Multi-Layer SOA Implementation Pattern with Service and Data Proxies for Distributed Data-Intensive Application System

Takdir
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
takdir.rex@students.itb.ac.id

Achmad Imam Kistijantoro
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
imam@stei.itb.ac.id

***Abstract*— Web service is becoming a widely solution used to realize Service Oriented Architecture (SOA). It enables application's business logics interoperable with each other by extending XML flexibility. Current early standardized web service technologies show significant progress in functional aspects interaction, but they are lack of data aspect considerations, such as transferring and integrating large amount of data that are distributed throughout different locations. In this research, we propose a SOA implementation pattern improving coordination of both functional and data aspects to be used across distributed environment by combining several advances in SOA and distributed systems. We define synchronization, replication, and routing mechanism to support distributed systems. This paper introduces the design as preliminary result of our research. It is designed by analyzing previous approach to avoid semantic contradictions of SOA principles and its existing standards and technologies. Next, we need to realize and evaluate them in order to be used in the real application.**

Keywords*— *data-intensive; SOA; web service; data integration; distributed data; distributed process; distributed system

I. INTRODUCTION

Service Oriented Architecture (SOA) has become a recent adopted approach in application system design. It introduces reusability of functionality to reduce efforts in repeating constructions of similar functional aspects. Web services technology, which extends XML portability, offers solution in making application's business logics interoperable with each other. End-product applications, called composite applications, are composed of web services, which execute the corresponding logics.

However, current web service technologies and standards mainly focus on functional aspects of distributed components interaction [1]. Data is treated only as input (parameter) and output (return value), thus data sources cannot be accessed directly by clients. As a result, transforming data-intensive applications, which are characterized by (i) a set of functional operations processing large amounts of data and (ii) the delivery and transformation of huge data sets between those functional activities [2], into a SOA-based environment by simply using existing web services standards without well-designed implementation pattern results the severe process of parsing a large amount of data.

Various approaches have been proposed as solutions to data-intensive applications. Some of them (e.g. Web Data- and Artifact- centric Service (W-DAS) [1] and Reference Resolution System (RRS) [2]) are simply separating data access and functional aspects, whereupon composite applications use different communication ports to access the web services and the data sources directly. In this scenario, applications are tightly coupled with the data sources and the web services cannot guarantee whether the data access is available if it is being reused by others. Reusability is sacrificed using this scenario. Other approaches (e.g. Stream-Based Web Service Invocation [3] and a new bulk data transfer with service encapsulation [4]) try to modify web services messaging protocol (SOAP) to handle direct reference to data sources, but none of them is acceptable as new standard due to fundamental changes which affect other existing web services standards.

On the other hand, locating data and its related functional logic (web service) centralized in single place will reduce performance and scalability because of network traffic bottleneck. It also decreases the data availability which is a critical point in the data-centric business. Decentralizing them closer to their concerned users gives some advantages. First, it will improve response time and decrease network traffic. In addition, partitioning data and its functional aspects across multiple systems can improve query performance. Another advantage is maximizing availability by removing a single point of failure.

Choosing the decentralization approach does not mean we totally evade all related issues. Well-considered synchronization and replication strategies become success criteria in decentralizing resources. Effective routing that point request to the proper location is another important consideration.

In distributed systems field, Data Grid offered a distributed infrastructure that integrates distributed and independently managed data resources. Associated with it, recently, a lot of inventions were offered such as WebSphere Information Integrator (WSII), Mobius, Open Grid Services Architecture -

Data Access and Integration (OGSA-DAI), and Data Distribution Service (DDS). These all provide mechanism to integrate data from geographically distributed environment for applications working across administrative domains.

In contrast, there is only few research focuses on synchronization of data aspect and functional aspect to deliver SOA flexibility and reusability advantages for implementing data-intensive application system. In this research, we aim at creating SOA implementation pattern considering both data and functional aspect to be used across distributed environment. It requires well-designed integration pattern which is not only considers the service level (web services), but also the process level (BPEL) design to ensure all elements of SOA support these aspects pervasively. We propose a comprehensive SOA implementation pattern design considering SOA design principle and data integration perspective.

