The Design and Architecture of CalDAV-based Enterprise Calendar Service

Erdong Ma, Naijia Liu, Yu Zhang, Yu Peng
Information Technology Center
Tsinghua University
Haidian District, Beijing, China
med@tsinghua.edu.cn, lnj@tsinghua.edu.cn, zhangyuthu@tsinghua.edu.cn, py@tsinghua.edu.cn

Abstract—**Personal calendar applications develop rapidly in recent years, while the enterprise calendar is lagging behind. This paper describes the protocol stack and content of standard calendar service, and then analyzes the similarities and differences between standard calendar and enterprise calendar from several perspectives such as the enterprise calendar scenarios, authorization structure and producing / consuming patterns of information. Considering the enterprise calendar features, we discuss the architectures of enterprise calendar from multiple angles - data architecture, authentication and authorization, daily management etc. Finally, we discuss several urgent tasks in the future.**

***Keywords*—Calendar; Enterprise Calendar; WebDAV; CalDAV; Event; Todo**

I. THE STANDARD CALENDAR SERVICE

A. *The Protocol Stack*

Calendar service include four international protocols: WebDAV, CalDAV, iCalendar and jCal.

WebDAV (Web Distributed Authoring and Versioning, network Distributed Authoring and Versioning) is a HTTP protocol extension developed by the IETF (Internet Engineering Task Force) which enables collaboration between users and makes the documents on the server editable [1]. CalDAV (Calendaring Extensions to WebDAV) is an extension to WebDAV, which defines the standard of managing / sharing calendar and scheduling information. CalDAV is the core protocol in the whole protocol stack[2].

iCalendar is the standard of calendar data exchange. It enables users to send "Meeting Request" or "Mission" via e-mail or sharing ics files. iCalendar standard is independent of the communication protocols. A calendar event could be sent by email, or you can share and edit it through WebDAV server [3]. Because of the inflexible structure and bad readability of iCalendar standard, people developed jCal standard using a more flexible JSON format to write iCalendar data [4]. jCal is still in the draft stage, however by virtue of its advantages, it is likely to replace the iCalendar and become the new industry standard in the coming future.

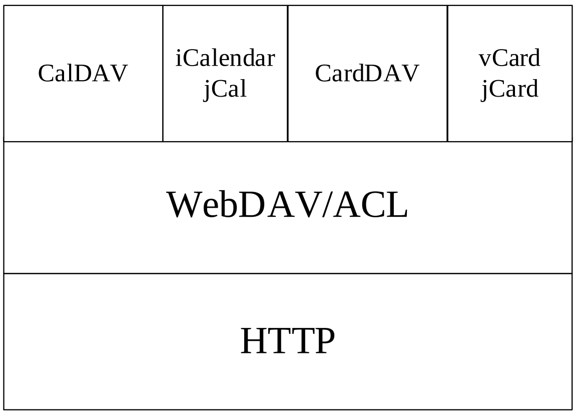

Figure 1. The Protocol Stack of Calendar Service

B. *The Content of Standard Calendar Service*

Calendar service has four core concepts: calendar, event, todo and journal. Calendar is a container, which includes events, todos and journals. A person can have multiple calendarsto organize different types of events and todos; Event may be one activity lasting for a period of time or something you want to do. A standard event should have attributes like name, location, start time and end time. Todo generally has a planned time and the actual completion time, it emphasizes more on the concept of time point than time period. Journal does not take time, it plays a role of organizing and taking memos. The basic features of the calendar service include the following four aspects: personal schedule maintenance, schedulesharing, events reminders and inviting people to your event.

II. ENTERPRISE CALENDAR SERVICE AND ITS FEATURES

Standard calendar service is designed for individuals or small groups, while enterprise calendar service targets organizations and companies. Enterprise calendar differs from standard calendar service in many ways.

