0-003

User-Centered Dynamic Service Invocation Control

林 冬惠† 村上 陽平† 田仲 正弘†
Donghui Lin Yohei Murakami Masahiro Tanaka

1. Introduction

Service composition environments enable people to create, manage services, and share their services with each other, while users can get additional value of services by composing them based on various requirements. The Language Grid [1] is a typical example of service composition environment in the domain of language services, which uses the collective intelligence approach to gather service users and providers together, and coordinate their incentives.

Although service composition environment like the Language Grid provides easy access to services, there are several major problems with the increasing of users and services in the environment. First, since physical locations of users and service entity hosts are distributed in different areas around the world, invocation response of services might be very slow if users invoke services that are physically far from them. Moreover, when invoking a composite service that consists of several atomic services, the invocation response is always much slower if all those atomic services are located in service entity hosts that are far from each other. Second, service providers in the service composition environment always have their own policies to provide services. One typical policy is about service license issues. Therefore, it is difficult to apply previous approaches like caching technologies in the areas of contents delivery network (CDN) [3] to the service composition environments since service entities are always not allowed to be copied on all service entity hosts due to license constraints by the providers.

In this paper, we aim to design dynamic service invocation control mechanisms to improve response performance of services while considering service license constraints by service providers. To achieve this objective, we propose the approach of dynamically controlling (switching) availability of services among service entity hosts based on the invocation request from users. For atomic services, we propose several mechanisms including the request-based invocation control, the user-based invocation control and the hybrid invocation control that combines the above two mechanisms. For composite services, we propose the individual invocation control mechanism and the group invocation control mechanism.

2. Service Invocation Problem

We first use an example of the Language Grid to show the service invocation problem. The infrastructure of Language Grid is aimed at connecting two kinds of servers (core nodes and service nodes). Core nodes manage all requests to language services, while service nodes (or we call service entity hosts) actually invoke atomic services. If the requested service is a composite one as shown in Figure 1, core nodes invoke a corresponding Web service workflow that includes multiple atomic services (Service 1, 2, 3, 6, 7 in Figure 1).

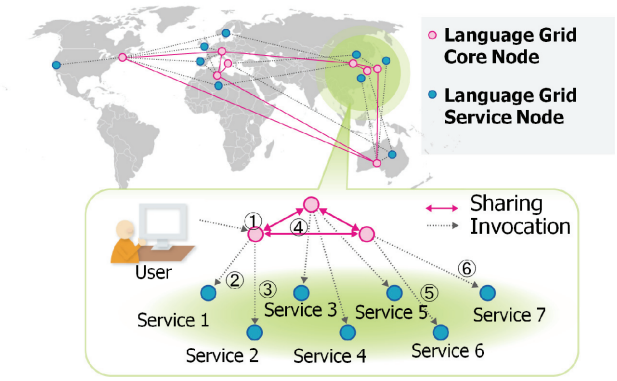

Figure 1: Example of the Language Grid

Several experiments have been conducted to investigate the response time for service invocation by Language Grid user from Thailand. The user invokes an atomic service (J-Server Japanese-English translation service) and a composite service (composite Japanese-English translation service combined with dictionary which is composed by three atomic services on the Language Grid) for 10 times from both the service entity host in Thailand that is near to the user and the service entity host in Japan that is far from the user. In the experiment, it takes 2.54 times and 4.87 times of response time for invoking an atomic service and a composite service respectively on the host in Japan than in Thailand for Thai user. Similar experiments have been conducted between Language Grid users in Japan and in Denmark. When invoking an atomic service on the service entity host in Japan, user from Denmark costs 5.39 times of response time.

3. Dynamic Service Invocation Control

To improve the response performance of service invocation with the service license constraints of service providers and distributed physical locations of users and services, we propose several dynamic service invocation control mechanisms.

3.1 Invocation control for atomic services

The most straight and simple approach for invocation control of atomic service is to dynamically make the service entity of a service available for invocation on the nearest service host to the user for each request, which is called *request-based invocation control*. However, the biggest problem of the request-based invocation control is that there might be many times of changing service host of service by service migration or license control and therefore it always costs much time to control the

†Language Grid Project, National Institute of Information and Communications Technology (NICT), Japan

availability of service entity among service host. To deal with this problem, we further propose the *user-based invocation control* mechanism to reduce the switching times among service hosts, where availability of service entities are controlled periodically based on the service invocation trends from users. However, although minimum cost of switching of service entity is required in user-based invocation control, it might lack of flexibility to deal with the cases when real service invocation request situation is not the same as what is anticipated. To consider both the switching cost of service hosts and flexibility of handling different service invocation request situations in real cases, we propose the *hybrid invocation control* mechanism by combining the above two approaches for different cases.