II. Literature Review

A. *SOA, Web Services, and BPEL*

There are many different definitions of SOA that vary according to the area under study. In this paper, we simply define SOA as a *software architecture* that provides flexible integration pattern of software components and optimize reusability that can reduce effort in software development. In the real world implementations, these two terms (i.e. flexible and reusable) are expanded into various paradigms such as stateless, loosely coupling, and composable.

SOA has become a widely used concept in software industry [4]. Today, SOA are an active research field with significant progress in web services (WS) based technologies [5]. Web Services technology was a choice to implement SOA because of its interoperability among various platforms. World Wide Web Consortium (W3C) introduced web services standard architecture in 2004 [6]. It is prominent technology to make application functionalities (business logic) accessible by others.

On top of Web services, the Business Process Execution Language for Web Services (BPEL4WS, or BPEL for short) becoming widely syntax for representing readable business process for automation [7]. BPEL drives communication of web services in order to deliver business services that contain several interconnected processes. Web service address interoperability issues in service level, whereas BPEL orchestrate web services in process level.

B. *Problems of current SOA technology to be used in data-intensive application systems*

Web services adopted eXtensible Markup Language (XML) to represent data structure attached in web services messaging protocol called Simple Object Access Protocol (SOAP). The requester sends the data to web services in form of function parameters and receives output as return value variable. Putting a bunch of data as parameter or attachment results the serious problem on parsing large quantities of data [4].

Transforming data from data storages (e.g. relational database) to XML format and vice versa continuously causes data synchronization problems when handling concurrent data update operations. If we want to ensure that a BPEL process always uses the latest version of data, we have to implement data synchronization manually [8]. Adding such data synchronization and data checking codes in several places in the business logic can lead to a bad process design with code duplication and the code size and complexity are increased [8].

In data-centric business, data becomes primary focus which its behaviors have to be intensively managed. Data flow has to be controlled centrally to ensure data integrity and follow correct treatment in data validation process. There are two major aspects, which are not covered in current web service and BPEL, which should be considered when treating data as a first class citizen [9]. First, data flows should be explicitly handled and modelled within the process description at design time. The second aspect is the need of specialized data transfer mechanisms that enable these modelled data flows executed efficiently.

C. *Related Work*

Most of authors, who have identified the problem of inefficient data access and manipulation in BPEL processes, claim that BPEL is not appropriate for data-intensive processes where large amount of data are passed throughout the process and between business partners [8]. In line with it, several researchers have proposed various approaches to improve service and process level support.

Habich et al have presented their Data-grey-box web service, an extension of web services, as a solution to optimize data transfer between web services endpoints [10]. Continuing their works, they then proposed BPEL extension called BPEL-Data Transition (BPEL-DT) for data-grey-box orchestration purpose [8]. A study that concerned with data flows has also been offered by them to treat data as primary business concentration [9]. Their recent research using this approach focused on data-flow optimization on cloud technology [11]. However, their web services extension cannot be implemented in a standard-conform manner because contradicts the web service semantics [12].

Another approach for data-centric process model based on Business Artifacts (BAs) has been proposed by the research group of Hull et al. In contrast with Habich, they started the study from business operation and process modeling using business artifacts approach [13]. The business artifact centric approach considers data as an integral part of business processes models, and it defines the process model and its operations in terms of interacting key business artifacts [1]. A data-centric web services model that integrates functional and data perspective, named Web Data- and Artifact- centric Service (W-DAS), which based on previously introduced the Guard-Stage-Milestone (GSM) approach for specifying Business Entity Lifecycles (BEL's) was introduced in another research [14]. Their web service extension exposed direct data access to client. Therefore, it caused another flexibility problem because of tightly coupled between data sources and composite applications.

Krizevnik and Juric handled data synchronization issues using their proposed BPEL variable, called *data-bound variables* [8]. The variables are automatically synchronized with the data in data sources. Thus, checking data validity manually can be excluded from BPEL design to improve the BPEL process design. Similar approaches attaching or modifying BPEL variables for data purposes have been previously done in BPEL-D [15] and RRS [2], but this kind of approach contradicts with the BPEL semantics where BPEL variables are fundamentally used to share data instead of treating them as data references [12].