A. *Usage Scenarios*

Enterprise calendar contains two parts: personal calendar and calendar subscription. The common scenarios of personal calendar include: calendar CRUD operations, CRUD of calendar events and todos, maintenance of repeated events and calendar sharing. Querying is the main usage scenario of calendar subscription[5]. Calendar publishers are often responsible of management, individuals have no right to modify. Technically, subscription data is often stored and maintained in business systems, calendar has read-only access to this part of data. But if the subscription data is stored locally in the calendar application, then you can grant CRUD rights for a particular user role.

B. *Authorization Structure*

Standard calendar service is an internet-oriented, loosely packed and disorganized structure. Apart from users' full control of their own calendars, there are lot of mutual authorizations among users. User groups based on the mutual authorizations may have star structure, net structure or a combination of both. In such authorization structures, promoters and maintainers are end users[6].

Enterprise calendar is organization-oriented. So it naturally has the same tree structure with normal organizations. Enterprise calendar users can still share personal calendars with other users. Taking into account the requirement for cross-departmental project, authorization structure would also includes horizontal, multiple departments authorizations. This kind of authorization, not the same with the way of standard calendar which is determined by individuals, generally combines with the approval systems of companies. When individuals' situations change, such as entry, transfer, termination etc., authorizations also require approval and execution of specific operation from the enterprise.

C. *Producing and Consuming Patterns of Information*

Information of calendar service mainly consists of events and todos. In standard calendar service, all of the information is generated by individuals. Information consumers may be individuals, or members of a group. If the group members have read and write access to a public calendar, these members are both producers and consumers at the same time.

Enterprise calendar information mainly comes from the enterprise's own MISs (Management Information System). Enterprise information technology has developed over years, and all companies already have information systems suitable for their own situations. These information systems have become an integral part of enterprises. They contain a lot of complex business logic, deal with and produce large-scale information in every second. Calendar service, as a relatively independent system, the best way to embed into the ecological environment of current enterprise information systems is purely playing a role as a new channel for information display, but not going deep into business logic. In this case, calendar content comes from the existing information systems, while individuals within the enterprise could read part or whole of them according to their own authorities.

III. ARCHITECTURE OF ENTERPRISE CALENDAR SERVICE

A. Data-layer Architecture

Calendar service is a standard way to expose data - the server displaying and receiving data via a standard way, the client subscribing data through the same way. As mentioned above, the standard calendar data mainly comes from individual inputs, while information systems are the main sources of enterprise calendar data. Previously, corporate events dispersed in various information systems, and the calendar service should break through the barriers between information systems and integrate them together. This kind of integration may use data synchronization mode or data sharing mode.

a) Data Synchronization Mode

The data of enterprise calendar services comes from other business systems, and business systems were often developed in different time periods and different technical architectures, so the data synchronization has become the most crucial and the most complex module in enterprise calendar.

Data synchronization can be divided into real-time synchronization and timing synchronization, in the sight of trigger types, can be divided into three categories: business trigger, user trigger and timing trigger, each of them could apply to a specific kind of scenario. You could also mix a variety of ways to ensure real-time, accurate and complete data. Given the importance and complexity of data synchronization, corporations generally need to create an independent dispatch center, located between calendar service and business systems, to coordinate a variety of synchronizations. Enterprises and individuals will generate information every second, at the same time vast amounts of data is expired or become invalid, so an appropriate cleanup and exit mechanism is necessary.