3.2 Invocation control for composite services

Simple invocation control mechanism for composite service can be realized by applying invocation control mechanisms for all individual atomic services that consist the composite service. However, since the individual invocation control mechanism does not even consider the existence of composite service and the group characteristics of atomic services, applying optimal invocation control mechanisms for atomic services individually does not guarantee that it is also optimal for the whole composite service. Therefore, we propose the group invocation control mechanism for composite service. That is, the group invocation control considers the group of atomic services that consist composite service together rather than separately.

The detailed definitions of the service invocation problem and algorithms of dynamic invocation control are described in [2].

4. Evaluations

4.1 Response performance of atomic services

The results shown in Figure 2 shows that request-based invocation control, user-based invocation control, hybrid invocation control mechanisms can all bring average stable response for different invocations. Moreover, the hybrid invocation control mechanism that considers both potential users for most service invocation requests and potential users for continuous requests performs best.

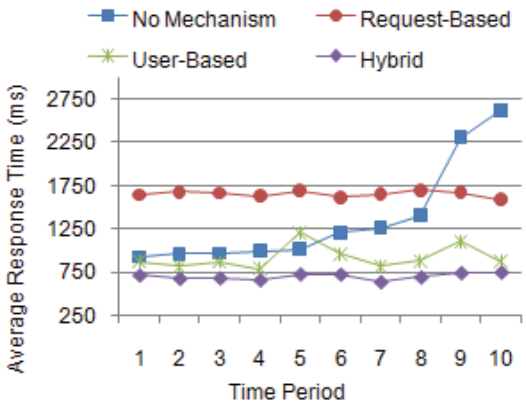

Figure 2: Response performance of atomic service

4.2 Response performance of composite services

Since hybrid invocation control mechanism for atomic service can bring the best response performance of atomic services, we use it in individual invocation control mechanism and group invocation control mechanism for composite services. In Figure 3, three atomic services compose the composite service we use for the experiments with sequential patterns. The result in Figure 3 shows that using group invocation control for composite service can get better response performance comparing to individual invocation control.

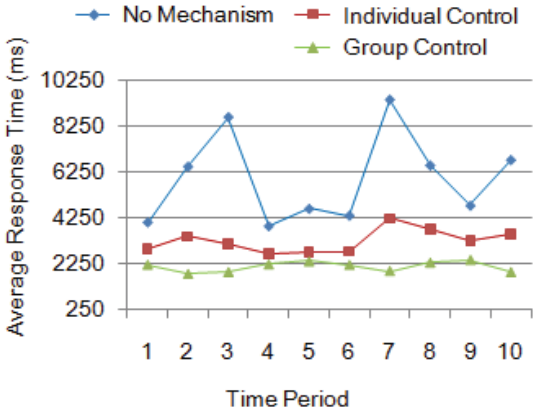

Figure 3: Response performance of composite service

5. Conclusion

This paper deals with the issues of service invocation control considering the service license constraints by service providers. We proposed the dynamic service invocation mechanisms in service composition environments based on examples in the real world for both atomic services and composite service to improve the response performance. The experimental results shows that (1) for atomic services, the invocation control mechanism that considers both potential users for most service invocation requests and potential users for continuous requests can improve the response performance better than other mechanisms; (2) For composite services, group invocation control can improve the response performance more than individual invocation control.

Acknowledgements

This work was supported by Strategic Information and Communications R&D Promotion Programme (SCOPE) from the Ministry of Internal Affairs and Communications of Japan.

References

[1] Ishida, T. Language Grid: An Infrastructure for Intercultural Collaboration. *IEEE/IPSJ Symposium on Applications and the Internet (SAINT-06)*, pp. 96-100, keynote address, 2006.

[2] Lin, D., Murakami, Y., and Tanaka, M. Dynamic Service Invocation Control in Service Composition Environments. *Proceedings of 2010 International Conference on Services Computing (SCC2010)*, 2010.

[3] Pathan, M. and Buyya, R.: A taxonomy of CDNs, *Content Delivery Networks*, pp.33–78 (2008).