i, the amount of demands for stakeholder i.

r_i is influenced by unit service price g_i. We suppose that quality attributes are $Q = \{q_i\}$. Then we have $r_i = r_i(q_1, \cdots, q_n; g_i)$.. The model does not distinguish between design value and execution value. Sometimes the market changes. Then we have $r_i = r_i(q_1, \cdots, q_n; g_i, t)$. Then the cost of design or redesign to meet specified service quality attributes/functionalities is calculated from $C = C(q_1, \cdots, q_n)$.

To calculate value in the process and lifecycle [14], the formula is given as

$$\text{Value} = \sum_{i=1}^{m}\left[\left((1-\prod_{j=1}^{n}(1-E_{ij})\right)\times R_i \times V\right] \quad (7)$$

- V, the total value of the project
- $E_{ij} \in [0,1]$, the effectiveness of a specific process activity on mitigating the risk of Q-attribute i if it is performed in phase j
- $R_i \in [0,1]$, the risk of Q-attribute i to the total value of the project.

And cost is calculated by

$$C = \sum_{j=1}^{n} C_{aj} + C_r \quad (8)$$

- C_{aj} the cost of a process activity at phase j
- C_r the potential cost of rework

4.4. Value in Development Coordination

Coordinations among SMEs are similar to the SaaS markets [15]. In this example, we assume that an ASP buys software infrastructures from an AIP. We suppose that the AIP provides service capacity μ in w per unit of capacity to ASP. The overall profit for AIP is $P_I = B_I - C_I = w\mu - C_I$ (9)

- C_I, the cost to provide service capacity μ

The cost of managing capacity and user access increases more when capacity μ increases. The case can be depicted as

$$C_I(\mu) = e_1\mu + e_2\mu^2 \quad (10)$$

where e_1 and e_2 are arbitrary real numbers, and usually $e_1 >> e_2$. ASP gains profits through their new service (value-added service). We denote D as the quantity of requirements, p as the price per quantity. Then the profit function for ASP is

$$P_S = B_S - C_S = pS - w\mu = pS - w\mu \quad (11)$$

- S, the quantity an ASP sells out

S is often influenced by ASP's price p. The joint profit function is

$$P_A = P_I + P_S = pS - C_I \quad (12)$$

The information is always sensitive in a business environment. And if ASP's pricing information and the corresponding market demand are available to the AIP, then API coordinates the SaaS by setting a unit capacity price. In actual case, application service provider in SMEs may know less market requirements or pricing strategy. To maximize the joint profits, we have $P_A' = 0$ or equivalently a mathematical programming problem

$$\max P_A \quad (13)$$

If demand function in economics is represented as $D(p)$, the constraint is $S(p) \le D(p)$. In this case, AIP tries to determine a reasonable capacity μ and price w, and ASP tries to determine a reasonable price p. $p > c$ and C_s is a quadratic form $c_1\mu + c_2\mu^2$. Demirkan, Cheng and Bandyopadhyay [15] proved that objective function P_A is strictly concave in both p and μ. The conclusion ensures the existence of a unique optimal solution. But the price w serving as an internal transfer price has no effect on the

overall profit. We need to seek other conditions to determine a reasonable w, which will be discussed in our future work.

5. Simulations and Analysis

This section illustrates two typical quantitative research questions for service development value calculation and value exchange in service lifecycle. The first simulation distinguishes between overall design value and output value. The second describes the effect of service coordination.

5.1. Process Simulations

Modelica is an object-oriented modeling language developed and promoted by the Modelica Association since 1996 for primarily modeling, simulation and programming of physical and technical systems and processes. Models in Modelica are defined in discrete (event-based), differential or algebraic equations (hybrid DAEs). Based on the PNlib tools aiming at xHPN simulation proposed in [6], we choose the mashup service development lifecycle proposed in [7] and model the process as in Figure 6. In Figure 6, resources are uniformly represented in (monetary-like) value. Resources are set to be used in the development process but do not disappear to meet the major property of assets during development process. Two lifecycle strategies are conducted. One sets the deadline for each phase and the other tries best to ensure that value increases through making full use of existing resources (we use "quality-first" to name the strategy). Two subtypes of deadline strategies are conducted. The first deadline development strategy is a rude version without allowing scheduling ahead after the former phase is completed. The second deadline development strategy enables the next phase to start in advance if the former phase is completed in advance.