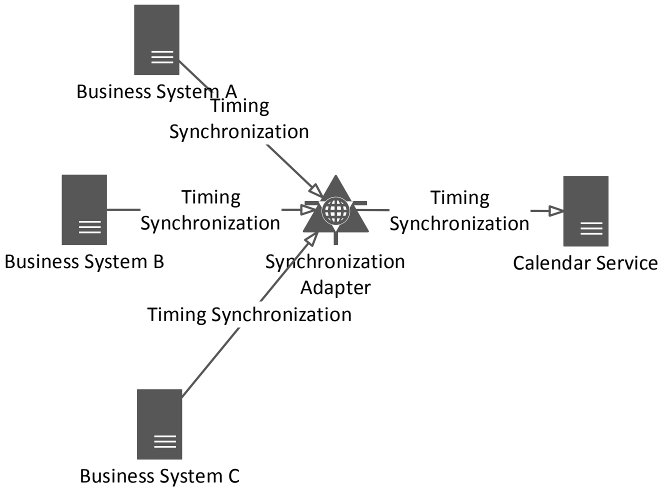

Figure 2. Data Synchronization Mode

b) Data Sharing Mode

The core concept of data synchronization is making a copy of data. If the data source changes, the copy must follow. The advantage of this model is simplicity, you can sidestep the complexity of business logic, start directly from the data layer. But the disadvantages are also obvious, it is almost impossible to achieve real-time data synchronization. A compromise could achieve quasi-real time synchronization through incremental synchronization. Logically, for two copies of the data, when working on large data sets and frequently changing conditions, real-time synchronization is a problem with no solution. At real-time critical scenario, data sharing mode is a better choice.

Data sharing mode makes sure there is only one copy of the data source in all systems. The data layer of calendar application needs to work with two kinds of data sources: the data in local database and remote services. Local data is stored in the local relational database. Remote data is stored in remote server and exposed through a standard way like restful + json. Here we have two difficulties: firstly, there is a lot of work to do to refactor the data layer, secondly the adaptation is necessary between business logic of business systems and data logic of calendar application. The goal is to make the modifications of business system as little as possible, while achieving the calendar application docking.

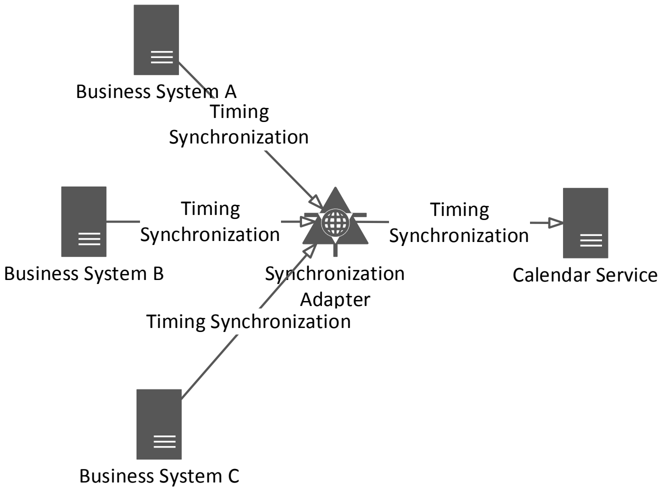

Figure 3. Data Sharing Mode

User trigger is a common option in data sharing mode - when a user logs in, application would read the data of current user from remote business systems in real time. The distinction between data sharing mode and data synchronizing mode is that calendar service does not store business data locally in the former.To reduce the modifications of the open-source calendar service solution, you can leverage in-memory database in practice. A pure memory database which has identical structure to the database of calendar service itself could be created in calendar server. Calendar service reads business data from business system, inserts it to the in-memory database, and then reads the data from disk database and in-memory database simultaneously. In this manner, the modifications of calendar service could be limited to the data layer, a large amount of legacy code could be reused, and the the deployment is easy to scale. Considering these advantages, it is a feasible way in practice.

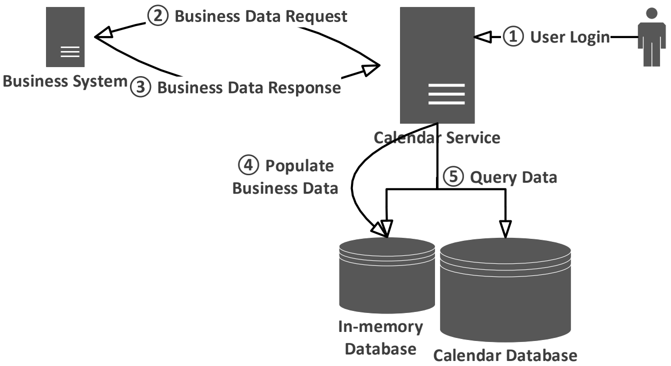

Figure 4. In-memory Database Architecture

B. *Authentication and Authorization*

The protocol stack of standard calendar service does not contain authentication part. Therefore, to complete the docking between calendar service and enterprise applications, authentication module is essential. Depending on the differences between the real scenarios, authentication can be divided into two types: client-server authentication and server-server authentication.

