اصول پایه سرویس گرائی

امیر رضا مهجوریان، فریدون شمس

mahjoorian@esoa.ir, f_shams@sbu.ac.ir

مقدمه:

"معماری سرویس گرا[1]" مفهومی جدید نیست و از دهه ۹۰ وجود داشته است ولی آنچه جدید است توانائی اجرا و عینیت بخشیدن به آن است که به کمک ابزارها و پروتکل های مربوطه میسر شده است. معماری سرویس گرا از دیدگاه های مختلف قابل بررسی است، هر فرد یا ذینفع بر طبق جایگاه خود تصویری از معماری سرویس گرا دارد، در ادامه از سه دیدگاه کارشناسان حرفه، معماران و طراحان معماری سرویس گرا مورد بررسی قرار می گیرد[۱و۳].

کارشناسان حرفه: مجموعه ای از سرویس ها که سازمان مایل به ارائه آنها به مشتریان یا شرکاء خود است. (تعریف سرویس کسب و کار)

معماران[2]: سبکی از معماری که حاوی قوانین، الگوها و ضوابطی است که منجر به ایجاد خصایصی نظیر پیمانه ای بودن[3]، بسته بندی[4]، اتصال سست[5]، استفاده مجدد[6] و ترکیب پذیری[7] شده و از نظر ساختار از یک ارائه دهنده سرویس و یک درخواست کننده سرویس تشکیل شده است.

طراحان و پیاده سازان: یک سبک(مدل) برنامه نویسی که از استانداردهائی مانند (SOAP[8]،UDDI[9]،WSDL[10]، ..) و فناوری هائی نظیر سرویس های وب استفاده می کند و قابلیت تعامل پذیری بین مولفه های نرم افزاری را بدون توجه به سکو و فناوری پیاده سازی آنها پشتیبانی می کند.

---

[1] Service Oriented Architecture
[2] Architects
[3] Modularity
[4] Encapsulating
[5] Loosely Coupling
[6] Reusability
[7] Composability
[8] Simple Object Access Protocol
[9] Universal Description Discovery and Integration
[10] Web Service Definition Language

تعریف معماری سرویس گار:

برای معماری سرویس گرا تعاریف متنوع و بعضا مختلفی ارائه شده که هر کدام از نگاهی به تبین خصوصیات آن پرداخته اند، برای درک بهتر این مفهوم و آگاهی از کلیه برداشت ها و نگاه های موجود، در ادامه تعدادی از این تعاریف آورده شده است.

یک چارچوب استراتژیک از فناوری که به تمام سیستم های داخل و خارج اجازه ارائه یا دریافت سرویس های خوش تعریف را می دهد [۴].

روشی برای طراحی و پیاده سازی نرم افزارهای گسترده سازمانی به وسیله ارتباط بین سرویس هائی که دارای خواص اتصال سست، دانه درشتی و قابل استفاده مجدد هستند [۵].

سبکی از معماری که از اتصال سست سرویس ها جهت انعطاف پذیری و تعامل پذیری حرفه و بصورت مستقل از فناوری پشتیبانی می کند و از ترکیب مجموعه ای از سرویس های مبتنی بر حرفه تشکیل شده که این سرویس ها انعطاف پذیری و پیکربندی پویا را برای فرآیندها محقق می کنند [۶].

ضرورت و فواید معماری سرویس گرا

فواید معماری سرویس گرا از نگاه اشخاص مختلف موضوعی است که باید به آن توجه داشت چرا که هر فرد بسته به جایگاه خود و نوع وظیفه ای که دارد از دیدگاه خاص خود به معماری سرویس گرا می نگرد. در ادامه به بررسی تاثیرات معماری سرویس گرا از دیدگاه اشخاص مختلف در سازمان می پردازیم [۲و۸]

مدیر ارشد اجرائی(CEO[11]): محصولات یا فرآیندهای جدید به سادگی توسط فناوری اطلاعات اجرا خواهند شد. سیستم های انعطاف پذیر دیگر مانعی بر سر تغییر و تکامل سریع فرآیندها نخواهند بود.