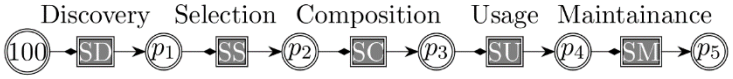


Figure 6. Development Process for Simulation

The time interval between two stochastic transitions follows the exponential distribution Exp(λ) = exp($-\lambda x$) with $\lambda = 1$, which in the simulation means that after every indefinite time the value of resources increases. We assume that the information directly provided by customers or requirements values 100. Fixed value vector $V = \{v_i\}$ means the usage of input value for each firing at phase i. Fixed value vector is set to $(4,6,8,8,30)$. Relative rate vector $R = \{r_i\}$ means the scale from input value to output value. Relative rate vector is set to $(5,10,15,20,25,30)$. When fires, a transition converts fixed value v_i in input places to value $v_i r_{i+1} / r_i$ in output places. In deadline strategies, deadline vector is set to be at day $(20,50,90,120,150)$. Table 3, shows maximum generated values after relevant phases and Table 4, shows residual values after relevant phases. In the simulation, a transition fires with a minimum value requirement on input places. It's normal that a small amount of value is unavailable in each phase. From the data in Table 3, and Table 4, we find that the quality-first method gains the most added value takes the longest time. Within the deadline strategies, value gains more if scheduling ahead is allowed.

Table 3. Transferred Value of Each Strategy

	SD	SS	SC	SU	SM	Total
Quality-first	192.0	297.0	397.7	490.0	576.0	591.7
Deadline 1	200.0	197.0	330.7	230.0	252.0	469.7
Deadline 2	194.0	288.0	384.0	320.0	360.0	516.0

Table 4. Residuals of Each Strategy

	SD	SS	SC	SU
Quality-first	2.0	1.0	2.7	10.0
Deadline 1	2.0	49.0	146.7	20.0
Deadline 2	8.0	0.0	128.0	20.0

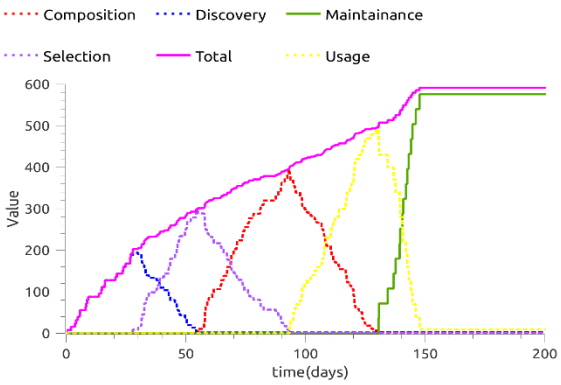


Figure 7. Quality-First Development Strategy

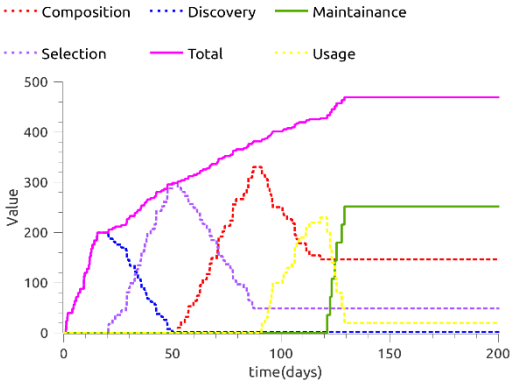


Figure 8. Deadline Strategy Without Scheduling Ahead

Figure 9. Deadline Strategy with Scheduling Ahead

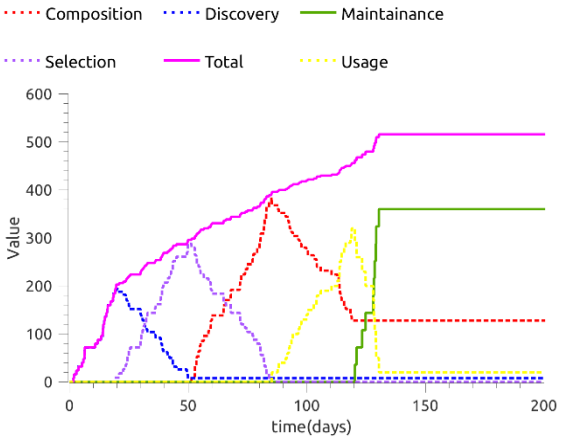
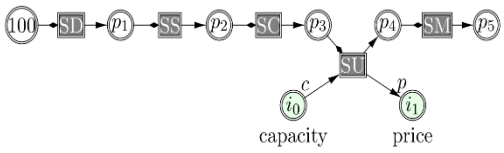


Figure 10. Buying Capacity from AIP



Value in Figure 7, Figure 8, and Figure 9, represents the accumulated value of assets until the peak of curves and then indicates their usage. Used value is not included in the total value to avoid redundancy. In this scenario, we recognized design value as total

value in Figure 7, Figure 8, and Figure 9, and recognized output value as value after maintenance phase in Figure 7, Figure 8, and Figure 9.

