VINCA ProCommunity: A Service "Virtual Organization" Postprint for Cloud Application Integration

Authors: Fang Jun, Zhao Zhuofeng, Wen Yan, Qi Kaiyuan

Date: 2017-03-10T00:00:00+00:00

Abstract

Application-layer cloud computing, which aims to achieve cross-domain resource sharing and application collaboration, must be capable of dynamically and flexibly constructing service-oriented "virtual organizations" targeted at specific business or application objectives for effective service management and collaboration. However, due to varying business requirements, foundational service resources used to construct industry integration applications often belong to different management domains. The diversity, variability, and autonomous management requirements of services determine that it is difficult to construct and maintain a stable, unified service space. To address this contradiction, it is necessary to transform physically dynamic, disordered, and unorganized service collections into bounded, ordered, and manageable/controllable logical service spaces. This paper focuses on elaborating the technical challenges involved, the research progress on related issues, and the partial work we have undertaken, and introduces the VINCA service community software implemented based on relevant research achievements.

Full Text

Preamble

Vol. 9 No. 1
Information Technology Letters
VINCA ProCommunity: A Service "Virtual Organization" for Cloud-Based Application Integration
Jun Fang, Zhuofeng Zhao, Yan Wen, Kaiyuan Qi

Abstract

Application-layer cloud computing, aimed at achieving cross-domain resource sharing and application collaboration, requires the dynamic and flexible construction of service "virtual organizations" oriented toward specific business or application objectives to enable effective service management and collaboration. However, due to varying business requirements, the foundational service resources used to construct industry integration applications often belong to different administrative domains. The diversity, variability, and autonomous management requirements of services make it difficult to construct and maintain a stable, unified service space. To resolve this contradiction, it is necessary to transform physically dynamic, disordered, and unorganized service collections into bounded, ordered, and manageable logical service spaces. This paper focuses on the technical challenges involved, reviews research progress on related issues, and introduces some of our own work, including the VINCA service community software implemented based on our research findings.

Keywords: service management, service community, business service, service availability, application integration

1. Introduction

Internet technologies, particularly emerging architectures, patterns, and technologies such as Service-Oriented Architecture (SOA), Software as a Service (SaaS), cloud computing, and Web 2.0, have enabled an increasing number of Internet resources and applications to be provided as services. Sharing and integrating services that encapsulate various network resources to construct integrated application software has become a new trend in application integration research.

A vast number of collaborative services on the Internet constitute the Internet of Service (IoS) [**?**]. In this context, services exhibit characteristics of morphological diversity, dynamic changeability, and business interactivity. From the perspective of supporting application integration, service resources also demonstrate domain relevance: due to different business requirements, services often belong to different administrative domains with varying organizational management requirements. Application-layer cloud computing, which aims to achieve cross-domain resource sharing and application collaboration, needs to dynamically and flexibly construct service "virtual organizations" oriented toward specific business or application objectives, thereby transforming these dynamic, disordered, and unorganized service collections into bounded, ordered, and manageable logical service spaces. This paper refers to these service "virtual organizations" as service communities. Focusing on service management within the service community mechanism, this paper discusses three key issues:

(1) **How to define, construct, and organize service communities oriented toward industry or domain applications?**
A service "virtual organization" differs from a simple service container. As

a virtual organization of services, how should an industry-oriented service community reflect industry or domain boundaries, and how should service management-related constraints be characterized? These are the questions that the definition of service communities must answer. Construction and organization are often complementary. A typical approach to organizing service communities involves first constructing "static" service communities with defined boundaries oriented toward specific industries or domains, then dynamically generating new service communities based on existing ones according to cross-domain integration requirements. Addressing community construction and organization means answering questions such as: under what conditions can communities be established, what operations are used to establish them, and what are the relationships between communities?

(2) **How to abstract and organize dispersed, autonomous, and diverse services to provide a relatively stable and unified resource view for the construction of integrated applications?**
Internet resource services exhibit diverse forms, such as Open API[1] and Web services. These services have different interface description methods, or lack explicit interface descriptions altogether, and their invocation methods also vary. Such heterogeneity poses significant obstacles to composing these different types of services. An effective service abstraction and organization method is needed to provide a stable and unified resource view within service community boundaries, thereby reducing the difficulty of constructing integrated applications.

