Design and Implementation of Mobile Proxy

Case Study: Mobile Data Collection

Aris Prawisudatama
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
soedomoto@gmail.com

I Gusti Bagus Baskara Nugraha
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
baskara@stei.itb.ac.id

***Abstract*—Data collection using mobile device is something that is quite common recently. However, most of those applications are using static validation rules. The use of mobile proxy to duplicate rule and data is believed could improve the flexibility and implementation of the validation rule. Rule could be easily designed in modular basis, installed, updated, as well as deleted from application using OSGi framework. This paper is aimed to research about the design and implementation of mobile proxy that could be used to implement dynamic validation rule on SOA-based mobile device.**

Keywords—proxy; mobile proxy; dynamic proxy; service oriented architecture; SOA; REST

I. INTRODUCTION

Data collection is one of the main tasks of statistical institution in the world, including BPS-Statistics Indonesia. Data collection process is conducted using various media, ranging from conventional method, such as paper questionnaire to the modern method using mobile device. Until now, most of data collection conducted by BPS-Statistics Indonesia are still using paper questionnaire. However, BPS is now starting using mobile device for collecting data.

In data collection, the most important thing to be considered is the data quality. Data quality is measured using consistency and validation parameter. Data collection using paper questionnaire still rely on human's carefulness and thoroughness in implementing the rule. Using mobile device has a benefit related to consistency and validation. Thus, generally, using mobile device could increase data quality.

In consistency and validation, rules that manage workflow between questions are necessary. Rule is based on concepts and definitions adopted from the questionnaire. Rules implementation on a mobile based data collection are usually embedded on the applications that is used. In fact, the rule can be changed at any time during the implementation of data collection takes place, so that the rule contained in the application must be updated. Alternative solutions that can be used is to implement the concept of Service Oriented Architecture (SOA) using Web services.

SOA implementation on a mobile device requires a server running Web service, or called as a service provider. Service provider run the service in the form of the rule of consistency and validation. Meanwhile, an application on the mobile device acts as a consumer service, which sends the data to be validated and saved to the Web service. In reality, the data collection is a condition that requires mobility. The obstacles faced by, among others, battery life and signal availability internet. The internet signal is necessary so that applications can still communicate with a Web service. In order to keep running on condition connectionless, applications need to duplicate the rule locally on the mobile device, so the application would be to consume Web services that are available locally

II. THEORETICAL BACKGROUND

A. Proxy

In the perspective of a distributed system, a proxy can be interpreted as a server that acts as a liaison between the client and the server. Proxy is a representation of the entire set of servers, resulting in conducting all communication, the client will connect directly to the proxy [1]. Proxies can be used for several purposes: as a proxy firewall located on the perimeter between the Internet and your private network, accelerate web performance, provide remote access to internal servers, and provide anonymity for network conversations [2].

There are several types of proxies are often used, for examples [2]:

1) Web Proxies

One of their most popular uses inside a network is increasing web performance. Some studies have shown that as much as half the requests for information across the Web are duplicates of other recent requests. Caching frequently requested web pages can dramatically speed up web browsing. Proxy servers that provide web caching are often referred to as proxy caches or web caches. When a proxy cache is used, browsers are directed to make their HTTP requests to the proxy cache instead of directly to the destination web server.

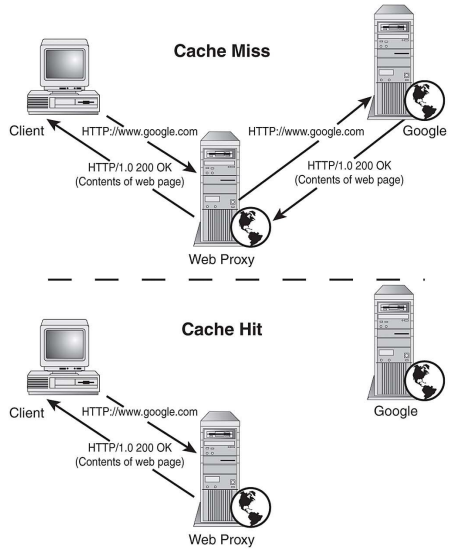

Fig. 1. Web Proxy Illustration

2) Reverse Proxies

Reverse proxies act as a trusted intermediary that external users must use to gain access to internal servers that would not normally be Internet accessible. Reverse proxy works by authenticating external user, and then give an access based on the rules.

3) Anonymizing Proxies

Anonymizing proxies work exactly like normal proxies, but are used for the purpose of protecting client identity while client use services across the Internet. Client requests are forwarded to the anonymizing proxy (usually over an SSL connection), which hides client identifying details (such as IP address) by making the request on client behalf. Illustration of anonymizing proxy is shown in Fig. 2.