مدیر ارشد اطلاعاتی(CIO[12]): رفع معضل بزرگ یکپارچگی مجموعه سیستم های سازمان، یکی از بزرگترین مشکلاتی که فناوری اطلاعات در سازمانها بزرگ با آن روبرو بوده است. بدین ترتیب پاسخگوئی به نیازهای حرفه نیز بهبود میابد.

مدیر پروژه های تولید و توسعه سیستم های اطلاعاتی : تقسیم پروژه ها به اجزاء کوچکتر که می توانند مستقل از هم انجام شوند به سادگی محقق می شود. همچنین کنترل پیشرفت هر زیر پروژه نیز مستقلا قابل محاسبه و کنترل خواهد بود.

---

[11] Chief Executive Officer
[12] Chief Information Officer

توسعه دهندگان سیستم: در گذشته یکی از سخت ترین و طاقت فرساترین کارها برای تولید کنندگان سیستم های اطلاعاتی انجام یکپارچگی و اتصال با دیگر سیستمها بوده درحالیکه اکنون وب سرویس رهیافت غالب برای تولید سرویس هائی مستقل از فناوری است که توسط دیگر سرویس های تحت وب قابل فراخوانی هستند.

کاربران سیستم ها: سیستم ها به سادگی نیاز کاربران را برآورده می کنند، مشکل انتقال اطلاعات بین سیستم ها به کمک یکپارچگی حل شده است و پیچیدگی های فناوری برای کاربران مخفی است. دیگر لازم نیست کاربران اطلاعات را با رسانه های ذخیره سازی از یک سامانه به سامانه دیگر انتقال دهند.

ارکستریشن[۱۳]

دو واژه پر کاربرد در حوزه کسب و کار و معماری سرویس گرا که معمولا به جای هم اشتباه گرفته می شوند، ارکستریشن و کاریگرافی[۱۴] نام دارند. ارکستریشن در خصوص ترتیب اجرای سرویس ها در فرآیند بحث می کند، ارکستر اصلی مجموعه ای از سرویس ها را فراخوانی می کند تا نتیجه مورد نظر حاصل شود و فرآیند تکمیل گردد، ممکن است سرویس های خارج سازمان نیز در این راستا فراخوانی و استفاده شوند، این کار با کمک موتور فرآیند محقق می شود. در عوض کاریگرافی به فرآیندهایی گویند که بدون موتور فرآیندی(رهبر ارکستر) اقدام به تبادل پیام کرده و ترتیب و توالی پیامهای مبادلاتی را خود بازیگران ثبت و کنترل می کنند.

بنابراین ارکستریشن به معنای وجود یک موتور فرآیندی است که ترتیب و توالی را کنترل کرده و از شرکاء داخلی یا خارجی برای انجام کارها استفاده می نماید. نمونه این مدل سیستم مدیریت فرآیندهای حرفه(BPMS[15]) است که فرآیندها در موتور فرآیندی اجرا می شوند.

کاریگرافی به معنای پردازش های توزیع شده بین چند فرآیند است که بدون یک رهبر مرکزی با هم تعامل دارند یا چندین موتور فرآیندی که در کنار و هم سطح هم اجرا می شوند و با همکاری هم هدفی را محقق می سازند. نمونه این موضوع در پردازش های توزیع شده و یا فعالیت های بین سازمانی که هر دو طرف با مشارکت هم به دنبال یک هدف هستند دیده می شود[۷].

---

[13] Orchestration
[14] Choreography
[15] Business Process Management Systems

اصول معماری سرویس گرا

سرویس گرایی ریشه در معماری نرم افزار دارد و منشعب از رهیافت تقسیم و غلبه[۱۶] است، یک مساله به اجزاء کوچکتر و خوش تعریفی شکسته می شود و هر جزء یک مساله کوچکتر و با پیچیدگی کمتر خواهد بود. بدین ترتیب یک موجودیت پیچیده به اجزاء کوچک و قابل حل تقسیم می شود که با انجام کلیه این اجزاء، مساله حل می شود. این تئوری در گذشته نیز بصورت های مختلف پیاده سازی شده بود، برنامه نویسی شیء گراء[۱۷] و توسعه مبتنی بر مولفه[۱۸] دو نمونه از این پیاده سازی ها هستند، سرویس گرایی حالت ممتاز و عالی پیاده سازی برای این شیوه است.