(3) **How to flexibly and effectively monitor and ensure service reliability to guarantee the availability and effectiveness of constructed integrated applications?**
In the Internet computing environment, as the focus of application lifecycle management shifts from traditional development and construction phases to operation and maintenance phases [**?**], comprehensive control information about distributed service resources and integrated application behavior and status is required during the O&M phase. Meanwhile, due to the uncontrollable and uncertain nature of service resources in operational models, integrated applications cannot run stably and reliably forever. The contradiction between the potential unreliability of services (such as availability and quality of service) and the high availability requirements of integrated applications necessitates flexible and effective service monitoring and assurance mechanisms.

The service community proposed in this paper aims to provide a service management method that supports service space partitioning. It can dynamically construct and organize service communities oriented toward application domains, provide integrated service organization methods, and offer flexible and efficient

[1]Open Application Programming Interface, an open platform (open application programming interface)

service operation management and control mechanisms. This paper introduces the research status of the aforementioned issues from the perspectives of related work and our research progress, and describes the functional characteristics of the VINCA service community software developed based on our research findings.

2. Domain-Driven Service Community Modeling Technology

To support the implementation of cross-domain resource sharing and application collaboration in application-layer cloud computing, it is first necessary to reflect domain or problem domain characteristics from a service organization perspective, incorporating service resources that meet problem domain requirements into the service organization system to achieve interconnectivity of heterogeneous resources within a certain scope. This divide-and-conquer approach constitutes a logical partitioning of the global unified resource space. Community modeling technology primarily addresses issues including community structure models, community generation methods, and operational rules. Below, we introduce related research from two aspects: resource virtual organizations and service organization in service computing.

2.1 Resource Virtual Organizations

Virtual organizations can be studied from both structural and process perspectives. Structural research focuses on the constituent elements and properties of virtual organizations, while process research focuses on their behavior and operations. Current understanding of virtual organizations varies across different perspectives, but goal-driven nature, dynamism, and member autonomy are consistently recognized as fundamental characteristics.

Due to different research groups understanding and studying virtual organizations from various angles, there is currently no unified consensus on virtual organization models. Katzy et al. [**?**] proposed a modeling method positioning framework to study different virtual organization modeling approaches. Using this framework, various modeling methods can be classified and analyzed. For example, the St. Gallen Management Model [**?**] understands virtual organizations from an organizational management perspective; VSD (Value System Designer) is a management-oriented business process redesign method; GERAM (Generalized Enterprise Reference Architecture and Methodology) is positioned for enterprise integration and system design implementation; and Rosettanet[2] is a standard for inter-system collaboration. These models still primarily target individual enterprises.

Camarinha-Matos [**?**] proposed characterizing a virtual organization from four aspects: Relationship Model, Roles Model, Process Model, and Deontic/Values

[2]A non-profit organization jointly founded by forty IT companies in June 1998, with the purpose of establishing, applying, and promoting open e-commerce standards.

Model. The Relationship Model characterizes relationships between components in a virtual organization, such as control, dependency, ownership, and peer relationships. The Roles Model describes all roles in the virtual organization and their positioning. The Process Model describes the dynamic processing of business processes. The Values Model defines constraints on all members of the virtual organization, such as interoperability constraints and behavior constraints.

Additionally, Saabeel [**?**] also proposed a virtual organization model that characterizes a virtual organization from both structural and process aspects. The structural aspect defines the elements of a virtual organization and relationships between them, while the process aspect defines the functions and roles required for reconfiguring and restoring the virtual organization. This model primarily considers virtual organization construction from an organizational management perspective. Song (W.) [**?**] proposed a virtual organization conceptual model for the grid environment to dynamically integrate resources according to user requirements.

2.2 Service Organization in Service Computing

Traditional service organization in service computing primarily aims at service discovery, focusing on establishing service metadata repositories, but lacks consideration for modeling application domains and their boundaries. For example, UDDI[3] is an early work involving service management boundaries, which attempted to establish a Web service management center through unified service description, publication, and discovery protocols. It was entirely resource-centric and organization-independent. Works such as StrikeIron, XMethods, programmableWeb, and WebServiceX established Internet-based Web service information centers through Web service collection on the Internet and provider registration. These are essentially service metadata repositories that lack consideration for modeling application domains and their boundaries.

