

بهینه سازی عملکرد پروتکل Z39.50 و گسترش قابلیت‌های آن به

وسیله تغییر در معماری

مهدی شاملو^{*}، ناصر مزیننی[†]، هما توکلی[‡]

چکیده

برای بازیابی اطلاعات از مجموعه‌های گوناگون، یک پروتکل ارتباطی مبتنی بر استاندارد غیر اختصاصی با نام Z39.50 وجود دارد. این پروتکل لایه کاربرد، برای جستجو در بانک‌های اطلاعات و محیط‌های کامپیوتری توزیع شده استفاده می‌شود. قطعات اصلی آن مبدا و مقصد نام دارد. مستندات این استاندارد مفصل و از نظر منطقی کامل است اما توسعه دهندگان آن در عمل دچار مشکلاتی می‌شوند. این مشکلات را نمی‌توان با تغییر در پیاده‌سازی نمونه‌ها بر طرف کرد و برای حل آن‌ها باید معماری را تغییر داد. در این مقاله معماری جدیدی پیشنهاد می‌شود که مهمترین تفاوتش با معماری پیشین استاندارد، افزودن قطعه واسط ZVirtualServer است. هنگامی که برخی قابلیت‌ها و امکانات در سرویس دهنده یا سرویس گیرنده پیاده شود، در زمان اجرا مشکلات فراوانی پیش می‌آید. اما ما با پیاده سازی آن‌ها بر روی ZVirtualServer، موفق شدیم این موانع را رفع کنیم. سپس در محیط آزمایشگاه قطعات اصلی پروتکل z39.50 و همچنین ZVirtualServer را پیاده سازی کردیم تا یکایک موارد مشکل زای پیشین، در آن بررسی شود. نتایج بدست آمده در این مرحله، رضایت بخش بود.

کلمات کلیدی

جستجو و بازیابی اطلاعات توزیع شده، کتابخانه دیجیتال، مبدا و مقصد، Z39.50، ISO23950، ZVirtualServer

Improving Z39.50 Protocol and Enhancing its Facilities by Modifying Architecture

Mahdi Shamlo, Nasser Mozayani, Homa Tavakoli
(Iran University of Science and Technology)

Abstract

Z39.50 is a standard communication protocol for retrieving data from different sources of information. This is an application layer protocol which is usually used for searching databases in distributed networks. Although the standard has been overwhelmingly defined, in practice, developers encounter numerous problems. These problems may not be removed by implementation techniques, so some modifications in the architecture is needed.

In this paper, we have proposed a new concept, which we called ZVirtualServer interface, in order to ameliorate the architecture. ZVirtualServer helps us to resolve different implementation constraints in client and server side. We have implemented basic parts of Z39.50 for testing our new interface. Obtained results were promising and the interface is currently implemented in a real world digital library system at IUST.

Keywords

Distributed Information Search and Retrieval, Digital Library, Origin/Target, Z39.50, ISO23950, ZVirtualServer

* دانشجوی کارشناسی، دانشگاه علم و صنعت ایران، دانشکده مهندسی کامپیوتر، Shamlo@iust.ac.ir

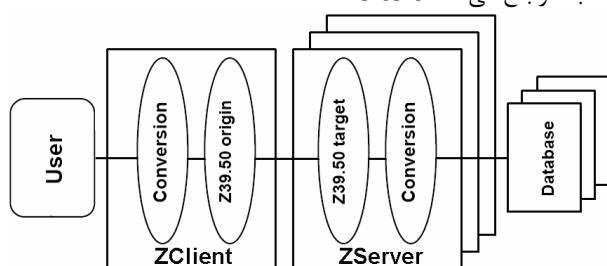
† عضو هیأت علمی دانشگاه، دانشگاه علم و صنعت ایران، دانشکده مهندسی کامپیوتر، Mozayani@iust.ac.ir

‡ دانشجوی کارشناسی، دانشگاه علم و صنعت ایران، دانشکده مهندسی کامپیوتر، Homa_Tavakoli@comp.iust.ac.ir

۱- مقدمه

این پروتکل به ارتباط بین ماشین‌های سرویس گیرنده و سرویس دهنده محدود شده و بر ارتباط بین کاربر انسان^{۱۳} و مبدا یا ارتباط بین مقصد و پایگاه‌های داده اش نظارتی ندارد [۷].