5.2. Outsourcing Usage/Deployment

In this case, an SME ASP buys computing resources that support the running of developed service, and maintain the service by developers itself. We assume that the collaboration has no impact on service quality and only affect service capability. Functionalities and requirements are not affected by this collaboration. In this case, p_3 adds its value by buying computing capacity from an AIP. The change is shown in Figure 10. At this time we ignore internal processes from p_3 to p_4 such as components' immigration to the AIP's infrastructure. We assume that services are deployed as a whole. We use the former profit calculation formula $P_S = pS - C_S$ in Section 4. And we set

$$C_S = a_0 + a_1\mu + a_2\mu^2 + w\mu \tag{14}$$

- a_0, fixed cost
- a_1, linear cost
- a_2, quadratic cost (extra cost with the increment of buying capacity)

and $S(p) = d_1 - d_2 p$, which means the quantity of service decreases linearly with the increment of price per unit p. After setting internal transfer to w, AIP's, ASP's and joint profit function are depicted as

$$P_I = B_I - C_I = w\mu - (e_1\mu + e_2\mu^2) \tag{15}$$

$$P_S = B_S - C_S = pS - a_1 - a_2(\mu) - a_3(\mu) \tag{16}$$

$$P_A = P_S + P_I = pS(p) - C_S + w\mu - (e_1\mu + e_2\mu^2) = (d_1 p - d_2 p^2) - k_1\mu - k_2\mu^2 - a_0 \tag{17}$$

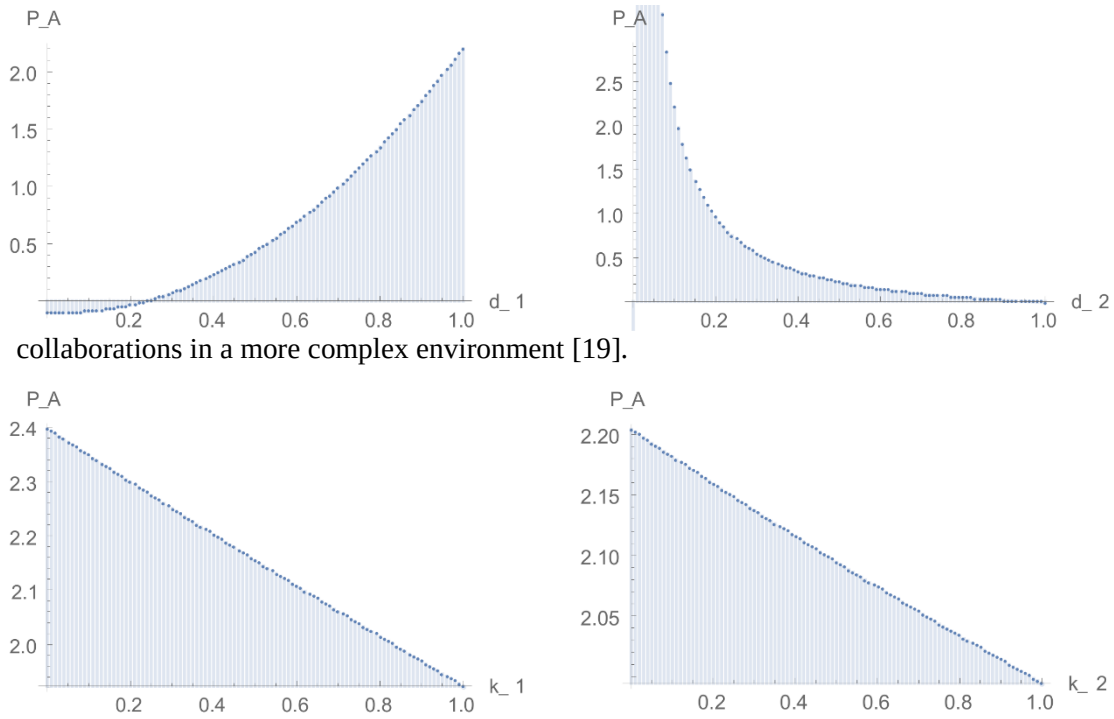
- $k_1 = a_1 + e_1$
- $k_2 = a_2 + e_2$,