B. Caching

One property of a proxy is a locality [1]. On the concept of locality, some of the client request will be responded by proxy, and the buffer will be stored locally by proxy in the form of cache. Cache can be regarded as a temporary object that can be used later [3]. Caching has several uses, such as [4]: increasing the availability of the data, reducing latency to the client, reducing the load on the server, and reduces bandwidth consumption.

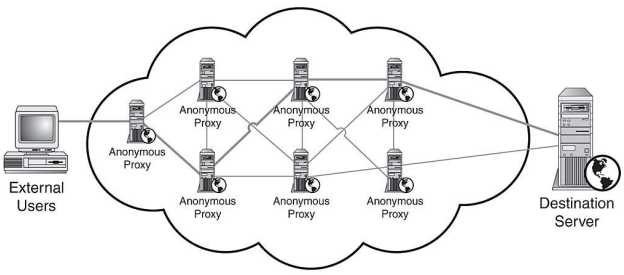

Fig. 2. Anonymous Proxy Illustration

In caching, there are two algorithm that are important, replacement algorithm and consistency algorithm [5]. Replacement algorithm is an algorithm used to determine which cache should be replaced and cache to be maintained. Replacement algorithm uses approach data that most widely used before. While consistency algorithm is an algorithm used to ensure consistency with the original resource cache on the server. Common algorithm to ensure the consistency of the cache is the Time to Live (TTL).

C. Mobile Proxy

Mobile proxy is a proxy server that runs on the mobile device. Mobile proxy to explore the capabilities of a proxy to be more dynamic. Implementation of mobile proxy is in the form of a mobile host that acts as a router for other nearby mobile devices [6].

D. Dynamic Proxy

Dynamic Proxy is a proxy that can be created automatically when needed. Cobarzan et al propose a novel proxy-cache system that can spawn new proxies when necessary by using a split operation [4]. How split operation works is a proxy that needs to be split sends a code to the other nodes to be transformed into a proxy.

Each node on the design Cobarzan et al consists of two components, which are daemon and the dispatcher. Specifically, the following function of each component:

1) Dispatcher

The dispatcher is the core of the proxy, and disabled by default. Here is a function of dispatchers:

- Handle incoming request, if not possible, the request will be forwarded to other dispatcher / proxy or request denied.
- Managing the proxy code. Proxy code is code that is sent to another node to be transformed into a child proxy of the proxy who initiated.
- Managing the child proxy process. Dispatchers have full control over every child proxy that is initiated: start, stop, pause, and restart.

2) Daemon

While daemon is a process that automatically runs on every

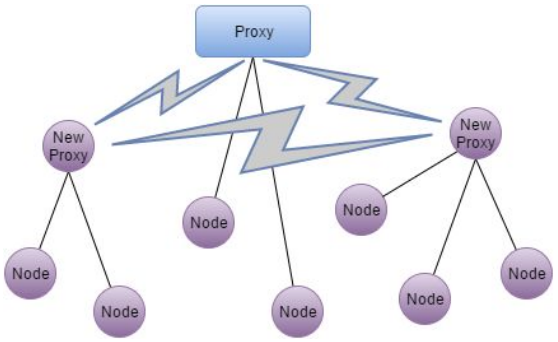

Fig. 3. Dynamic Proxy

node, and monitor a code sent by another proxy. The functions of the daemon are:

- Managing the proxy code. Daemon manage proxy code that is sent by a dispatcher who initiated.
- Managing these proxies. In addition to the parent proxy daemon will also have full control over the dispatcher at the nodes where it goes: start, stop, pause, and restart; depending on conditions on the node.

E. SOA

Service Oriented Architecture (SOA) is a way of designing software as interoperable software services. A service is a software component that is available over a network and can act either as a client or a server or be composed by combining other services [7]. Services in SOA are lightweight, loosely-coupled and communicate using messages over well-defined interfaces. Today, most SOAs are built using Web Services.

Because SOAs are built on light-weight and interoperable services, they are inherently flexible and scalable. The flexibility comes from the fact that services can be replaced according to changing needs and that new services can be created by combining other services. SOAs scale well thanks to their parallel nature as services can easily be duplicated or migrated as needed [8].