Multiple IT vendors have developed service management and governance tools for enterprise-level service asset management. IBM' s WSRR, HP' s Systinet, and SAP' s ESRR are examples of such products. They store and manage various assets such as services, documents, and processes, support asset cataloging and classification management, and publish these assets to registries to support discovery and retrieval.

To overcome the limitations of centralized service repositories and ensure service discovery efficiency while reducing maintenance difficulty, researchers have begun exploring "distributed" service repository models. Different works adopt different topological structures for distributing nodes, such as hierarchical structures, hybrid peer-to-peer (P2P) networks, or completely unstructured P2P networks. A typical work, MSWDI (METEOR-S) [**?**], is a semantic-based, fed-

[3]Universal Description, Discovery, and Integration, a cross-platform XML-based specification for describing, discovering, and integrating web services.

erated framework for service publication and discovery. It organizes the topology of various registry operation nodes based on service repository ontologies. Services are published to specific operation nodes based on their semantics, while gateway nodes and auxiliary nodes are used to maintain service repository ontology relationships, routing service publication and discovery operations to matching operation nodes. This service repository organization model still adopts the traditional service discovery perspective, lacks characterization of domain or problem domain features, and cannot directly support the resolution of application integration problems.

To meet the customization and evolutionary derivation requirements of service management boundaries, we adopt a domain-oriented and meta-modeling approach, proposing the service community concept and corresponding service community model [**?**]. A service community is a service management unit customized by integrated application managers according to their problem domain' s service management requirements, and can also be viewed as a logical container for services established by a third party. The service community model supports the customization of management boundaries for specific industries or businesses, primarily comprising elements such as service meta-modeling support, business specification definition, and service management and control policy definition. It allows users to flexibly customize service management boundaries —service communities—that include specific service models, business specifications, and management and control policies. On one hand, service communities determine logical boundaries for services and achieve logical partitioning of service resources through autonomous customization of service models, business specifications, and management and control policies. On the other hand, with cross-domain collaboration and integration as the goal, service communities achieve evolution of service management boundaries through derivation operations. Based on the evolution forms of service management boundaries, we define two basic derivation patterns: (1) derivation on a single service community, i.e., extracting partial content from an existing service community to form a new community; and (2) derivation through fusion of two service communities, i.e., forming a new community by merging two existing service communities.

3. Service Integration Technology Based on Virtualization

Network resources exhibit diverse forms, requiring research into business service models and modeling methods with large granularity, business semantics, and the ability to shield the heterogeneity of network resources. The concept of virtualization was first proposed in the 1960s and continues to attract significant research attention today. Different from virtualization of cloud computing infrastructure, this paper focuses on service virtualization oriented toward business application construction. Depending on the virtualization level, service virtualization mainly includes service encapsulation technology, service abstraction technology, and corresponding virtualization mapping mechanisms.

3.1 Resource Service Encapsulation

Resource service encapsulation technology aims to encapsulate individual or multiple resources such as data, software, web pages, and devices as network-accessible services. Considerable work has been done in this area. At the software tool support level, Apache' s open-source projects Axis and CXF provide relevant interfaces for Java service encapsulation. Regarding specific service encapsulation technology, reference [**?**] discusses service-oriented methodologies for software applications. Publishing databases as services is one of the main approaches to solving data integration problems, as it shields the heterogeneity of data sources related to specific vendor database technologies. Database operations are published through services, eliminating the need for clients to install drivers related to specific database systems or concern themselves with any updates or changes to database systems and drivers, while also freeing users from issues such as JDBC[4] connection pools. References [**?**] represent work in this area.

Current service encapsulation of Web information resources primarily relies on wrapper technology, which is a class of programs that automatically extract information from Web sites and convert it into structured data. For example, the Pollock system [**?**] divides Web information resource service encapsulation into two steps: construction-time and runtime. During construction-time, existing wrapper generation technologies are used to encapsulate Web information sources while generating WSDL[5] documents for Web services. During runtime, wrappers are virtualized as Web services, converting SOAP[6]-based interactions into wrapper-specific interactions.