ارتباط تعریف شده در پروتکل، از نوع ارتباط گرا^{۱۴} و حالت دار^{۱۵} است. یعنی مبدا برای ارتباط با مقصد یک جلسه^{۱۶} می‌سازد و تا زمانی که تعامل برقرار باشد، ارتباط حفظ می‌شود. در پیاده سازی، مبدا/مقصد فرم درخواست‌ها و پاسخ‌های محلی خود را به زبان Z39.50 تبدیل می‌کنند. در نتیجه در سمت مبدا، رابط کاربری سازگار برای جستجو در مقصد بدست می‌آید. در سمت دیگر نیز مقصد درخواست‌های رسیده را به زبان پایگاه داده خود تبدیل می‌کند و پس از استخراج و تبدیل مجدد اطلاعات، آن را در قالب پروتکل Z39.50 به مبدا ارجاع می‌دهد [۸] [۵].



شکل (۱): معماری پروتکل Z39.50

از این پس سکوی حاوی مبدا، ZClient و سکوی حاوی مقصد، ZServer نامیده می‌شود.

۴- بررسی نقاط ضعف و کاستی‌های پروتکل Z39.50

در این قسمت به بررسی کاستی‌ها و نقاط ضعف این پروتکل در عرصه عمل می‌پردازیم. البته این موضوع مباحث گسترده‌ای دارد و باید در همه جهات بررسی شود ولی در اینجا، فقط از دیدگاهی خاص به مسئله می‌نگریم و راهکارهای ارائه شده نیز برای حل همین قسمت خواهد بود. موارد زیر مشکلاتی است که با اجرای تغییرات پیشنهاد شده در این مقاله، برطرف خواهند شد.

۴-۱- مشکلات ارتباط با پایگاه‌های فراوان و پراکنده

در معماری پروتکل Z39.50، وظیفه ارتباط با چند پایگاه سرویس دهنده و به تبع آن دانستن آدرس آن‌ها، بر عهده ZClient می‌باشد [۹]. از طرفی در پروتکل Z39.50، ZClient به عنوان ماشین سمت کاربر مشخص شده است. این امر خود عامل بازدارنده‌ای برای ایجاد یک شبه پرتال و فراجستجوگر^{۱۷} در سمت سرویس گیرنده می‌باشد. معماری نسخه سوم پروتکل Z39.50، توانایی دور کردن کاربر را از به خاطر سپردن آدرس ZServerها ندارد. البته بعضاً در پیاده سازی ZClientها سعی می‌شود تا با استفاده از جداول آدرس پیش فرض و راه کارهای دیگر، مشکل مذکور از دید کاربر پنهان

کتابخانه‌های دیجیتالی به ارتباط با کاربران و نیز ارتباط بین یکدیگر نیاز دارند. مهمترین و فراگیرترین پروتکل استاندارد جهت این ارتباط، Z39.50^۱ است. دلیل اصلی نشر پروتکل Z39.50 در آغاز، ایجاد ارتباط بین کتابخانه‌های دیجیتالی با یکدیگر و با کاربران بود، اما به تدریج در حوزه گالری‌های هنر، موزه‌ها، آرشیوها و ... نیز گسترش یافت.

موسسه NISO^۲ با مجوز ANSI^۳ وظیفه تعریف استانداردهای مرتبط با کتابخانه‌های دیجیتالی، اطلاعات و نشر الکترونیکی را بر عهده دارد. این موسسه در سال ۱۹۸۸ نسخه اول پروتکل Z39.50 را تدوین نمود. پس از بازنگری و اصلاح مشکلات، نسخه دوم را در سال ۱۹۹۲ منتشر کرده و با افزودن امکانات بیشتر و تکمیل سایر کاستی‌ها در سال ۲۰۰۳ نسخه سوم آن را ارائه نمود [۱].

۲- پروتکل Z39.50

پروتکل Z39.50، امکان جستجو و بازیابی اطلاعات را بر روی پایگاه‌های دور دست^۴ فراهم می‌سازد. استاندارد Z39.50 یکی از چندین استاندارد NISO است که دو حوزه مکانیزاسیون سیستم‌های سنتی و تکنولوژی‌های نوین را در بر دارد و مدیریت، ذخیره و بازیابی اطلاعات را در آن‌ها استاندارد می‌کند [۱].