which constraints to $S(p) \le D(p)$, $S(p) \le \mu$ and $w < p$. The problem of $\max(P_A)$ can be solved using numerical computing under most scientific computing platforms. We assume that μ^* and p^* are separately optimal capacity and price and set a group of variables on parameter d_1, d_2, k_1 and k_2. The base value of parameters is set to $d_1 = 1, d_2 = 0.1, k_1 = 0.4, k_2 = 0.01$, under which we figure out that $\max(P_A) = 2.2017$, $p^* = 5.20481, \mu^* = 0.479519$. Figure 11 shows the relation between $\max(P_A)$ and four parameters by separately fixing three of the four parameters d_1, d_2, k_1 and change only one parameter. After getting the optimal joint profit, ASP and AIP needs to find his or her optimal profit and negotiate profit distribution, during which parameter w is determined.

6. Related Work

Software process modeling and simulation have been applied for over 30 years, mostly of which aimed at effort, cost, reliability predictions and risk assessments [16]. Lavallee and Robillard [17] viewed software process modeling from a knowledge perspective, and added a new knowledge modeling layer to SPEM. Service supply chain techniques are still under rapid development. Several non-trivial differences between service supply and goods supply chain have been proposed. Santanna-Filho, Rabelo and Klen [1] researched

innovations for SMEs through collaborative networks of SOA-based software providers. The paper concentrated on collaborations in a virtual organization to achieve software innovation. Berre, Lew, Elvesater *et. al.,* [18] related value models, process models, user interface and interaction flow models, service architectures and service contract models under service innovation and service realization in VDML and ServiceML. Smart contracts are protocols that facilitate, verify or enforce the negotiation or performance of a contract. The execution and verification of contracts is promising to lead to deep collaborations in a more complex environment [19].



(a) $d_2 = 0.1, k_1 = 0.4, k_2 = 0.01$ (b) $d_1 = 1.0, k_1 = 0.4, k_2 = 0.01$

(c) $d_1 = 1.0, d_2 = 0.1, k_2 = 0.01$ (d) $d_1 = 1.0, d_2 = 0.1, k_1 = 0.40.$

Figure 11. Variability of Max(P_A) with d_1, d_2, k_1 and k_2

7. Conclusions and Future Work

We have separately used xHPN to describe and analyze technical developments processes and business value-related processes. Each aspect is in company with an illustration in Section 5. In order to generate an in-depth interaction between business and techniques, it is inevitable to reorganize entities such as roles, lifecycles, quality attributes, requirements, collaborations and contracts. Since more and more concepts and relations are built on or immigrated to model-driven frameworks, we plan to build a detailed metamodel for xHPN which enables both process evaluation through simulation and transformation between other models like OMG's SPEM and VDML in the future. Another direction is to combine workflow with knowledge database to match service development, and to provide suggestions for improving service quality and decision quality. In the future, software reuse and revolution will be considered.

Acknowledgments

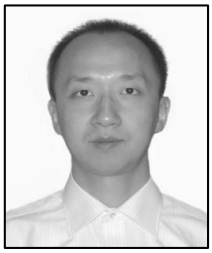
The authors acknowledge the support of the NSFC of China (No. 61363007 and No. 61462022) and Hainan NSF (No. 20156234), and Postdoctoral Science Foundation of Zhejiang Province of China under Grant No.BSH1502119. * refers to the corresponding author