3.2 Business-Level Service Modeling

The core of clustering abstraction technology is the abstract service model problem. Based on different perspectives such as business and information technology, existing service models can be divided into three categories according to different concerns [**?**]:

- **Service models reflecting only the business level**: These models purely consider business-level content without considering their association with computing resources. They mainly include activity models in business and enterprise modeling, such as the business activity model in MIT Process Ontology [**?**], the Generic Activity Module (GAM) defined by ISO TC184, the activity unit defined by CIM-OSA, the generic activity module defined by PERA, and the functional activity model in the IDEF0 method.
- **Service models resulting from encapsulating computing re-**

[4]Java Database Connectivity, a Java API for connecting and executing queries on databases.
[5]Web Services Description Language, an XML-based language for describing Web services.
[6]Simple Object Access Protocol, a protocol for exchanging structured information in Web services.

sources: These models are primarily designed from the perspective of encapsulating computing resources. Representative works in this category include Web Service and RESTful[7] Service. Currently, with the growing popularity of "Software as a Service" technology, service models and description languages for characterizing Software as a Service, such as SaaS-DL [**?**], have also emerged. These define what aspects of services should be described and how to describe them. Service registries manage various service information according to the definitions of metadata models. Represented by UDDI, most current service registries attempt to provide a generic service data meta-model. To support sharing of service resources, service metadata description models should be domain-oriented during establishment, and the construction methods should be flexible and extensible. To address this issue, reference [**?**] proposes an adaptive service metadata description model that divides service metadata models into two categories: generic metadata models and extended metadata models. Generic metadata models describe public description information required by all services, while extended metadata models describe supplementary or further refined description information needed by certain services for specific domains and requirements.

- **Service models for associating business requirements with computing resources**: The purpose of proposing this third category of service models is to conveniently associate business requirements with computing resources. This association is generally implemented in two ways. The first adopts a manual approach to associate business activity/service models with computing resources, mainly including activity models in business processes and industry-specific informatization specifications, such as the activity model in WfMC[8], the activity model in BPMN[9] proposed by OMG[10], the activity model in BPEL[11], the Unified Service Action Model of the HL7 organization in the healthcare domain, and activity models in e-commerce application standards such as ebXML and RosettaNet. The second adopts an automatic approach to associate business activity models with computing resources, mainly including semantic service models, activity models described using ontologies, and activity models in action theory, such as OWL-S, WSMO, WSDL-S, SWSL, TOVE, TOL, as well as action models in situation calculus, action description languages, and dynamic logic.

[7]REST: Representational State Transfer, a software architectural style for distributed hypermedia systems.

[8]Workflow Management Coalition, an organization that defines workflow standards.

[9]Business Process Modeling Notation, a standard for business process modeling proposed by the OMG organization.

[10]Object Management Group, an international standards organization.

[11]Business Process Execution Language, an XML-based language for describing business processes.

3.3 Virtualization Mapping Mechanism

Virtualization mapping mechanism is a concrete implementation of aggregation ideas. The goal of virtualization is to associate constructed business services with IT-layer Web service resources to ensure the executability of business services.

Service clustering abstracts services with identical or similar functionality into service agents, making it easier to support dynamic binding, discovery, and replacement of services, and achieving location transparency of service access. Typical technologies include Service Domain [**?**] and Service Container [**?**]. The goal of service transformation is to bring new functionality to services by transforming basic information in service descriptions. Unlike traditional hard-coding methods, service transformation approaches can be more flexible and adaptive. A typical work in this area is reference [**?**]. Composition virtualization refers to virtualizing a group of service resources with control logic coordination relationships, providing a single usage interface for this group of service resources while shielding users from the specific coordination relationships between service resources. The Business Process Execution Language (BPEL) is representative of this type of virtualization technology.

To solve heterogeneity problems, it is necessary to uniformly abstract and describe these network resources, shielding the differences among different network resources and effectively reducing the difficulty of composition and construction. To this end, we first abstract a business-level service [**?**] that can embody domain knowledge, directly correspond to reusable business capabilities, and be compatible with various physical implementations of Internet resources, serving as the basic element for composition and coordination. A business service is defined as an abstract representation of the concrete implementation of a business activity, data, interface, or process fragment. Regarding resource metadata description, we propose a service model and its extension mechanism suitable for domain metadata description [**?**]. Additionally, we propose a corresponding service virtualization mechanism [**?**] that uses certain methods or technical means to encapsulate, process, abstract, and transform various network resources, thereby establishing a mapping between business services and network resources to help business services truly 落实到 specific software-layer resources at runtime.

