****

**دانشکده فنی و مهندسی**

**کارشناسی ارشد مهندسی کامپیوتر نرم‌افزار**

**گروه مهندسی کامپیوتر و فناوری اطلاعات**

**موضوع:**

**پیاده سازی سرویس های نرم افزاری(بر اساس روش های مطرح شده در سمینار1)**

**نگارش:**

**علیرضا رزمجو**

**980218761**

**استاد راهنما:**

**دکتر رضوی**

**تابستان 1400**

****

**فهرست مطالب**

[فصل اول تعريف هدف 6](#_Toc90839045)

[1.1 نگاهی به پیکربندی سرویس گرا 6](#_Toc90839046)

[2.1 چرخه حيات 6](#_Toc90839047)

[3.1 مراحل soa 8](#_Toc90839048)

[4.1 اصول طراحی مدل قالب سرویس ها 13](#_Toc90839049)

[فصل دوم 17](#_Toc90839050)

[ادبيات موضوع 17](#_Toc90839051)

[1.2 تعاريف اصلي 18](#_Toc90839052)

[مدل تکامل یافته معماری سرویس گرا 33](#_Toc90839053)

[چرخه حيات معماري سرويس گرا 34](#_Toc90839054)

[2.2 پارامتر های مهم کیفی 41](#_Toc90839055)

[3.2 بررسي فاكتور دانه بندي سرويس ها 49](#_Toc90839056)

[فصل سوممعرفي روش هاي گذشته 53](#_Toc90839057)

[1.3 دسته بندي روش هاي موجود 54](#_Toc90839058)

[Zimmermann روش اول 2.3 61](#_Toc90839059)

[Zimmermann روش دوم 3.3 63](#_Toc90839060)

[Zhang روش 4.3 63](#_Toc90839061)

[SOMA روش 6.3 65](#_Toc90839062)

[Portier روش 7.3 73](#_Toc90839063)

[Inganti روش 8.3 74](#_Toc90839064)

[فصل چهارم – ويژگي هاي راه حل مورد انتظار 80](#_Toc90839065)

[1.4 نتایج مورد انتظار از سرویس ها 81](#_Toc90839066)

**چكيده**

سرویس می تواند معانی دیگری نیز داشته باشد، گاه معنای آن وسیع تر است، برای مثال یک جز توزیع شده که واسط آن با کسب و کار طراحی شده است. گاهی معنای آن جزئی تر  می باشد مانند وب سرویس مبتنی بر **SOAP**   
ابتدایی ترین مشکل برای پیاده سازی **SOA** این است که اکثریت با مفهوم اصلی ساختمان بلوکه ای، یعنی همان سرویس، موافق نیستند. بسیاری از تعارف سرویس، به مفاهیم بی ثبات و مختصر اشاره دارند، مانند قابلیت انعطاف پذیری و چابکی که برای برنامه نویسان بسیار مبهم می باشد. به جرأت می توان گفت، هیچکس در انتخاب نمی خواهد یک نرم افزار غیرقابل انعطاف و کند که نیازهای عملی را برآورده نمی سازد خلق کند. اما در واقعیت برخلاف کوشش های زیاد اغلب این اتفاق رخ می دهد. دانستن اینکه **SOA** می تواند سازگاری و چابکی به ارمغان بیاورد ستودنی است اما این به تنهایی کمک کننده نیست.  
تمامی تعاریف دیگر برای دنیای واقعی خیلی جزئی تر می باشند، برای مثال: تعیین کردن اینکه سرویس باید جزئی از نرم افزار باشد که زبانش **SOAP** است می تواند گمراه کننده باشد. اما این یک محدودیت ساختگی است، اشخاص زیادی هستند که از **REST** استفاده می کنند و فکر می کنند که در حال ارائه سرویس هستند. شاید بهتر است که تعریف ما شامل چیزهایی درباره پیام های مبتنی بر **XML** که تفاوت بین **RESTfull webservice** و **SOAP-Based service** را به وجود می آورند، باشد. اما باز هم یک محدودیت برای تعریف کلی ما به حساب می آید.  
چگونه به یک تعریف جامع و خاص برای سرویس دست پیدا کنیم که برای اهداف متنوع سرویس در دنیای واقعی پذیرفتنی باشد؟  
بهتر است تمامی جنبه های تعریف سرویس را پیش از اینکه بیشتر پیش رویم، بررسی کنیم.