قدرت Z39.50 در تفکیک رابط کاربر طرف سرویس گیرنده^۵ از سرویس دهندگان^۶ اطلاعات، موتورهای جستجو و پایگاه‌های داده است که دیدی واحد را برای جستجو و دریافت اطلاعات منبعی که با یکدیگر متفاوتند، ارائه می‌دهد. این امر به توسعه دهندگان طرف سرویس گیرنده توان جمع آوری و ذخیره سازی یکسان داده‌هایی را عرضه می‌کند که از پایگاه‌های مختلف اطلاعاتی استخراج شده اند [۲]. در حقیقت Z39.50، پروتکل ارتباطی است که برای جستجو و بازیابی اطلاعات (اسناد تمام متن^۷، اطلاعات کتابشناختی، تصاویر، چند رسانه^۸ و ...) در محیط‌های توزیع شده، طراحی شده است [۳].

معماری اصلی پروتکل Z39.50 مبتنی بر سرویس گیرنده/سرویس دهنده می‌باشد و از ابزارهای منبع باز^۹ و غیر انحصاری در پیاده سازی حمایت می‌کند [۴] [۵].

۳- بررسی معماری فعلی Z39.50

وظیفه قطعات^{۱۰} اصلی Z39.50، جستجو و بازیابی داده‌هایی است که بر روی چندین سرویس دهنده ذخیره می‌شود. این پروتکل ساختارهای داده و قوانین ارتباطی را به گونه‌ای تعریف می‌کند که ماشین سرویس گیرنده بتواند بانک‌های داده ماشین سرویس دهنده را جستجو کرده و رکوردهای نتیجه را بازیابی نماید. قطعه اصلی استاندارد Z39.50 در طرف سرویس گیرنده، مبدا^{۱۱} و در طرف سرویس دهنده، مقصد^{۱۲} نام دارد [۶].

رکوردهای اولیه در مجموعه نتایج مربوط ترین رکوردها خواهند بود. اگر تعداد نتایج یافت شده بسیار زیاد باشد، به احتمال زیاد کاربر به دریافت رکوردهای ابتدایی بسنده خواهد نمود.

اما اگر کاربر از طریق ZClient درخواست جستجوی خویش را به چندین ZServer بفرستد، مسئله بسیار متفاوت خواهد بود. اگر در حالت، راه حل پیشین انتخاب شود، مشکل جدیدی بوجود می‌آید. در صورتی که هر ZServer به صورت مستقل به مرتب سازی نتایج یافت شده خویش بپردازد و سپس خروجی را به ZClient ارسال کند، ZClient باید مرتب سازی ثانویه ای را به صورت محلی انجام دهد. - این مرتب سازی بر اساس امتیاز محاسبه شده در تابع ارزیابی است. - خصوصاً اگر تعداد رکوردهای درخواست شده از جانب کاربر محدود باشد، باید داده‌های فراوانی به ZClient منتقل شود که حتی ممکن است هیچ گاه به رویت کاربر نرسد. - همانطور که پیش از این مطرح شد انجام این عمل به دلیل لزوم استخراج تمامی نتایج در محل مرتب سازی رکوردها می‌باشد. -

در مدل اصلی پروتکل بنا بر سبک بودن ZClient گذارده شده، بدین معنا که نباید وجود سکویی خاص و قدرتمند را پیش نیاز اجرای آن قرار داد [۱۴]. اما با روش حاضر، برای استحصال هدف فوق، ناچاریم این اصل را نادیده بگیریم.

۴-۴- ذخیره سازی نهان^{۲۰} نتایج

عموماً سرویس گیرندگان کتابخانه‌های دیجیتال، کاربران پرتعدادی هستند که بعضاً حوزه جستجوی آنها با یکدیگر اشتراک دارد. این عامل توجیه خوبی برای انجام عملیات ذخیره سازی نهان بر روی ZServer و ZClient است که قصد داریم در اینجا به بررسی آن بپردازیم.

روش اول، ذخیره سازی مجموعه نتایج استخراج شده جستجو به صورت تمام و کمال است. در این وضعیت اگر کاربر دیگری عبارت کاربر پیشین را مجدداً جستجو نماید، سیستم اقدام به تحویل مجموعه نتایجی که به صورت نهان ذخیره شده، می‌کند تا مجدداً به پایگاه داده رجوع ننماید (در سمت سرویس گیرنده). روش دوم، ذخیره سازی رکوردها به صورت مستقل است. بدین صورت که اگر در مجموعه نتایج جستجو، رکوردهای مشابه با جستجوهای قبلی مشاهده شود، سیستم آنها را از ذخیره‌گاه نهان خوانده تا لزومی برای مراجعه به پایگاه اصلی نباشد (در سمت سرویس دهنده) [۱۵]. البته روش‌های دیگری هم وجود دارد که در اینجا لزومی به تشریح آنها نیست. اما باید جایگاه سرویس ذخیره سازی نهان پروتکل را معین کرد.