4. Service Availability Assurance Technology

Service resources have the characteristic of "use without ownership," exhibiting uncontrollability and uncertainty. This makes integrated applications based on service composition and coordination unable to run stably and reliably forever. Su Myeon Kim et al. continuously monitored over 1,000 Web services published in UBR (Universal Business Registry) for two years and found that more than 16% of Web services failed every week. As the scale of services continues to expand and the scale of integrated applications increases, how to address the impact of service uncontrollability and uncertainty on integrated application

availability—especially during the runtime phase—has become an urgent problem for service availability assurance technology to solve.

4.1 Service Availability Metrics and Measurement

We believe that service availability is a multi-factor concept, particularly considering the dynamic characteristics of services at runtime. For example, availability can be measured from a temporal perspective as instantaneous availability, steady-state availability, and inherent availability; or from a performance perspective as access efficiency availability and performance load availability. Simultaneously, to meet application runtime management requirements, it is urgent to jointly determine an appropriate service availability indicator system in conjunction with integrated application availability assurance and adaptive mechanism requirements.

Regarding service availability characterization and measurement, most current work defines service availability as "the probability that a service can be normally accessed or used within a period of time under specific conditions." According to this definition, service availability is often expressed as the proportion of time during which a service can be normally used within a specific period. In addition to measurement methods based on normal usage probability, another category of research primarily conducts static measurement of service availability based on historical information or user feedback. However, these static measurement approaches cannot meet the dynamic measurement requirements of service availability needed for Internet application runtime phases. In reference [**?**], the authors propose an availability checking model for Web service availability measurement in mobile computing environments based on end-to-end Quality of Service (QoS) parameters, and implement measurement of a set of availability indicators from aspects such as service components, servers, networks, and devices by establishing availability checkpoints, ultimately providing decision support for availability assurance measures.

Furthermore, a closely related research area to service availability is service quality management, where availability is often an important indicator of quality management. Existing service quality assessment techniques are mainly divided into two categories: one assesses the service quality provided by components based on user feedback ratings, while the other uses monitoring methods to record the service quality provided by services in actual execution, assessing the provided service quality based on this foundation. The main deficiency of the first assessment method is that ratings are subjective, and users may provide false ratings for certain reasons, with no good mechanism to incentivize users to provide rating feedback. The main problem with the second assessment method is the impact of service quality information monitoring on overall system performance.

4.2 Proactive Service Availability Assurance Mechanisms

Due to the autonomous nature of service resources, it is generally difficult to directly control service resources to affect their availability. Traditional fault tolerance, backup, and replication mechanisms considered purely from the service provider's perspective are difficult to directly apply to resource-autonomous and open environments. Therefore, service availability assurance mechanisms need to shift from direct control methods purely oriented toward service providers to "proactive" assurance methods that combine direct control means and indirect guidance means oriented toward users, managers, and providers, focusing on early warning, service replacement, dynamic adjustment, and reputation incentives.

Service monitoring is the foundation for ensuring service availability and Internet application availability. Service and application status and related availability information obtained through monitoring can be used to measure service availability and assist in proposing availability assurance strategies. Currently, there are three main types of service monitoring methods [**?**]: monitoring through message listening between service users and providers; monitoring through implanting monitoring code at the service provider side; and monitoring through trusted third-party intermediary methods.

Reference [**?**] considers improving service availability from the perspective of monitoring service interaction processes, but current work mainly proceeds from the angle of improving service interaction protocols, such as extending SOAP message protocols. This requires service users and providers to use communication protocols that comply with specific constraints when interacting, demanding additional implementation work from service providers and thus violating the autonomous nature of services to some extent. Reference [**?**] proposes a trust determination method for monitoring intermediaries from the perspective of ensuring the trustworthiness of third-party monitoring intermediaries.

Service replacement is a primary availability assurance mechanism for Internet applications, with its core issue being service compatibility determination. In reference [**?**], the authors simulate Web service behavior using Labeled Transition Systems (LTS) and define concepts of compatibility between two Web services under different conditions. Based on different compatibility definitions, the authors further propose concepts of Web service replaceability that satisfy different conditions. However, these compatibility and replaceability concepts are based on the interaction composition of two Web services and do not consider applicability to multiple services. In reference [**?**], target services are given in "contract" form. After existing component services fail, new services need to maintain consistency with the contracts of existing services to be replaceable. Web service communication in this work is synchronous. Reference [**?**] provides conditions that must be satisfied during service replacement based on service compatibility descriptions, but such replacement is based on synchronous message interaction for Web service composition, and no corresponding algorithms

are proposed or verified. A common assumption in the above work is that the availability of replacement services is guaranteed, and none consider service replacement in conjunction with service availability conditions. Additionally, reputation incentives are a system behavior guidance mechanism that can serve as an effective approach to indirectly adjusting Internet application availability from the perspective of group behavior [**?**]. How to conduct reputation evaluation of services based on historical service availability measurement, thereby guiding service providers to further provide services with higher availability from their own perspective, has become a key focus.