Web browsers need to maintain the stateful client-server communication through cookie-session mechanism. If the calendar service could provide an independent web entrance to end users, programmers would have to develop authentication modules based on business requirements - such as user name and password authentication module. If the calendar service is supposed to embed into business systems as a module, SSO (Single Sign On) should be your problem to solve. With respect to the client-server authentication, authentication between servers is a kind of authentication with a high level of trust. Servers are often deployed within the same LAN, or are limited to certain ip sections. An agreement between servers is required, so you can encrypt urls and http heads through one-way encryption algorithm.

Authorization of standard calendar is designed for specific calendar read / write permissions. However business systems usually have role-based authorization. Mapping data of roles and calendars could be maintained either in business system side or in calendar side[7].

C. *Daily Management*

Calendar management mainly includes three aspects: monitoring of system status, user management, roles and authorizations management. Calendar management includes monitoring and summary statistics, such as the current service version, database version, activated service, the number of registered users, the number of authorizations and other information that could allow us to have an overview of the running calendar service. In calendar service, there are just user management, calendar management exists as part of it. We can add, edit or delete users and their calendars in the user management module. Calendars can be shared among users through the delegation mechanism. The authorization of delegated calendars can be managed by the end users or administrators. If the role-calendar mapping relationship was maintained in calendar server, the management interface is required too.

IV. CONCLUSIONS

Calendar service, as an electronic tool of time management, is widely used in all aspects of our life. With the recent popularity of mobile devices, the calendar reaches a new stage of development, people can easily view and manage their schedules on smartphones any time and anywhere. Taking into account the differences between enterprise calendar and standard calendar, the architecture of enterprise is totally different, especially the data synchronization module. We need to introduce a more flexible and reliable design to deal with the data integration with business systems.

Current CalDAV specification was published in March 2007, and iCalendar standard in September 2009, after years of development, these specifications become increasingly inadequate while facing current customer demands. In recent years, the IETF (Internet Engineering Task Force) has done a lot of work in this area, made a number of amendments to the existing specifications, particularly passed jCal standard to replace iCalendar in May 2014. In enterprise calendar area, safety related specifications and bulk manipulation interfaces are also urgent works.

REFERENCES

[1] Dusseault, L. Ed. HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV) RFC4918, 2007. http://tools.ietf.org/html/rfc4918

[2] Daboo, C., Desruisseaux, B. and Dusseault, L.: Calendaring Extensions to WebDAV (CalDAV), RFC4791, 2007. https://tools.ietf.org/html/rfc4791

[3] Desruisseaux, B. Internet Calendaring and Scheduling Core Object Specification (iCalendar), RFC5545, 2009. https://tools.ietf.org/html/rfc5545

[4] Kewisch, P., Daboo, C., Douglass, M. jCal: The JSON Format for iCalendar. RFC7265, 2014. https://tools.ietf.org/html/rfc7265

[5] Yoshii, H., Nomura, Y. and Taniguchi, H.: A Practical Method for Forecasting the Future Calendar Events of Ambiguous Recurrence, 2012 Seventh International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA), pp. 638–643.

[6] Yoshinari Nomura, Yuya Murata, Hideo Taniguchi, Masakazu Urata, Shinyo Muto. Bring Your Own Calendar: A CalDAV-based Virtual Calendar System. 2013 Eighth International Conference on Broadband, Wireless Computing, Communication and Applications, pp. 551-556.

[7] Lisa Dusseault, Jim Whitehead. Open Calendar Sharing and Scheduling with CalDAV. IEEE Computer Society. Mar.2005. pp. 81-89.