اگر این سرویس در ZClient قرار داده شود ظاهراً به جستجوهای کاربر سرعت می‌بخشد، اما بنای استاندارد بر سبک بودن بستر سخت افزار و شبکه ZClient است و این قید، عامل بازدارنده‌ای برای داشتن یک ذخیره‌گاه نهان با راندمان^{۲۱} بالا است.

اگر این سرویس در ZServer نیز قرار گیرد، مفید است اما باعث می‌شود تا داده‌های ذخیره شده به صورت نهان، منحصر به هر

گردد [۱۰] اما با این وجود نیز همچنان پروتکل Z39.50 در این حوزه دچار ضعف است.

۴-۲- مشکلات جستجوی به صورت همروند

اگر کاربران بخواهند بر روی تعداد زیادی از ZServer ها جستجو نمایند، باید به هر یک متصل شوند و نتایج را دریافت کنند. همانطور که در مرحله پیشین به آن اشاره شد، این امر مستلزم صرف زمان فراوان برای دریافت نتایج است.

راه حل پیشنهادی نسخه سوم پروتکل، پیاده سازی قطعه ZClient به صورت همروند می‌باشد [۱۱]. البته این عبارت به صراحت در مستندات پروتکل وجود ندارد، ولی در نسخه سوم با افزودن برخی امکانات، شرایط برای پیاده سازی آن فراهم شده است [۱۲]. ایده مطرح شده برای حل این مشکل بسیار مفید است اما معایبی را در پی دارد که در عمل قطعات پیاده شده را غیر کاربردی می‌کند.

- اول این که سرویس گیرنده‌های Z3950 بسیار متنوع بوده و در بسیاری از محصولات به صورت درونی^{۱۸} پیاده شده اند. از این رو امکان تغییر در آنها براحتی میسر نیست.
 - دوم این که همروندی احتیاج به ایجاد ارتباطات موازی دارد. بسیاری از کاربران استفاده کننده از ZClient، بستر چنین ارتباطی را ندارند. بدین معنی که زیرساخت شبکه فقط اجازه برقراری تعداد محدودی ارتباط را به آنها در هر لحظه می‌دهد.
 - سوم کیفیت سکوی سخت افزاری و نرم افزاری است. ZClient برای اتصال و تعامل هم زمان با تعداد زیادی ZServer، به سکویی قوی نیاز دارد.
- با وجود دلایل فوق، پیاده سازی همروندی در ZClient، برای تعداد زیاد کاربران، چندان مناسب و کارآمد نخواهد بود.

۴-۳- ضعف در عملیات رتبه بندی نتایج

جستجوی رتبه‌دار به معنای مرتب سازی نتایج یافت شده در یک جستجو بر اساس رتبه است. هر رکورد استخراج شده در این نتیجه، بسته به عبارات مورد جستجوی کاربر، یک امتیاز دارد. هدف این است که به نحوی امتیاز را به رکوردی اختصاص دهیم که امتیاز بالاتر نمایانگر رکورد مطلوب تر برای کاربر جستجوگر باشد.

برای داشتن امکان جستجوی رتبه‌دار بین نتایج بدست آمده، باید دو شرط مهم را در نظر داشت. اول تعریف یک تابع ارزیابی^{۱۹} برای رکوردها است که در این جا به جزئیات آن اشاره ای نمی‌کنیم. اما شرط دوم، لزوم استخراج تمامی نتایج از پایگاه می‌باشد تا بتوان آنها را بر اساس خروجی تابع ارزیابی مرتب نمود [۱۳].

اگر بخواهیم جستجوی امتیازدار را بر روی یک ZServer پیاده کنیم، مدل مسئله واضح و بدون ایراد است. در هر جستجو ماشین Zserver، نتایج یافت شده را مرتب می‌کند. این مرتب سازی بر اساس خروجی تابع ارزیابی است که بر روی هر رکورد اعمال می‌شود.

کاربران مجاز را نیز باید در اختیار داشته باشد. در نتیجه اگر کاربری خواهد در چندین ZServer جستجو کند، یکپارچگی سیستم از بین خواهد رفت. اجرای متمرکز احراز هویت، حلقه مفقوده مسئله می‌باشد که در پروتکل دیده نشده است.