Regarding service availability assurance, combined with the characteristics of business service abstraction, we propose a method for improving service availability using business service abstraction [**?**]. This method first uses business service abstraction to aggregate specific services with similar functionality, and on this basis proposes two technologies—runtime request splitting and dynamic switching—to implement service availability improvement, along with corresponding business service execution algorithms. Compared with traditional service replica-based approaches, this method can avoid the unavailability problem of individual services. The method also proposes a runtime service selection algorithm based on real-time availability updates and service coverage of user requests to improve business service execution efficiency and reduce implementation costs.

Additionally, regarding service quality monitoring, to address the diversity of monitoring objectives and requirements, we propose a flexibly customizable service monitoring model [**?**]. It adapts to diverse monitoring needs through configurable data models and process models. The data model defines monitoring information (service quality information and exceptions), while the process model defines trigger conditions and calculation methods for monitoring based on ECA rules, and selects corresponding monitoring mechanisms (timed, message interception, etc.) and feedback mechanisms (service query, message notification).

5. VINCA ProCommunity Software

To support the service community model and corresponding modeling and management operations, we have implemented the VINCA ProCommunity service community software that supports distributed deployment and hierarchical interconnection. Its main functions include:

(1) **Service Community Creation**
Function Overview: Provides two methods—direct community creation and derivation based on existing communities—allowing the creation of static service communities oriented toward stable business domains, as well as dynamic generation of service communities according to application collaboration requirements.

- **Direct Creation**: Community administrators (such as application owners or autonomous domain administrators) can create a community by

simply clarifying the business specifications of the community, including service description models and service classification systems.

- **Derivation from Existing Service Communities**: New communities can be formed through projection operations on a single community under certain constraints, or through fusion operations among multiple communities.

Technical Features:
- Business specifications, such as introduced business data specifications and business service specifications, on one hand determine the logical boundaries of service communities and achieve logical partitioning of service resource spaces, and on the other hand focus on solving standardization and heterogeneity issues in application integration, helping virtual organizations solve Internet application integration problems.
- For potential inconsistencies between local resources and specifications, conversion and adaptation means are provided within a certain scope to ensure compliance with specification requirements without modifying local resources.

(2) **Integrated Service Catalog**
Function Overview: Capable of accessing and organizing diverse Internet service resources. To meet users' personalized service usage requirements, it provides a customizable service catalog that can invoke Internet services with various technical forms.

- Built-in basic service description model supporting general descriptions of mainstream services:
  - Basic metadata, including service name, service provider, service parameter semantics, etc. (for retrieval)
  - Service access metadata, including service access address, service interface parameters, etc. (for service invocation)
  - Service quality information, including response time, availability, etc. (for service management)
  - Business information, including business classification, subject terms, affiliated business services, business constraints, etc. (for service organization)
- Community administrators can customize new service types according to business needs and service requirements, extending description elements in basic information, service access information, service quality information, and business information to meet specific requirements in service description, service access, and service quality monitoring.
- Supports characterization of complex interaction relationships between services, such as competitive and collaborative relationships.
- Supports user personalized service space construction through multi-catalog browsing, personalized catalog customization, and catalog fusion.
- Built-in invocation methods for mainstream services, supporting invocation of SOAP Web Services and RESTful Web Services, and providing

an adapter framework to support autonomous access to service invocation methods. Asynchronous invocation mode is adopted for services with long access times.

Technical Features:
- Service resource types are diverse. Mainstream services such as SOAP Web services, RESTful Web services, and other forms of Web APIs often have different self-description forms, and different business domains generally have special description requirements for interface specifications, service quality, and other aspects. To address this, a unified service description mechanism is proposed to support the extensibility requirements of mainstream services during access. Community service catalogs can be personalized through add, delete, modify, and fusion operations.
- Service access involves not only service description but also service usability within the community. To address service diversity, a flexibly extensible service invocation and adaptation framework is proposed. To address service access reliability issues, asynchronous invocation mode is adopted.