There is also a research which attempts to optimize data transfer between web services by exploiting standard SOAP. Stream-based web services invocation has been proposed to solve performance problems and heavy resource consumption when services are used to process large amounts of data [3]. It changes the existing request–response paradigm in web services invocation into stream semantics. The evaluation shows great performance compared to traditional invocation mechanism. Another work used data-partitioning communication pattern that enables efficient flow of large data traffic between a workflow orchestrator and web services without modifying standard SOAP message [16].

On the other hand, researchers in grid technology represent The Open Grid Services Architecture (OGSA) which is a Grid system architecture based on web services concepts and technologies [17]. There are high-level components of OGSA for providing service-based platform such as OGSA-DQP (Distributed Query Processor) [18] and OGSA-DAI (Data Access and Integration) [19]. Service-based grid made data, which is scattered across various platforms, structures, and locations, available and accessible to distributed sets of users and their applications in comfortable manner. In many business domains, Grids and SOA are considered to improve application design, integration and execution [20]. There are many recent advances extending OGSA approach in grid technology to improve performance in data management (e.g. BlobSeer [21], and MAPFS-Grid [22]) that can be the solutions to data access and integration mechanism.

Centralized and distributed are well-known opposed approaches in computer science. In SOA implementation strategy, *centralized SOA* means all SOA participants depend on centralized control point. The composite applications cannot work without contacting the server to invoke the required services. In contrast, *distributed SOA* uses reachable local resources to deliver high available functionality, including the data, which is needed by these applications. Although distributed approach seems to have complex management, it should provide transparent resource access mechanism to users [23].

There are several studies mainly oriented to the implementation of distributed SOA. Some of them (i.e. Research on Distributed Architecture Based on SOA [23], and The SOA-Based Solution for Distributed Enterprise Application Integration [24]) gave basic ideas for application integration with distributed support. Their solutions need further researches and improvement to solve real world problems. Wang et al use proxy service as a gateway to route connections among distributed services [25]. By adopting OSGi framework, their approach can be a great solution to coordinate a number of clients in separated locations. A distributed replication of web service is an important aspect to obtain fault tolerant web services as shown in [26] and [27]. Moreover, security aspect of distributed system should be considered by implementing appropriate authentication and authorization mechanism [28].

According to our literature study, we can simply categorize these researches based on problems to be addressed into *Process and Data Flow Modelling*, *Data Transfer Mechanism*, *Data Access and Integration*, and *Distributed SOA*. We represent our discovered literatures in the following table.

TABLE I. LITERATURE DISTRIBUTION

Authors, Year	**Description**	**Problem Addressed**	**Contradicts web services / BPEL semantics**
Habich et al, 2008 Habich et al, 2009	Data-aware process execution	Process & Data Flow Modelling	Yes
Cohn et al, 2009	Process modelling based on Business Artifacts	Process & Data Flow Modelling	Yes
Wieland et al, 2009	Introduce data pointer variables in BPEL	Process & Data Flow Modelling	Yes
Krizevnik and Juric, 2012	Data-bound variables in BPEL	Process & Data Flow Modelling	Yes
Habich et al, 2007	Data propagation service model	Data Transfer Mechanism	No
Preißler et al, 2008	Stream-based Web Sevice Invocation	Data Transfer Mechanism	Yes
Sztromwasser et al, 2011	Data partitioning for Web Service	Data Transfer Mechanism	No
Vaculin et al, 2012	Web Data- and Artifact- centric Service	Data Transfer Mechanism	Yes
Foster, 2002 Antonioletti et al, 2005 Lynden et al, 2009	Enabling service-oriented in Grid technology	Data Access & Integration	No
Nicolae et al, 2010 Sánchez et al, 2010 Wöhrer et al, 2014	High performance access in Grid service	Data Access & Integration	No
Desmet et al, 2012 Ma et al, 2013 Xiong et al, 2013	Resources allocation & data replication	Data Access & Integration	No
He et al, 2009 Li & Wu, 2009	Basic ideas of application integration in distributed SOA	Distributed SOA	No
Wang et al, 2010	Adopting OSGi framework in SOA	Distributed SOA	No
Tang et al, 2009 Zheng & Lyu, 2008	Web services replication	Distributed SOA	No
Qi-rui et al, 2010	Authentication & Authorization in distributed systems	Distributed SOA	No

III. SOA Implementation Pattern Design

A. Design Overview

We suppose that a multi-layer SOA design can be a solution to preserve web service and BPEL semantics when used in distributed data-intensive application system. We propose a *Multi-layer SOA implementation pattern with service and data proxies for distributed data-intensive application system* which provides synchronization, replication, and routing mechanism in either data or service layer.