References

[1] J. F. Santanna-Filho, R. J. Rabelo, and A. A. P. Klen, "An innovation model for collaborative networks of soa-based software providers", L. M. Camarinha-Matos and H. Afsarmanesh, Eds., ser. IFIP Advances in Information and Communication Technology, vol. 434, Springer, **(2014)**, pp. 169–181.
[2] M. Papazoglou, "Web services and SOA: Principles and technology", Essex, England New York: Pearson Education, , ISBN: 978-0273732167, **(2012)**.
[3] M. H. Cancian, R. J. Rabelo and C. G. Von Wangenheim, "Collaborative business processes for enhancing partnerships among software services providers", Enterprise IS, vol. 9, no. 5-6, **(2015)**, pp. 634–659.
[4] R. Maull, J. Geraldi and R. Johnston, "Service supply chains: A customer perspective", Journal of Supply Chain Management, vol. 48, no. 4, **(2012)**, pp. 72–86.
[5] Y. Duan, C. Ren, N. Zhou, X. Sun, M. Tang and H. Gao, "A problem-value-constraint framework for minimizing under design and over design in web service based system development", In ICSS 2015, Weihai, Shandong, China, IEEE, May 8-9 **(2015)**, pp. 117–124.
[6] S. Proß, B. Bachmann, S. Janowski and R. Hofestädt, "A new object-oriented petri net simulation environment based on modelica", in WSC'12, Berlin, December 9-12 **(2012)**, pp. 300:1–300:13.
[7] F. Daniel and M. Matera, "Mashups and end-user development", in Mashups, Springer Science & Business Media, **(2014)**, pp. 237–268.
[8] P. N. Robillard, "The role of knowledge in software development", Commun. ACM, vol. 42, no. 1, **(1999)**, pp. 87–92.
[9] G. Geerts and W. McCarthy, "Expert opinion [accounting]," IEEE Intell. Syst., vol. 14, no. 4, **(1999)** July, pp. 89–94.
[10] R. Schuster and T. Motal, "From e3-value to rea: Modeling multi-party e-business collaborations," in Commerce and Enterprise Computing, 2009. CEC '09, **(2009)**, pp. 202–208.
[11] V. Boucharas, S. Jansen and S. Brinkkemper, "Formalizing software ecosystem modeling," in Proceedings of the 1st international workshop on Open component ecosystems, ACM, **(2009)**, pp. 41–50.
[12] E. Handoyo, S. Jansen and S. Brinkkemper, "Software ecosystem modeling: The value chains," in MEDES '13, Luxembourg, Luxembourg: ACM, ISBN: 978-1-4503-2004-7, **(2013)**, pp. 17–24,
[13] J. Gordijn and H. Akkermans, "Designing and evaluating e-business models," IEEE Intell. Syst., vol. 16, no. 4, **(2001)** July, pp. 11–17.
[14] L. G. Huang, B. Boehm, H. Hu, J. Ge, L. Jian and C. Qian, "Applying the value/petri process to erp software development in china," in Icse, **(2006)**, pp. 502–511.
[15] H. Demirkan, H. K. Cheng and S. Bandyopadhyay, "Coordination strategies in an saas supply chain," J. of Management Information Systems, vol. 26, no. 4, **(2010)**, pp. 119–143.
[16] N. B. Ali, K. Petersen and C. Wohlin, "A systematic literature review on the industrial use of software process simulation," Journal of Systems and Software, vol. 97, **(2014)**, pp. 65–85.
[17] N. Kerzazi, M. Lavallée and P. N. Robillard, "A knowledge-based perspective for software process modeling," E-Informatica, vol. 7, no. 1, **(2013)**, pp. 25–33.
[18] A. J. Berre, Y. Lew, B. Elvesæter and H. de Man, "Service innovation and service realisation with VDML and serviceml," in EDOC Workshops, **(2013)** September 9-13, pp. 104–113.
[19] S. Fernandez, "Understanding contracting performance an empirical analysis," Administration & Society, vol. 41, no. 1, **(2009)**, pp. 67–100.

Authors

Chengxiang Ren, He received his bachelor's degree in mathematics and applied mathematics from Hainan University, Hainan, China in 2014. He is learning Computer Science in Hainan University. His research interests include service-oriented computing, model-driven engineering and data analysis.



Yucong Duan, He received his Ph.D. degree in software engineering from Institute of Software, Chinese Academy of Sciences, P. R. China in 2006. He is currently a full professor and vice director of Computer Science Department in Hainan University, P. R. China. His research interests include software engineering, service computing, cloud computing, and big data. He is a member of IEEE, ACM and CCF (China Computer Federation).

Zhangbing Zhou, He received his Ph.D. degree from DERI, Ireland in 2010. He is currently a full professor at the School of Information Engineering, China University of Geosciences, Beijing, China, and an adjunct associate professor at the Computer Science Department, TELECOM SudParis, France. His research interests include process-aware information system and sensor network middleware.

Guohua Fu, He received the Ph.D in Management. He is currently the vice president of Hainan Unversity. He is among the pioneers to propose the value Engineering analysis in education in China. He has published more than 100 research papers and teaching material in Economics and Management in both Chinese and English languages.

Zhen Guo, She is pursuing her doctor's degree in network security in Xidian University, Shanxi Province, China. She is currently a lecturer of College of Information Science and Technology in Hainan University. Her research interests include network communication, network security and information system.

Honghao Gao, He received his Ph.D. degree in computer application technology from the School of Computer Engineering and Science of Shanghai University, Shanghai, P. R. China in 2012. His research interests include Web Service and model checking.