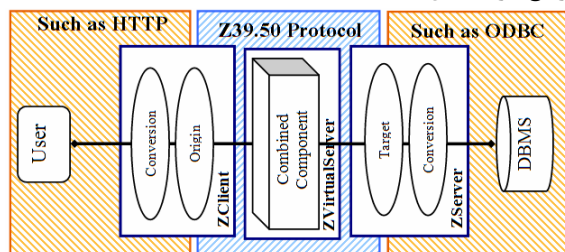
۵- پیشنهاد قطعه ZVirtualServer

موارد ذکر شده در بخش قبل، مشکلاتی است که در زمان اجرا رخ می‌دهد. سعی ما در این مقاله، پیشنهاد راهکار اصولی و قابل اجرا برای غلبه بر آن‌ها است. در صورت رفع مشکلات فوق می‌توان انتظار داشت پروتکل Z39.50 بیش از پیش فراگیر شده و رضایت استفاده کنندگان آن نیز افزایش یابد.

اگر یک سیستم نرم افزاری فراگیر دارای ایراد باشد، ابتدا سعی می‌شود تا با ایجاد تغییراتی در سطح پیاده سازی، مشکل را رفع کرد. در صورت حل نشدن مشکل، اقدام به رفع آن در سطح معماری می‌گردد. این اقدام به معنای ایجاد تغییراتی در ساختار معماری سیستم و نحوه ارتباط بین قطعات است. در نهایت اگر این روش نیز نتیجه ای در بر نداشت، به اجبار باید استانداردها و پروتکل‌های استفاده شده در سیستم را تغییر داد [۱۸].

در حل مسئله موجود نیز ترتیب بالا رعایت شده است. همانطور که در بخش قبل اشاره شد، مشکلات و نقاط ضعف نسخه سوم Z39.50 را نمی‌توان با پیاده سازی مجدد قطعات آن حل کرد. چرا که بسیاری از آن‌ها متأثر از معماری و نحوه ارتباط بین قطعات است. پس از تحقیق و بررسی فراوان، این تیم به ایده جدیدی رسیده که راه حل خوبی برای موارد فوق الذکر می‌باشد. پیشنهاد ما اعمال تغییرات در معماری و نحوه ارتباط قطعات پروتکل Z39.50 است. در این روش سعی شده تا در استاندارد ارتباطی Z39.50 هیچ گونه تغییری رخ ندهد. در نظر گرفتن این مورد سبب ایجاد سازگاری و مطابقت^{۲۸} با ابزارهای پیشین نسخه سوم خواهد شد.

مهمترین نکته قابل توجه در پیشنهاد ما، افزودن قطعه‌ای با نام ZVirtualServer در معماری استاندارد است. دلیل وجودی ZVirtualServer، ایجاد یک واسط برای دسترسی کاربران به ZServer ها است تا بتوان مشکلات مطرح شده در نسخه سوم پروتکل را مرتفع کرد. (شکل ۲)



شکل (۲): نحوه ارتباطات خارجی ZVirtualServer در معماری جدید استاندارد Z39.50

ZServer باشد که این مورد نیز عامل بازدارنده ای برای به دست آوردن یک ذخیره‌گاه نهان با راندمان بالا از دید کاربر خواهد بود.

۴-۵- مشکلات تقسیم و توازن بار ZServer

پروتکل Z39.50 از دیدگاه طراحی نرم افزار، نسبتاً کامل و جامع است اما در زمان اجرا و پیاده سازی نمونه‌های عملی و بزرگ، مشکلی به چشم می‌خورد که شاید در ظاهر به خود پروتکل مربوط نباشد اما جزء تبعات جانبی معماری آن است.

سرویس دهنده‌هایی که به صورت هم زمان به تعداد زیادی از کاربران سرویس می‌دهند، باید از زیر ساخت سخت افزاری قوی برخوردار باشند [۱۵]. برای چنین زیر ساختی به بیش از یک ماشین احتیاج است. در فضای کتابخانه دیجیتال نیز این شرایط برقرار است. برخی از ZServer ها مراجعین فراوانی دارند و در نتیجه باید قدرت پردازش و استخراج اطلاعات فوق العاده بالایی داشته باشند. در این وضعیت، پیاده سازی قطعات ZServer بر روی یک ماشین غیر ممکن است و باید به دنبال راه حلی بود تا بتوان بار ZServer را بر روی چند ماشین تقسیم کرد.

پروتکل Z39.50 در این حوزه وارد نشده و در صورت نیاز باید اجرای این سرویس را به لایه‌های پایین تر محول نمود.

تقسیم و توازن بار در لایه های پایین تر شبکه، به طور کامل مورد بررسی قرار گرفته و ابزارها و استانداردهای بسیاری برای آن موجود است [۱۶]. اما تنها به این نکته بسنده می‌کنیم که در صورت انتخاب این روش، بسیار سخت می‌توان مسیری یافت تا پروتکل لایه بالاتر یعنی پروتکل Z39.50 بدون تغییر باقی بماند.