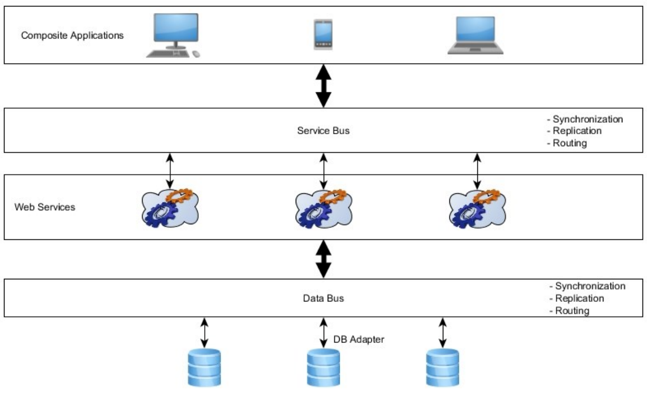

Fig. 1. Multi-layer SOA implementation pattern for distributed data-intensive application system

Data bus provides transparent data source access to the web services. Hence, web services can access the data using single port without knowing any details about data source, such as location, detailed query structure, and physical storages specifications. There are synchronization, replication and routing strategies behind the data bus. Physical data sources are replicated in different locations and accumulated centrally. Synchronization ensures data integrity by coordinating several data schema. Any incoming connections to access the data should be routed to appropriate data storage using certain routing mechanism.

Similarly, Service bus controls web service invocations from composite applications. Web services, which contain business logic, are replicated from central service repository to the local repositories that are closer to the corresponding client. The composite applications not need to be manually configured to find proper location to access required services.

B. Synchronization

Scheduled synchronization can optimize bandwidth usage and prevent system from bottleneck. We propose a mechanism to ensure the data and web service logic are consistent between local and central repository. Web service versioning are well-considered to make sure that new change of logic can be used as soon as possible.

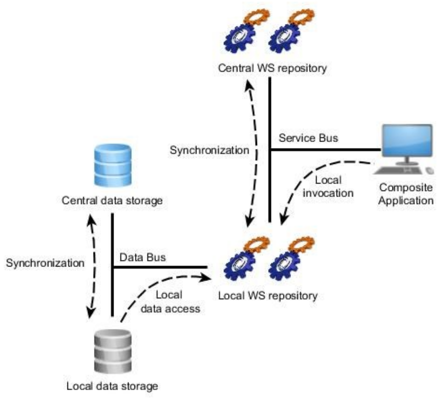

Fig. 2. Data and web service synchronization

Composite application invokes local web service which is connected to the data bus. Web service logic is serialized and loaded to the runtime environment using particular mechanism (e.g. Java Reflection API and dynamic class loader). Serialization converts web service logic into native binary format that can be stored to the secondary storage. Thus, it can be treated like data.

C. Replication

Data and web service are replicated using caching mechanism. They are placed close to the requester. Data replication depends on corresponding web service replication. They are stored in the local storage filling provided space and continuously renewed based on certain priority value. In case of long running data replication, data request are routed to either local or central data storage while replication process is running.

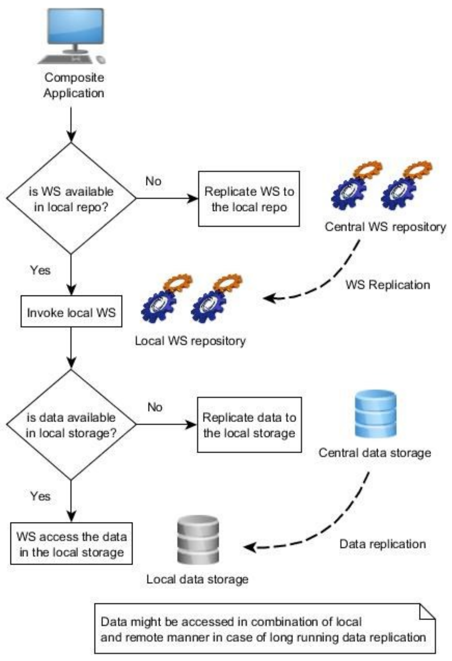
Composite Application
is WS available in local repo?
No
Replicate WS to the local repo
Central WS repository
Yes
WS Replication
Invoke local WS
Local WS repository
is data available in local storage?
No
Replicate data to the local storage
Central data storage
Yes
WS access the data in the local storage
Local data storage
Data replication
Data might be accessed in combination of local and remote manner in case of long running data replication

