SSCM: Middleware for Structure-based Service Collaboration

Dianfu Ma Min Liu Yongwang Zhao Chunyang Hu
State Key Laboratory of Software Development Environment of Beihang University
P.O. Box 7-28, 37# Xueyuan Road, Haidian District, Beijing 100083,P.R.China.
dfma@nlsde.buaa.edu.cn liumin@act.buaa.edu.cn zhaoyw@act.buaa.edu.cn hucy@act.buaa.edu.cn

ABSTRACT

The collaboration of services scattered over the internet is often required in solving a complicated scientific problem or finishing a complex business process. Centralized client/server architecture used in traditional service collaboration mechanism such as workflow has exhibited many weaknesses. We adopt the concept from peer-to-peer computing and Synergetics, and introduce an innovative middleware for structured-based service collaboration, in which each service node belongs to one or more structured collaboration organizations and is allowed to join and leave (planned departure or sudden failure) freely. We also propose neighborhood-based maintenance mechanism to cope with the dynamics of collaboration organization structure in a decentralized manner.

Categories and Subject Descriptors

C.2.4 [**COMPUTER-COMMUNICATION NETWORKS**]: Distributed System–*Distributed applications.*

D.2.11 [**SOFTWARE ENGINEERING**]: Software Architectures–*Patterns.*

General Terms

Design, Algorithms, Management

Keywords

Service Collaboration, Decentralization, Structure Maintenance.

1. INTRODUCTION

The use of Web Service [4] greatly impels the interoperability and collaboration in distributed computing environment. The traditional technology such as workflow, used in the service collaboration, is centralized and has many defects, e.g. single-fault, poor scalability, bottleneck, etc.

On the other hand, the peer-to-peer systems and applications are distributed systems with no centralized control. The sharing of resources and services is achieved through direct communication between systems. The reliability and scalability of current service-oriented computing architecture will be enhanced by introducing the concept of peer-to-peer computing into the service collaboration mechanism. Our work was also inspired by the perspective of Synergetics [3], whose thought is that every system consists of collaborative subsystems, which are organized according to a certain structure for specific function and tend to restructure automatically to regain equilibrium when they are out-of-order.

In the paper [1], by adopting the concept from peer-to-peer computing and Synergetics, we introduce a structure-based architecture for service collaboration, and propose a neighborhood-based maintenance mechanism for the dynamic structured collaboration organization.

2. SSCM MODEL

In our service collaboration architecture, each service node can invoke services residing in other nodes or be invoked by other nodes. And like human society, collaborating service nodes are aggregated into a structured organization, which could be of star-structure, tree-structure, line-structure, ring-structure, graph-structure or a compound of above. (See Figure 1)

Any node may belong to more than one organization, as shown in Figure 1, node A, B, C from three different organizations are aggregated into another organization of ring-structure.

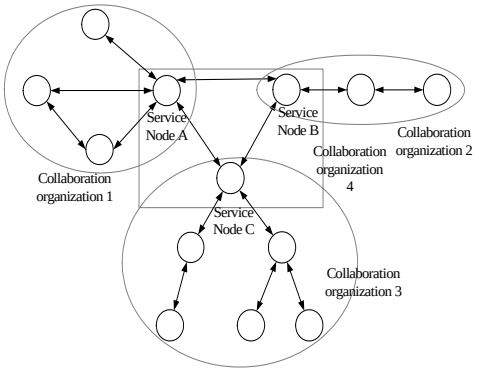

Figure 1 Structured service organization for collaboration

Moreover, the service nodes are allowed to join or leave (planned departure or sudden failure) a collaborating organization at any moment. This dynamic nature increases the scalability, flexibility and adaptability of service-oriented system while compounds the problem of structure maintenance for service organization.

SAC'08, March 16-20, 2008, Fortaleza, Ceará, Brazil.

[1] Our work is supported by grants from the China 863 High-tech Programme (Project No. 2006AA01Z19A), China 973 Fundamental R&D Program (No. 2005CB321903) and China National Science Foundation (No.90412011).

3. SSCM ARCHITECTURE

3.1 Core Components

In SSCM architecture as shown in Figure 2, there are three core components, user interface, structure maintenance, node behavior Engine.

- *User interface*: node administrator could configure the node through it, e.g., defining the node behavior and relationship with other nodes, selecting the structure maintenance strategy and etc.

- *Node behavior Engine*: it handles the service requests from other nodes and invokes the services in other nodes under certain conditions according to the node behavior specification, which is described using BPEL [1].