راه حل بهتر، پیاده سازی این سرویس در لایه کاربرد^{۲۳} یا همان پروتکل Z39.50 است. در این وضعیت باید از طرفی پروتکل Z39.50 را دست نخورده حفظ کرد و از طرف دیگر تعدد ماشین‌های ZServer را از دید کاربر پنهان نمود. با توجه به معماری استاندارد تعیین شده در نسخه سوم پروتکل، پیاده سازی این راه حل مقدور نخواهد بود.

۴-۶- مشکلات برقرای امنیت ZServer ها

Z39.50 پروتکلی استاندارد برای انتقال فراداده^{۲۴}، مشخصات کتابشناختی، محتوای دیجیتال و ... است که در بعضی کاربردها رعایت فاکتورهای مهم امنیت یعنی احراز هویت^{۲۵}، یکپارچگی^{۲۶} و محرمانگی^{۲۷} ضروری به نظر می‌رسد [۱۷].

از دیدگاه امنیتی، پروتکل Z39.50 به صورت ساده، به احراز هویت اشاره نموده و اجرای سایر موارد را به لایه‌های زیرین سپرده است [۱۲]. در عمل اگر تصمیم بر اجرای ملاحظات امنیتی باشد، باید ZServer را با ابزارهای ایمن سازی تجهیز کرد. دو نکته در اینجا حائز اهمیت است. اول این که نمی‌توان تمامی امکانات را بر روی بسترهای سخت افزاری متوسط و ضعیف فراهم کرد و دوم، هر ZServer موظف است به صورت مستقل احراز هویت را انجام دهد که بالطبع، لیست

۶- نتایج حاصل از معماری جدید پیشنهادی

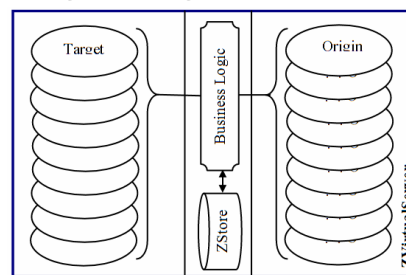
۶-۵- افزایش راندمان ذخیره سازی نهان نتایج

یکی از موارد مهم دیگر، قابلیت ذخیره سازی نهان نتایج در این قطعه است. حوزه جستجوی کاربران بعضاً با هم اشتراک دارد و به تبع آن، بسیاری از رکوردهای نتایج یافت شده، با یکدیگر مشابهند. در سیستم جدید، رکوردهای برگشتی از ZServer ها، در یک موجودیت موقتی^{۲۹} ذخیره می‌شود تا اگر در مجموعه درخواست‌های بعدی کاربران قرار داشت، احتیاج به استخراج مجدد آن‌ها نباشد. این قابلیت باعث افزایش سرعت جستجو و بازیابی نتایج می‌شود.

۶-۱- نامحسوس بودن تفاوت بین ZVirtualServer

و قطعات پیشین پروتکل

قطعه جدید به گونه ای طراحی شده تا از دید ZClient، مانند ZServer و از دید ZServer مانند ZClient باشد. در واقع واسط ارتباط با ZClient این قطعه از جنس مقصد و واسط ارتباط با ZServer آن از جنس مبدا است. این ملاحظه باعث سازگاری سیستم جدید با سایر سرویس دهندگان قدیمی Z39.50 می‌گردد. شکل (۳)



شکل ۳: اجزاء داخلی ZVirtualServer

۶-۶- تقسیم و توازن بار در ZServerها

قطعه ZVirtualServer می‌تواند درخواست‌های رسیده از جانب کاربران را به صورت متوازن، بر روی چند ZServer تقسیم کند. نکته مهم این روش، پنهان بودن این عمل از دید ZClient است. همانطور که پیش از این ذکر شد، برای پیاده سازی ZServerها در مقیاس بسیار بزرگ، به تقسیم و توازن بار بر روی چندین ماشین نیاز است و بزرگترین مشکل در این راه، ناتوانی در پنهان کردن تعداد ماشین‌ها می‌باشد که با پیاده سازی ZVirtualServer مرتفع خواهد شد.