Fig. 3. Data and web service replication

D. Routing

Routing plays important role in distributed system. It manages the network traffic and point requests to appropriate destination. Proxy service presented in our literature review [25] is an example of routing mechanism. In our design, routing mechanism resides inside the data and service bus. It checks the availability of resources in local, central, or another near repositories and decides which resources to be accessed by the requester efficiently.

IV. Conclusion and Future Work

We proposed a SOA implementation pattern which deals with distributed data-intensive application system. It is intended to be suitable for current SOA-related standards technologies (i.e. Web Services and BPEL). We avoid changing the existing SOA standards technologies by using multi-layer pattern that separates composite applications, web services, and data sources. It provides transparent resources (i.e. data and web service logics) access using combination of synchronization, replication, and routing mechanisms. Data integration is performed automatically, thus data can be viewed as a unity and does not need to be manually checked in BPEL syntax to ensure its integrity.

Our proposed design presented in this paper is preliminary result of our research. Our proposed design needs further research, especially in replication mechanism, followed by realization for evaluation purpose. Well-designed mechanism will result easy configuration, increase availability, and improve system performance.

References

[1] R. Vaculin, T. Heath, and R. Hull, "Data-centric Web Services Based on Business Artifacts," in *2012 IEEE 19th International Conference on Web Services*, 2012, no. 1, pp. 42–49.

[2] M. Wieland, K. Görlach, D. Schumm, F. Leymann, and G. Katharina, "Towards Reference Passing in Web Service and Workflow-Based Applications Towards Reference Passing in Web Service and Workflow-based Applications," 2009.

[3] S. Preißler, H. Voigt, D. Habich, and W. Lehner, "Stream-Based Web Service Invocation," *Proceeding Datenbanksysteme Business, Technol. und Web (BTW 2009),*, vol. 2012, pp. 407–417.

[4] M. A. N. Yi, Z. Wen, D. U. Chen-hui, and P. A. N. Yang-fa, "Research on bulk data transfer on OSS enterprise service bus," *J. China Univ. Posts Telecommun.*, vol. 19, no. June, pp. 112–115, 2012.

[5] M. García-valls, P. Uriol-resuela, F. Ibáñez-vázquez, and P. Basanta-val, "Low complexity reconfiguration for real-time data-intensive service-oriented applications," *J. Futur. Gener. Comput. Syst.*, 2013.

[6] D. O. D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, "Web Services Architecture, W3C Working Group Note, 2004, http://www.w3.org/TR/2004/NOTE-ws-arch- 20040211," 2004.

[7] OASIS, "OASIS Web Services Business Process Execution Language (WSBPEL)," 2007. .

[8] M. Krizevnik and M. B. Juric, "Data-bound variables for WS-BPEL executable processes," *Comput. Lang. , Syst. Struct.*, vol. 38, no. 4, pp. 279–299, 2012.

[9] D. Habich, S. Preissler, H. Voigt, and W. Lehner, "INNOVATIVE PROCESS EXECUTION IN SERVICE-ORIENTED ENVIRONMENTS," in *International Conference on Enterprise Information Systems*, 2009, pp. 299–302.

[10] U. A. Dirk Habich, Ste en Preissler, Wolfgang Lehner, Sebastian Richly and and A. M. Mike Grasselt, "Data-grey-box web services in data centric environments," in *Proceedings of the 2007 International Conference on Web Services (ICWS 2007)*, 2007, pp. 976–983.

[11] U. H. D. and L. W. R. S. Assmann, "Using Cloud Technologies to Optimize Data-Intensive Service Applications," in *2010 IEEE 3rd International Conference on Cloud Computing*, 2010, pp. 19 – 26.

[12] O. Kopp, K. Görlach, D. Karastoyanova, F. Leymann, M. Reiter, D. Schumm, M. Sonntag, S. Strauch, T. Unger, M. Wieland, and R. Khalaf, "A Classification of BPEL Extensions," *J. Syst. Integr.*, pp. 3–28, 2011.