در ادامه تعدادی از اصول پایه برای معماری سرویس گرا معرفی می شوند:

قابلیت استفاده مجدد:

اساسا سرویس ها برای استفاده مجدد طراحی می شوند، خواه این استفاده مجدد در زمان حال انجام شود و یا موکول به آینده شود، این امر با بکارگیری استانداردهای مستقل از فناوری و جداسازی پیاده سازی سرویس از واسط محقق می شود. قابلیت استفاده مجدد شامل انواع مختلفی چون بین چند نرم افزار، سرویس های ارکستریشنی و سرویس های بین سازمانی می شود. انواع مختلفی از سرویس وجود دارد، بعضی از این سرویس ها ارکستریشن دیگر سرویس ها هستند و دارای فراخوانی های زیاد هستند، گروه دیگری مانند سرویس های پایه(اولیه) خود مختاری کامل دارند و سرویس دیگری را فراخوانی نمی کنند.

قرارداد رسمی برای تعامل

سرویس هایی که نیاز به تعامل(استفاده) با هم دارند می بایست قواعد مربوط به چگونگی این ارتباط را بصورت رسمی و مشخص تعریف و منتشر نمایند. این قرارداد حداقل شامل تعریف نام و آدرس سرویس، عملیات ها، پیام های ورودی و خروجی برای هر عملیات و نوع این داده ها است. از آنجاکه تنها طریقه شناخت سرویس گیرندگان از یک سرویس در این قرارداد تعریف می شود، لذا سرویس دهنده باید به مواردی چون تعریف کامل و دقیق اطلاعات، مدیریت نسخه ها و نگهداری و انتشار آن توجه نماید.

اتصال سست سرویس ها

اصولا یکی از اصول اصلی این معماری بر پایه اتصال سست سرویس ها است، بطوریکه آنها به یکدیگر وابستگی شدید نداشته باشند و بیشترین استقلال از یکدیگر را داشته باشند. این به معنای آن است که یک

---

[16] Divide and Conquer
[17] Object Oriented Programming
[18] Component Based Development

سرویس ممکن است نیاز به اطلاعات دیگر سرویس ها و استفاده از آنها داشته باشد ولی مستقل از آنها(چگونگی پیاده سازی) اجرا می شود.

پنهان سازی پیاده سازی داخلی

تنها بخشی از سرویس که برای شرکاء و محیط قابل روئیت است، واسط سرویس است که تحت قالب یک قرارداد منتشر می شود. چگونگی پیاده سازی عملیات از دید محیط مخفی است و حتی ممکن است بدون اطلاع سرویس گیرندگان تغییر کند یا از فناوری جدیدی استفاده شود بدون انکه واسط و تعریف سرویس تغییر کند. در حقیقت انچه محیط اطراف از یک سرویس می داند "چه چیز" است و نه "چگونه"، بدین ترتیب سرویس ها بصورت جعبه سیاه می شوند که جزییات داخلی خود را مخفی می نمایند.

قابلیت ترکیب پذیری

سرویس ها از یکدیگر استفاده می کنند، بدین ترتیب دانه بندی های متفاوتی از سرویس ها ایجاد می شود و قابلیت استفاده مجدد ارتقاء می پذیرد. ارکستریشن نیز به نوعی براین اصل استوار است و از آن سود می جوید، این مورد شباهت هایی با قابلیت استفاده مجدد دارد با این تفاوت که قابلیت استفاده مجدد بر مهیا نمودن لوازم و استانداردها تاکید دارد در حالیکه ترکیب پذیری مربوط به چگونگی تعیین و طراحی سرویس ها از نظر دانه بندی و کارکردی است بگونه ای که بتوان یک سرویس کلان را از ترکیب دیگر سرویس ها ایجاد نمود.