F. SOAP and REST

A SOA can be rendered through the use of any service-based technology, such as SOAP or REST. Compared to each other, SOAP is more mature and standardized, widely supported and transport-protocol independent while REST gives developers larger freedom of choice and is easy to get started with thanks to the agility of the design and the lightweight approach. REST also works well for applications with a Web Interface since the basic REST operations are supported by any Web Browser. However, REST lacks standards for security, quality of service (QoS), etc and support for business automation languages whereas SOAP is highly standardized and supported. A more in-depth discussion of these technologies is given by Pautasso et al [9].

From a performance point of view, REST has better cache support, lightweight requests and responses as well as easier response parsing. REST also supports a large number of parallel clients and servers and reduces network traffic due to less overhead [10].

G. Mobile Host and Mobile Web Service

Mobile hosts is a mobile device that is running a web server and act as a web service provider [11]. Implementation of Web service on the mobile host can be done by adopting the SOAP or REST architecture. Because mobile devices have limitations in resources, both CPU, memory, and battery life, it is necessary to consider the performance of these two architectures in mobile devices.

Wagh and Toll compare the performance of SOAP and REST architecture in mobile devices based on Android [12], with the following results:

- Both SOAP and REST architectures can run on Android-based mobile devices,
- While testing, both SOAP and REST implementations generate an error rate of 0.0%,
- REST-based Web service provides execution time is 2 to 4 milliseconds faster than SOAP in different conditions (load level and the number of requests).

H. OSGi

The OSGi standard is essentially a service-oriented component model. Managed software components deployed in the OSGi platform are called “bundles,” and the bundles can be installed, updated, or removed on the fly without having to disrupt the operation of the device. Bundles are libraries or applications that can dynamically discover other services from the service directory or can be used by other bundles [13].

OSGi implementation in the context of SOA is to consider each class of service as a bundle that can easily be patched with other runtime services. In the context of this study each bundle represents a validation rule and consistency are used in the application.

III. Mobile Proxy Design

A. Design Overview

Proposed proxy mobile design consists of three components, data, services, and applications, which are connected via two buses, the data bus and bus service. Illustration of design can be seen in Fig. 4.

Data bus Provides transparent access the data source for web services. On the data bus there are methods of replication, synchronization, and routing. Central storage of data contained in a central place. By application, storage will be replicated somewhere locally. Synchronization is used to ensure synchronous local storage to central storage. Meanwhile, the routing is used to direct incoming connections to the data

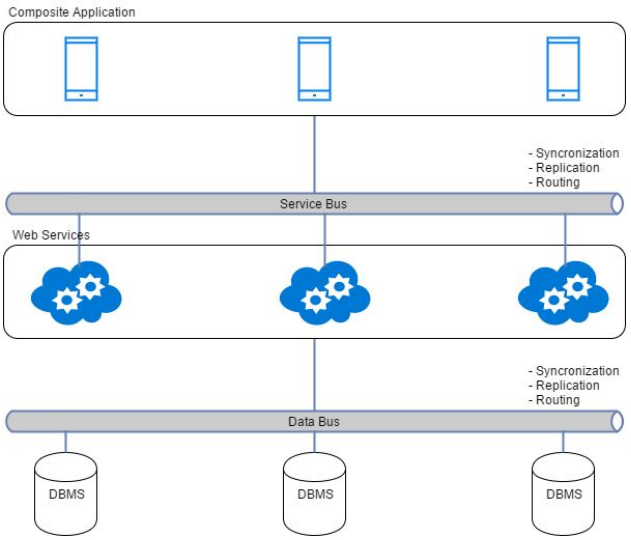

Fig. 4. Design Overview

storage by using a specific mechanism.

Same with the data bus, the bus service to handle web service invocation by the application. Web service that is a representation of the rule, will be placed at a centralized location. The application will then replicating it to a location locally and dynamically load and unload as needed. Synchronization ensures local web service in sync with a central repository.

B. Replication

Data and web service are replicated using caching mechanism. They are placed locally, same location with application. Data replication depends on corresponding web service replication. Each web services bundle needs different data each other. They are stored in the local storage filling provided space and continuously renewed based on certain priority value.