- *Structure maintenance*: it is responsible for reconstructing the node relationship represented with BPEL [1] when new nodes join or existed nodes leave intentionally even fail quietly. Besides, the node failure also needs to be detected in time.

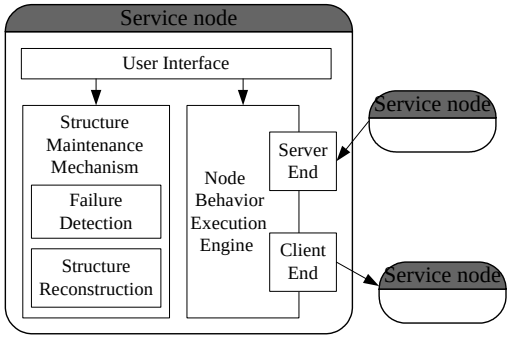

Figure 2 The functional framework of service node

3.2 Structure Maintenance

For space limit, the maintenance mechanism for new nodes arrival and existing nodes planned departure is skipped over in the paper. Only the case of node failure is discussed in detail.

Firstly, the node failure should be detected effectively. Here, we assume the node fails in a crash manner, just halting (non-Byzantine). As there is no master node, the node status (up or down) could only be observed by its neighbors through keep-alive messages

There are two different approaches to keep-alive messages. One is gossip [2] approach, in which a node periodically sends alive messages to its neighbors and the loss of regular alive message implies the possible node failure. The other is probe approach, in which a node periodically probes each neighbor and expects a prompt response. If receiving no response in a reasonable time, the node learns the neighbor is likely to have failed. The gossip approach obviously saves a half message communication than the probe one, but it is ill-suited in the case of asymmetrical connectivity, perhaps caused by NAT or firewall. In some cases, two approaches are used complementarily. For instance, if a node does not receive expected gossip messages from its neighbor, the node will probe the neighbor additionally.

After detecting the possible node failure, it steps to failure conclusion making stage. There are two broad categories of approach: non-cooperative and cooperative approach. Without loss of generality, our following discussions are all based on the probe keep-alive message approach mentioned above. The nodes being probed are called target node in the following text.

In non-cooperative approach, each node makes failure decision of target nodes all by its own observation on the probe losses. It derives the target node failure from the loss of $C_{probeloss}$ probes. The larger parameter of $C_{probeloss}$ results in smaller false positive probability but longer decision time. The cooperative approach is to let the adjacent nodes form a cooperative group to share their observations for the failure decision making. The positive information (target node is alive) reduces the probability of false positive while the negative information (target node is failed) speed failure decision. However both cases increase the control overhead because each node should keep additional member-list for each cooperative group it belongs to and extra communications are needed.

When detecting its neighbor failure, a node must contact with other nodes alive to restore the organization's structure. Especially when concurrent node failures occur, each node should prepare several substitutes in advance, i.e. each node should know the nodes in neighborhood besides the direct neighbors. Obviously, a larger number of nodes each node knows results in a higher maintainability. The relationship between them is out of the scope of this paper and is discussed in detail elsewhere.

In a line or a ring, if a node A detects its neighbor B's failure, and then it tries to contact with the node besides B along the line or the ring. Finally, A would link with a node alive in its neighborhood. Otherwise, the line or the ring can not be reconstructed. Similar to the ring-structure, if a tree node detected its parent's failure, it will try to contact with its grandparent and finally it contacts with its nearest ancestor alive, otherwise the tree is failed to be reconstructed.

4. CONCLUSION & FUTURE WORK

In this paper, we have made an exploration of Structure-based Service Collaboration Architecture, in which service nodes are aggregated into structured collaboration organizations. We also explained about the neighborhood-based structure maintenance mechanism which is implemented using BPEL [1]. SSCM can be applied to many fields, such as resource sharing among the collaborative resource holder that are organized into a ring or content delivery along end systems which are usually organized into a tree. In our future work, the structure maintainability of the service organization will be studied.

5. REFERENCES

[1] Tony Andrews, Francisco Curbera, Hitesh Dholakia et al, "Business Process Execution Language for Web Service. v1.1", http://www.ibm.com/developerworks/library/ws-bpel/, 2003.5.

[2] R. van Renesse, Y. Minsky, and M. Hayden. A gossip-based failure detection service. In *Middleware 1998.*

[3] H.Haken, "Synergetics: An Introduction", Springer, Berlin, 1978.

[4] Kun Yue, Xiaoling Wang, and Aoying Zhou, "Underlying Techniques for Web Services: A Survey", Journal of Software, 2004