خود مختاری سرویس ها

هر سرویس برای خود دارای منطق کاری و داده هایی است که حوزه عملکرد آن را تعیین می کند، این حوزه کاملا مشخص است. اگرچه سرویس ها از همدیگر استفاده می کنند (در غیر اینصورت اصلا قابلیت استفاده مجدد معنا نخواهد داشت) ولی به همدیگر وابستگی محکم نداشته و هر کدام منطق و کارکرد خاص خود را دارند. این اصل بر این موضوع تاکید دارد که یک سرویس باید بتواند چگونگی پیاده سازی داخلی خود را تغییر یا گسترش بدهد بدون اینکه برای این کار نیاز به اجازه یا تغییری در دیگر سرویس ها باشد. یک از مزایای معماری سرویس گرا قابلیت انعطاف پذیری آن است بگونه ای که یک سرویس می تواند مراحل پیاده سازی خود را تغییر داده و یا از ارائه دهنده های جدیدی برای فراخوانی سرویس های مورد نیازش استفاده کند بدون آنکه این موضوع نیاز به کسب اجازه یا تغییری در محیط یا شرکاء داشته باشد.

بی وضعیت(حالت) سرویس ها

سرویس ها نباید حالت و وضعیت جاری خاصی را ثبت کنند زیرا اصولا یک واحد قابل استفاده مجدد هستند که توسط شرکاء مختلفی استفاده شده و یک عملیات مشخص را انجام می دهند. کنترل و ثبت حالت و توالی انجام کارها در جای دیگری(مثلا ارکستر مرکزی) صورت می گیرد.

اگر پیاده سازی سرویس بگونه ای باشد که وضعیت فعلی پاسخ گویی به یک درخواست خاص را در خود ذخیره کرده و به نوعی وابسته به آن درخواست شود، آنگاه امکان پاسخ گویی و دسترس پذیری آن سرویس برای دیگر متقاضان پایین آمده و حتی صفاتی چون خود مختاری نیز مخدوش می شود. از طرف دیگر عملیات سرویس بگونه ای است که قابل مرحله بندی نبوده و یک تراکنشی است بدین صورت که با دریافت درخواست توسط سرویس دهنده، پاسخ مناسب به سرویس گیرنده ارسال می شود. ارائه کلیه کارکردهای یک سرویس در قالب چندین عملیات "مستقل از هم و تک تراکنشی" از ضرورت های اصل بی وضعیتی است.

قابلیت شناسایی و کشف

قرارداد واسط استفاده از سرویس ها بایست توسط شرکاء و کلیه عواملی که مجاز به استفاده هستند، قابل شناسایی و کشف باشد. از مزایای معماری سرویس گرا قابلیت جستجو بین ارائه دهنده گان مختلف برای یک سرویس معیین و انتخاب بهترین ارائه دهنده بر طبق معیارهای کیفیت سرویس و دیگر شاخص ها می باشد که توسط این اصل محقق می شود.

رابطه بین اصول معماری سازمانی سرویس گرا

با خواندن مطالب گفته شده در این بخش این سئوال به ذهن می رسد که شباهت یا تفاوت این اصول درچیست؟ برای مثال تفاوت خود مختاری با بی وضعیتی در چیست؟ چه ارتباطی بین اتصال سست با قابلیت استفاده مجدد وجود دارد؟ برای پاسخ به این سئوالات و ابهامات در ادامه به بررسی رابطه بین این اصول با همدیگر می پردازیم.