C. 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.

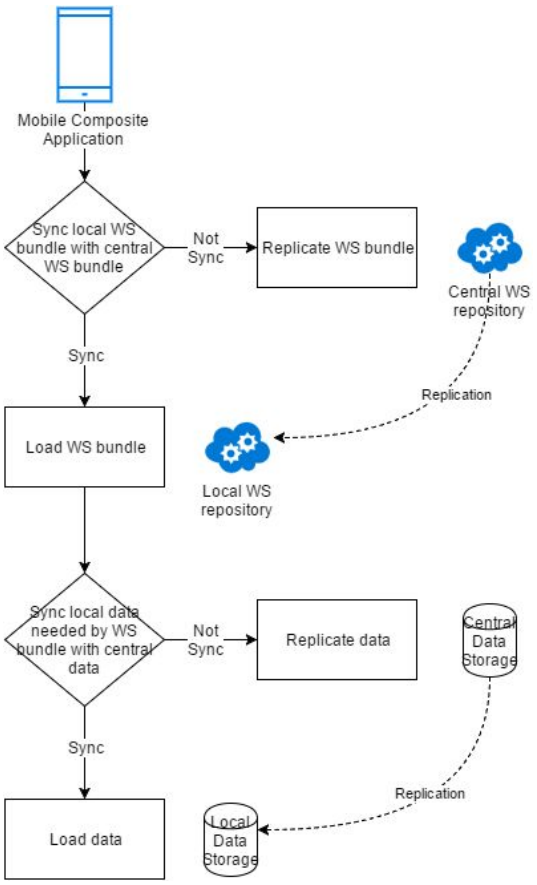

Fig. 5. Replication and Synchronization Mecahnism

Web service is packed using mechanism that enable distribution as a module, e.g. OSGi framework. A bundle of module is just a plain JAR, while in android it must be convert into dex contained JAR, since only dex is recognized by android. Bundle is then placed on a central repository that can explore by composite application. Composite application then compares the local bundle with a central repository with versioning mechanism. When the versions are different, then the composite application bundle will duplicate and load it dynamically using Activator in the OSGi framework.

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 [14] 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 mobile proxy design that allows the distribution web service dynamically. It is intended to be suitable for current SOA-related standards technologies. 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 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 in mobile device, especially android environment, followed by realization for evaluation purpose. Well-designed mechanism will result easy configuration, increase availability, and improve system performance.

References

[1] [1] M. Shapiro, “Structure and Encapsulation in Distributed Systems: the Proxy Principle,” presented at the Int. Conf. on Distr. Comp. Sys. (ICDCS), 1986, pp. 198--204.

[2] [2] S. Northcutt, K. Frederick, S. Winters, L. Zeltser, and R. W. Ritc, *Inside Network Perimeter Security*. Sams, 2005.

[3] [3] B. D. Davison, “A survey of proxy cache evaluation techniques,” in *Proceedings of the Fourth International Web Caching Workshop (WCW99)*, 1999, pp. 67–77.

[4] [4] C. Cobârzan, “Dynamic Proxy-Cache Multiplication Inside LANs,” in *Euro-Par 2005 Parallel Processing*, J. C. Cunha and P. D. Medeiros, Eds. Springer Berlin Heidelberg, 2005, pp. 890–900.

[5] [5] J. Shim, P. Scheuermann, and R. Vingralek, “Proxy cache algorithms: Design, implementation, and performance,” *Knowl. Data Eng. IEEE Trans. On*, vol. 11, no. 4, pp. 549–562, 1999.

[6] [6] B. Aziz and C. Jensen, “Adaptability in corba: The mobile proxy approach,” in *Distributed Objects and Applications, 2000. Proceedings. DOA’00. International Symposium on*, 2000, pp. 295–304.

[7] [7] P. F. Brown and R. M. Hamilton, “Reference model for service oriented architecture 1.0. Tech. rep.,” 12-Oct-2006. [Online]. Available:

http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf. [Accessed: 01-May-2016].

[8] [8] T. Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2005.

[9] [9] C. Pautasso, O. Zimmermann, and F. Leymann, "Restful Web Services vs. 'Big'' Web Services: Making the Right Architectural Decision," in *Proceedings of the 17th International Conference on World Wide Web*, New York, NY, USA, 2008, pp. 805–814.

[10] [10] P. G. Harrison and N. M. Patel, *Performance Modelling of Communication Networks and Computer Architectures (International Computer S*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1992.

[11] [11] S. N. Srirama, M. Jarke, and W. Prinz, "Mobile web service provisioning," in *null*, 2006, p. 120.

[12] [12] K. S.Wagh and R. C. Thool, "Web Service Provisioning on Android Mobile Host," *Int. J. Comput. Appl.*, vol. 81, no. 14, pp. 5–11, Nov. 2013.

[13] [13] "Architecture – OSGi™ Alliance." [Online]. Available: https://www.osgi.org/developer/architecture/. [Accessed: 15-May-2016].

[14] [14] Y. Wang, M. Song, and J. Song, "An extended distributed OSGi architecture for implementation of SOA," in *2010 International Conference on Advanced Intelligence and Awarenss Internet (AIAI 2010)*, 2010, pp. 416–419.