(3) **Proactive Service Management for Availability**
Function Overview: To effectively ensure the reliability of services already accessed in the service community, in addition to reviewing their validity during access, we provide standardized and unified means for defining service management and control policies, supporting the definition of service management and control content, monitoring methods, measurement and evaluation methods, and control modes to meet various needs such as service monitoring, service evaluation, and service quality assurance under specific management domains.

- Supports both message interception and proactive acquisition methods to adapt to different monitoring indicator characteristics. For example, for service invocation counts, message interception is more accurate than proactive acquisition; for service response time, proactive acquisition is more suitable.
- The service monitoring process does not affect the performance of business applications, especially when a large number of services need to be monitored.
- Based on monitoring results, corresponding management measures can be taken, such as statically controlling service status or dynamically performing service replacement.

Technical Features:
- Establishes a service availability measurement indicator system. Based on an extensible service monitoring framework, it achieves efficient, reliable, and flexible monitoring and evaluation, providing decision support for service availability assurance.

As a type of service management middleware, service community software has a wide range of applications. It can effectively manage service resources within

autonomous domains (such as within enterprises or industry alliances) and is particularly adept at cross-domain service sharing and management among multiple applications. It can either independently complete service management functions within autonomous domains or serve as foundational middleware to cooperate with other middleware or application software, such as ESB and application assembly software, to accomplish more advanced functions.

6. Conclusion

The VINCA service community proposed in this paper follows the principle of "divide and conquer," constructing business-oriented virtual connectivity with business as the unit. It can determine service management boundaries from perspectives such as service target scope, management specifications, and control strategies according to the needs of different industries or business domains. It provides integrated service resource abstraction technology that shields issues such as service diversity and heterogeneity, reducing the difficulty of application integration. From a runtime assurance perspective, it proposes fundamental methods for proactive service availability assurance. In summary, the VINCA service community is an effective method for ensuring the manageability of services for application integration under cloud computing models.

References

[?] Schroth, C., Janner, T.: Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services. IT Professional 3.36–41.2007.
[?] Katzy, B.R. and G. Sung, State-of-the-Art of Virtual Organization Modeling CeTIM at University Bw Munich, Germany., 2003.
[?] Ullrich, H. and W. Krieg, St. Gallen Management Model. Bern, Haupt, 1974.
[?] Camarinha-Matos, L.M. and A. Abreu, Towards a foundation for virtual organizations. Proceedings of Business Excellence 2003 - 1st Int. Conference on Performance measures, Benchmarking, and Best Practices in New Economy, Portugal, 10-13 June., 2003.
[?] Saabeel, W., et al., A Model of Virtual Organisation: A Structure and Process Perspective. Electronic Journal of Organizational Virtualness, 2002. 4(1): p. 1-17.
[?] Song, W. and X. Li, A Conceptual Modeling Approach to Virtual Organizations in the Grid. Proceedings of the 4th International Conference on Grid and Cooperative Computing (GCC2005), 2005: p. 382-393.
[?] Verma K et al. Meteor-S WSDI: A scalable P2P infrastructure of registries for semantic publication and discovery of Web services. Journal of Information Technology and Management, 6(1):17-39
[?] 王卓昊, 赵卓峰, 房俊, 王希诚. 一种 SaaS 模式下的服务社区模型及其在全国科技信息服务网中的应用. 计算机学报, 已录用.
[?] 王威. 基于 SOA 的科技项目管理遗留系统重构关键技术研究. 中南大学硕士学位论文.
[?] K. Karasavvas and et al. Introduction to OGSA-DAI Services. In Lecture