۶-۲- بهبود قابلیت فرآجستجو

قطعه جدید می‌تواند درخواست‌هایی که از ZClient می‌رسند را پس از تحلیل و بررسی به چندین ZServer ارسال کند و پس از جمع آوری همه نتایج، به صورت مدیریت شده آن‌ها را به ZClient برگرداند. این قابلیت باعث می‌شود تا کاربر ZClient، مشکلی در ایجاد ارتباط با چندین ZServer نداشته باشد. در این حالت کاربر، حتی نیازی به دانستن آدرس و مشخصات ZServerهای اصلی نخواهد داشت.

۶-۷- مدیریت یکپارچه و تسهیل در تهیه آمار

در طراحی جدید، قابلیت مجتمع سازی درخواست‌های کاربران بر روی قطعه ZVirtualServer در نظر گرفته شده است. بدین معنا که می‌توان به طور متمرکز به مدیریت سرویس دهی و تهیه آمارهای مختلف پرداخت. عملیات آمار گیری زمانی ارزشمند و قابل قبول است که حوزه تحت پوشش بزرگ و جامع باشد. کاربران ZVirtualServer به مراتب بیشتر و پرتعدادتر از یک ZServer مستقل خواهند بود.

۶-۳- بهینه سازی جستجوی همزمان چندین پایگاه

قطعه جدید قابلیت جستجوی همزمان (و همروند) را بر روی تعداد زیادی ZServer دارد. این قابلیت باعث کوتاه شدن زمان جستجو برای کاربر ZClient می‌شود. البته برای پیاده کردن این امکان خصوصاً در مقیاس بزرگ به بستر قوی سخت افزار و شبکه احتیاج است. اما با توجه به جایگاه ZVirtualServer (که در گروه ماشین‌های سرویس دهنده قرار دارد)، فراهم کردن این بستر دور از دسترس نخواهد بود.

۷- آزمایش و مقایسه

این تیم در سال ۸۴ موفق شد برای اولین بار در ایران، ZServer و ZClient نسخه سوم استاندارد Z39.50 را در محیط آزمایشگاهی پیاده سازی کند^{۳۰}. سپس با ادامه مسیر تحقیق و پژوهش، طرح ZVirtualServer را مطرح نمود و موفق به اجرای آن گردید.

پس از پیاده سازی قطعه ZVirtualServer، سعی شد تا میزان وصول به فاکتورهای مطرح شده در بخش قبل، ارزیابی گردد که گزارش آن به شرح زیر است:

جدول (۱): میزان حصول نتایج

مورد	میزان حصول	شرح
۱-۶	به صورت کامل	قطعه ZVirtualServer، قابلیت جستجو و بازیابی اطلاعات را از روی ZServerهای استاندارد دارد. قطعه ZVirtualServer، قابلیت سرویس دهی به ZClientهای معتبر را دارد.
۲-۶ و ۳-۶	به صورت کامل	قطعه ZVirtualServer، قابلیت جستجو همزمان بر روی چندین ZServer و جمع آوری نتایج آن‌ها را دارد تا به صورت مجتمع شده در اختیار کاربر قرار دهد.

IFLA General Conference and Council, Buenos Aires, Argentina, August 2004

[11] Hakala Juha, Husby Ole, Schomacker Tommy, Dalgaard-Muller Bodil, "Scandinavian Virtual Union Catalogue Project Final report", Helsinki University Library, The National Library of Finland, January 2004

[12] ZIG Group, Library of Congress Z39.50 Tutorials, Library of Congress Z39.50 Agency, 2000, <http://www.loc.gov/z3950/agency/resources/tutorials.html>

[13] Buckley Chris, Ryall Peter, Z39.50 & Ranked Searching, Library of Congress Tutorials, 2000, <http://www.loc.gov/z3950/agency/indexing/peter.html>

[14] Willian J, "Ongoing March Toward Digital Government, Computer", Vol. 34, p 38, IEEE Computer Society, 2001

[15] Wolfgang Adamczak, Annemarie Nase, "Gaining Insight from Research Information", 6th CRIS2002, Poland, 2002

[16] Krasimir Trichkov, "Distributed Searching in Web Information Systems", CompSysTech, Bulgaria, 2005

[17] Benoit Guillon, Dominique Burger, Bruno Marmol, "A Secure Internet Service for Delivering Documents", 8th ICCHP 2002, Linz, Austria, Proceedings Springer Berlin, Heidelberg, July 2002.

[18] Sajan, Mathew, Software Engineering, New Delhi, S. Chand & Company Ltd., 2003.