[13] D. Cohn and R. Hull, "Business Artifacts : A Data-centric Approach to Modeling Business Operations and Processes," in *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2009, vol. 32, no. 3, pp. 1–7.

[14] R. Hull, E. Damaggio, F. Fournier, M. Gupta, A. Nigam, P. Sukaviriya, and R. Vaculin, "Introducing the Guard-Stage-Milestone Approach for Specifying Business Entity Lifecycles," in *Proceedings of International Workshop onWeb Services and FormalMethods (WS-FM)*, 2010, no. 257593, pp. 1–22.

[15] F. Leymann, "Role-based Decomposition of Business Processes using BPEL Rania Khalaf," *Int. Conf. Web Serv. 2006. ICWS '06*, pp. 770 – 780, 2006.

[16] K. Petersen, "Data partitioning enables the use of standard SOAP Web Services in genome-scale workflows . 1 Introduction," vol. 8, no. 2, 2011.

[17] I. Foster, M. Jeffrey, and S. Tuecke, "Grid Services for Distributed System Integration," no. June, pp. 37–46, 2002.

[18] S. Lynden, A. Mukherjee, A. C. Hume, A. a. a. Fernandes, N. W. Paton, R. Sakellariou, and P. Watson, "The design and implementation of OGSA-DQP: A service-based distributed query processor," *Futur. Gener. Comput. Syst.*, vol. 25, no. 3, pp. 224–236, Mar. 2009.

[19] M. Antonioletti, M. Atkinson, R. Baxter, A. Borley, N. P. Chue, B. Collins, N. Hardman, A. Hume, A. Knox, M. Jackson, A. Krause, S. Laws, J. Magowan, N. W. Paton, D. Pearson, T. Sugden, P. Watson, and M. Westhead, "The Design and Implementation of Grid Database Services in OGSA-DAI," *Concurr. Comput. Pract. Exp. - Grid Perform.*, vol. 17, no. 2–4, pp. 357 – 376, 2005.

[20] S. Desmet, B. Volckaert, and F. De Turck, "Design of a service oriented architecture for efficient resource allocation in media environments," *J. Futur. Gener. Comput. Syst.*, vol. 28, no. 3, pp. 527–532, 2012.

[21] B. Nicolae, G. Antoniu, L. Bougé, D. Moise, and A. Carpen-Amarie, "BlobSeer: Next-generation data management for large scale infrastructures," *J. Parallel Distrib. Comput.*, vol. 71, no. 2, pp. 169–184, Feb. 2001.

[22] A. Sánchez, M. S. Pérez, J. Montes, and T. Cortes, "A high performance suite of data services for grids," *J. Futur. Gener. Comput. Syst.*, vol. 26, no. 4, pp. 622–632, 2010.

[23] H. Li and Z. Wu, "Research on Distributed Architecture Based on SOA," *Int. Conf. Commun. Softw. Networks, 2009. ICCSN '09*, pp. 670–674, 2009.

[24] X. He, H. Li, Q. Ding, and Z. Wu, "The SOA-Based Solution for Distributed Enterprise Application Integration," pp. 1–7, 2009.

[25] Y. Wang, M. Song, and J. Song, "AN EXTENDED DISTRIBUTED OSGI ARCHITECTURE FOR IMPLEMENTATION OF SOA," *Conf. Adv. Intell. Awarenss Internet (AIAI 2010)*, pp. 416 – 419, 2010.

[26] C. Tang, Q. Li, B. Hua, and A. Liu, "Developing Reliable Web Services Using Independent Replicas," *2009 Fifth Int. Conf. Semant. Knowl. Grid*, pp. 330–333, 2009.

[27] Z. Zheng and M. R. Lyu, "A Distributed Replication Strategy Evaluation and Selection Framework for Fault Tolerant Web Services," *2008 IEEE Int. Conf. Web Serv.*, pp. 145–152, Sep. 2008.

[28] P. Qi-rui, W. Cheng, W. U. Jing, L. I. Jun, L. I. Qing, and S. Bei-en, "An Authentication and Authorization Solution Supporting SOA-based Distributed Systems," no. 2007, pp. 535–538, 2010.