Notes in Computer Science, volume 3458, pages 1-12, May 2005.
[?] Michael Koch, Markus Hillenbrand, Paul Müller. A Generic Database Web Service for the Venice Service Grid. 2009 International Conference on Advanced Information Networking and Applications.
[?] Erdogan Dogdu, Yanchao Wang and Swetha Desetty. A Generic Database Web Service. In Proc. of the 2006 Int. Conf. On Semantic Web and Web Services.
[?] Y Lu, Y Hong, et al. Pollock : Automatic generation of virtual Web services from Web sites. ACM Symposium on Applied Computing(SAC). New Mexico, USA, 2005.
[?] Z. Baida, J. Gordijn, et al. A Shared Service Terminology for Online Service Provisioning. In Proceedings of the 6th International Conference on Electronic Commerce, Delft, Netherlands, 2004,
[?] T. W. Malone, K. Crowston, et al. Organizing Business Knowledge: The MIT Process Handbook. Cambridge, MA: MIT Press. 2003.
[?] K. Zhang, X. Zhang, W. Sun, L. Wei, H. Liang, Y. Huang, L. Zeng, and X. Liu. A Policy-driven approach for Software-as-Services Customization. E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, July 2007,
[?] Woodrow C, Singh V. 2009. Websphere registry and repository (wsrr) set-up, administration & standards. http://www.labor.state.ny.us/cioshares/pdf/WSRR_SetUp_Admin_&Standards.pdf.
[?] Y.S.Tan, V. Vellanki, J. Xing, B. Topol, and G. Dudley. Service Domains. IBM SYSTEMS JOURNAL, 2004, 43(4): 734 -755.
[?] B. Benatallah, et al. Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web Services. In Proceedings of International Conference on Data Engineering (ICDE), San Jose, CA, 2002, 297-308.
[?] A. Dogac, G. Laleci, S. Kirbas, et al. Artemis: Deploying Semantically Enriched Web Services in the Healthcare Domain. Information Systems, 2006, 31(4-5): 321-339.
[?] Yanbo Han, Jing Wang, Peng Zhang. Business-oriented service modeling: A case study. Simulation Modeling Practice and Theory. 17 (2009), 1413-1429.
[?] Zhuofeng Zhao, Jun Fang, Jing Cheng. A Business Services based Approach for Deploying SOA in Scientific Information Integration. Journal of Harbin Institute of Technology, 2008, Vol. 15, Sup. 1, pp:131-135.
[?] 房俊, 虎嵩林, 韩燕波, 刘晨. 一种支持业务端编程的服务虚拟化机制 VINCA-VM. 计算机学报, 2005, 28(4), pp: 549~557.
[?] K. Tan, S. Mustapha. Measuring Availability of Mobile Web Services.
[?] R. Jurca, W. Binder, B. Faltings. Reliable QoS Monitoring Based on Client Feedback. WWW 2007, Banff, Canada, pp.1003-1011, May, 2007.
[?] N. Thio and S. Karunasekera. Automatic Measurement of a QoS Metric for Web Service the 2005 Australian Software Engineering Conference, Recommendation. Proceedings of Washington, DC, USA, pp. 202 -211, 2005.
[?] H. Rajan and M. Hosamani. Tisa: Toward Trustworthy Services in a Service-Oriented Architecture. IEEE Transactions on Services Computing, VOL. 1, NO. 4, pp: 201-212, 2008.

[?] M. Mecella, B. Pemici, P. Craca. Compatibility of e-services in a cooperative multi-platform environment. TES2001, Springer 2001, LNCS 2193, pp.44-57.
[?] L. Bordeaux, G. Salun, D. Berardi, and M. Mecella. When are Two Web Services Compatible?TES2004, Springer, 2005, pp.15-28.
[?] S.D. Kamvar, M.T. Schlosser, H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P Networks. Proceedings of the 12th international conference on World Wide Web, pp.640-651, May 2003.
[?] 温彦, 房俊, 刘晨. 一种利用业务服务抽象提升服务可用性的方法. 计算机学报, 已录用.
[?] Kaiyuan Qi, Yanbo Han, Zhuofeng Zhao, Jun Fang. An Adaptive Service Monitor Providing Runtime Extensibility. The 5th IEEE International Symposium on Service Oriented System Engineering. 2010, pp.165-172.

Author Biographies:

Jun Fang: Assistant researcher and Ph.D. at the Software Integration and Service Computing Research Sub-center, Institute of Computing Technology, Chinese Academy of Sciences. fangjun@software.ict.ac.cn

Zhuofeng Zhao: Senior engineer and Ph.D. at the Software Integration and Service Computing Research Sub-center, Institute of Computing Technology, Chinese Academy of Sciences.

Yan Wen: Ph.D. candidate at the Software Integration and Service Computing Research Sub-center, Institute of Computing Technology, Chinese Academy of Sciences.

Kaiyuan Qi: Ph.D. candidate at the Software Integration and Service Computing Research Sub-center, Institute of Computing Technology, Chinese Academy of Sciences.

Note: Figure translations are in progress. See original paper for figures.