به صورت کامل در محیط آزمایشگاهی	۴-۶	این قابلیت به طور کامل در محیط آزمایشگاه به انجام رسیده، اما به دلیل استاندارد و فراگیر نبودن جستجوی رتبه‌دار در ZServer های معتبر، امکان اجرای آن در محیط واقعی فراهم نشده است.
در دست پیاده سازی	۵-۶	پیاده سازی این امکان، محتاج بررسی و تحقیق مفصل در این رابطه است. اما از نظر منطقی، کاملاً قابل اجرا می‌باشد.
به صورت ابتدایی در محیط آزمایشگاهی	۶-۶	پیاده سازی کامل این امکان نیز محتاج بررسی و تحقیق مفصل در این رابطه می‌باشد. اما به طور محدود، در محیط آزمایشگاه، بر روی دو ZServer اجرا شد.
به صورت ابتدایی در محیط آزمایشگاهی	۷-۶	لازمه تهیه و جمع آوری آمار واقعی، سرویس دهی در محیط واقعی است. اما از نظر منطقی، کاملاً قابل اجرا می‌باشد. در آزمایشگاه به صورت ساده تست شده است.

۸- نتیجه

در این مقاله، نقاط ضعف و کاستی های پروتکل و استاندارد ارتباطی Z39.50 بررسی شد. برای حل آن ها، اقدام به ایجاد تغییراتی در معماری استاندارد گردید که مهمترین آن افزودن قطعه ZVirtualServer به عنوان واسط بود. با افزودن این قطعه قابلیت ارتباط با Zserver های پراکنده، فرا جستجو، جستجوی همزمان، جستجوی رتبه‌دار، ذخیره‌سازی نهان، توازن بار و مدیریت متمرکز، بدست آمد. در صورتی که پیش از این، یا پیاده سازی آنها مقدور نبود و یا در اجرا، هزینه بسیار زیادی نیاز داشت. سپس پیشنهاد ارائه شده در محیط آزمایشگاه پیاده سازی شد که در مقایسه با نسخه اصلی بسیار کارتر و قوی تر عمل نمود.

مراجع

[1] Juha Hakala, "Z39.50: Application Service Definition and Protocol Specification", NISO Press, Maryland, 2003

[2] Moen Bill, Z39.50 and Metadata, Library of Congress Tutorials, University of North Texas, 2000, <http://www.loc.gov/z3950/agency/indexing/moen.html>

[3] Turner Fay, "THE Z39.50 INFORMATION RETRIEVAL STANDARD", International Federation of Library Associations Institution, National Library of Canada, 1997

[4] ZIG Group, Z39.50 Attribute Architecture, Ver 1.1, 1999, <http://www.loc.gov/z3950/agency/attrarch/arch.html>

[5] Lynch C, "Using the Z39.50 Information Retrieval Protocol", RFC1729, 1994 - RFC Editor

[6] Trichkov K, Tsenov M, Trichkova E, "Z39.50", Proceedings of the 5th international conference on Computer Communications and Network, Chicago, 2004

[7] Cousins Shirley, Sanders Ashley, Establishment of an M25 Z39.50 target and testing, September 2003, <http://ccinterop.cdfr.strath.ac.uk/documents/biannual%20progressreport%20feb2004.pdf>

[8] Trichkov Krasimir, "Standardization Approach for Information Retrieval in WAN", CompSysTech, Portuguese, 2004

[9] Dempsey, Lorcan, "The Recombinant Library: Portals and People", Journal of Library Administration, 2003

[10] Cristina Magliano, "Holding data in the SBN Union catalog", World Library and Information Congress: 70th

زیر نویس ها

- ¹ ISO23950
- ² National Information Standards Organization
- ³ American National Standard Institute
- ⁴ Remote
- ⁵ Client
- ⁶ Server
- ⁷ Full-text Data
- ⁸ Multimedia Data
- ⁹ Open Source
- ¹⁰ Component
- ¹¹ Origin
- ¹² Target
- ¹³ Human User
- ¹⁴ Connection Oriented
- ¹⁵ Stateful
- ¹⁶ Session
- ¹⁷ Meta Searcher
- ¹⁸ Embedded
- ¹⁹ Evaluation Function
- ²⁰ Caching
- ²¹ Cache Hit
- ²² Distributing and Load Balancing
- ²³ Application Layer
- ²⁴ Metadata
- ²⁵ Authentication
- ²⁶ Integrity
- ²⁷ Privacy
- ²⁸ Compatibility
- ²⁹ Temporal Entity

³⁰ این طرح به عنوان یک بخش از سیستم یکپارچه کتابخانه دیجیتال در هفتمین دوره جشنواره

خوارزمی برگزیده گردید