- استفاده مجدد: قابلیت کشف و شناسایی به عنوان ضرورتی برای تحقق این اصل به حساب می آید. همچنین خودمختاری و بی وضعیتی سرویس ها تاثیر مستقیمی بر قابلیت استفاده مجدد دارند زیرا به دنبال کاهش وابستگی هستند، اتصال سست نیز به نوبه خود این امر را تسهیل می نماید.
- قرارداد سرویس: این اصل را می توان پس نیاز دیگراصول دانست. درنتیجه اعمال اصول پنهان سازی، ترکیب پذیری و شناسایی، نیاز به قرارداد سرویس بوده و به نوعی این اصل تضمین کننده اجرای موارد گفته شده است.
- اتصال سست: این اصل پیش نیاز محقق شدن قابلیت استفاده مجدد، ترکیب پذیری و خود مختاری است، در صورتیکه سرویس ها با یکدیگر گره خورده و به شدت به هم وابسته باشند، امکان ترکیب پذیری، خود مختاری و استفاده مجدد از آنها با مشکلات جدی مواجه شده و درنتیجه اصول دیگر نیز خود به خود نقض می شوند.
- پنهان سازی سرویس: قابلیت استفاده مجدد به کمک این اصل تسهیل می شود چرا که ارائه سرویس از پیاده سازی آن جدا شده و یک ارائه دهنده سرویس به راحتی می تواند پیاده سازی خود را تغییر داده بدون آنکه سرویس گیرندگان دچار مشکل شوند. از طرف دیگر قرارداد سرویس جهت پشتیبانی از پنهان سازی بوده و به جای جزییات پیاده سازی تنها واسط ارائه سرویس را تعریف می نماید.
- ترکیب پذیری سرویس ها: مواردی چون اتصال سست، بی وضعیتی و قابلیت استفاده مجدد از جمله پیش نیازهای این اصل هستند و قرارداد سرویس نیز پشتیبانی کننده ترکیب پذیری سرویس ها می باشد.
- خود مختاری سرویس: این اصل از پیش شرط ها و ملزومات تحقق مواردی چون قابلیت استفاده مجدد و ترکیب پذیری بوده و نقش مهمی در بهبود اتصال سست و بی وضعیتی دارد.

- بی وضعیتی: خود مختاری و بی وضعیتی لازم و ملزوم یکدیگر هستند و بدون یکی، دیگری نیز بطور کامل محقق نمی شود. همچنین این اصل پیش نیاز ترکیب پذیری و قابلیت استفاده مجدد است.
- کشف سرویس: این اصل با کمک قرارداد سرویس محقق می شود، برای اینکه یک سرویس قابل کشف و شناسایی باشد باید قرارداد واسط خود را منتشر کند. همچنین قابلیت استفاده مجدد نیز به نوعی منوط به شناسایی و کشف سرویس هاست.

در جمع بندی مطالب گفته شده، مشخص می شود که اصولی چون اتصال سست و خودمختاری پیش نیازهای انجام دیگر اصول بوده و در عوض قابلیت استفاده مجدد و ترکیب پذیری به میزان بیشتری از دیگر اصول استفاده می کنند.

خلاصه

در این نوشته ابتدا معماری سرویس گرا را معرفی نموده و چندین تعریف برای ان ذکر کردیم, سپس فواید و ضرورت استفاده از این رهیافت را بررسی نمودیم. ارکستریشن را در معماری سرویس گرا معرفی کردیم و گفتیم معماری سرویس گرا چه مشخصاتی دارد. این مشخصات را در قالب اصولی بیان نمودیم شامل: قابلیت استفاده مجدد، قرارداد رسمی برای تعامل، اتصال سست، پنهان سازی پیاده سازی داخلی، قابلیت ترکیب پذیری، خودمختاری، بی وضعیتی و قابلیت شناسائی.

مراجع:

[1] Iranian Information Architecture committee: http://www.esoa.ir
[2] Manes, A.T. 2003, Web Services: Manager's Guide, Addison-Wesley.
[3] Chatterjee, Sandeep and Webber, 2004, Developing Enterprise Web Services: An Architect's Guide, Upper Saddle River, Prentice Hall.
[4] Linthicum, D. 2004, What Level Is Your SOA? Choose for what you need and maybe a little better, Available: http://webservices.sys-con.com/read/47277.htm
[5] Oasis: SOA Adoption Blueprint, 2006, Available: http://www.oasis-open.org/committees/download.php/17616/06-04-00002.000.doc
[6] Borges, B., Holley, K. and Arsanjani, A. 2004, Service-oriented Architecture, Available: http://searchwebservices.techtarget.com/originalContent/0,289142,sid26_gci1006206,00.html?topic=299037
[7] Khoshafian, S. 2006, Service Oriented Enterprsies, Auerbach.
[8] Knipple, R. 2005, Service Oriented Enterprise Architecture, MS Thesis, IT-University of Copenhagen.