# فصل اول تعريف هدف

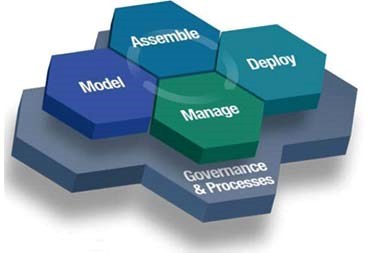
براي توسعه نرم افزارها در سطح يك سازمان، به منظور جلوگيري از بروز پيچيدگي، نياز به انتخاب سطح تجريد مناسب است. يكي از روش هاي موجود براي اين كار استفاده از سرويس ها و معماري سرويس گرا است. به طور كل مي توان گفت معماري سرويس گرا رهيافتي است براي ساخت سيستم هاي توزيع شده كه نيازمندي هاي نرم افزاري را به صورت سرويس ارائه مي كند. اين سرويس ها هم توسط ديگر نرم افزارها قابل فراخواني هستند و هم براي ساخت سرويس هاي جديد مورد استفاده قرار مي گيرند.

## 1.1 نگاهی به پیکربندی سرویس گرا

معماری سرویس گرا (Service-Oriented Architecture) نوعی الگوی طراحی است. این الگو برای ارائه خدمات به برنامه‌های دیگر از طریق پروتکل های معماری سرویس گرا طراحی شده است. در پاسخی دیگر به سوال معماری سرویس گرا چیست، می‌توان گفت رهیافتی برای ساخت سیستم‌های توزیع شده است که انواع کارکردهای نرم افزاری را در قالب سرویس ارائه می‌کند. این سرویس‌ها علاوه بر اینکه به وسیله سایر نرم افزارها قابل فراخوانی هستند، برای ساخت سرویس‌های جدید نیز مورد استفاده قرار می‌گیرند.

## 2.1 چرخه حيات

بر اساس طرح IBM براي معماري سرويس گرا مي توان يك چرخه حيات در نظر گرفت. در شكل 1 اين چرخه نشان داده شده است[9]:



**شكل شماره 1 -1: چرخه حيات معماري سرويس گرا [11]**

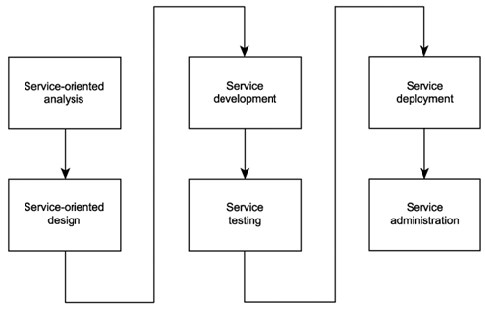
همانطور كه در شكل مشاهده مي شود، چرخه حيات معماري سرويس گرا از چهار مرحله ،مدل ،گردآوري نصب و مديريت تشكيل شده و مرحله حاكميت و فرآيندها به عنوان زيرساختي براي اين مراحل چهارگانه عمل مي كند. چارچوب كلي كار در اين چرخه حيات به اين صورت است كه در فاز مدل نيازمندي هاي حرفه جمع آوري مي شود و براساس اين نيازمندي ها فرآيندهاي حرفه طراحي مي گردند، سپس در فاز گردآوري سرويس هاي موجود و سرويس هاي جديد در كنار هم قرار داده شده و فرآيندهاي مورد نياز حرفه را شكل مي دهند. در فاز نصب فرآيندهاي توليد شده در محيط مشتري نصب شده و در فاز مديريت كاربران فرآيندهاي نصب شده را هم از لحاظ فني و هم از لحاظ مطابقت با نيازها مورد يررسي قرار مي دهند. در فاز مديريت بازخوردهاي كابران به منظور توسعه فرآيندهاي جديد حرفه به فاز مدل وارد مي شوند و چرخه را ادامه مي دهند. در زير همه اين مراحل، حاكميت و فرآيندها جهت گيري سياست هاي حرفه را براي پروژه معماري سرويس گرا تعيين مي كنند.

همانطور كه در شكل شماره 1 نشان داده شده است، اولين گام در چرخه حيات معماري گام مدلسازي مي باشد. در اين گام سرويس هاي مورد نياز براي پاسخگويي به نيازهاي حرفه تعيين شده و ويژگي هاي اصلي آنها مشخص مي شود. به عبارت ديگر در اين گام دو فعاليت اصلي انجام مي شوند: 1. تشخيص سرويس ها (service identification) و 2. تعيين ويژگي هاي سرويس ها (service specification) و 3. عينيت بخشي به سرويس ها (service realization)

در ادامه براي آشنايي بيشتر با فرآيند توليد سيستم هاي نرم افزاري با استفاده از روش معماري سرويس گرا، به بيان فازهاي پايه اي در معماري سرويس گرا مي پردازيم.

## 3.1 مراحل soa

اصولا فازهاي اصلي براي توسعه پروژه هاي مبتني بر سرويس گرايي همانند فازهاي متداول در توسعه پروژه هاي مبتني بر روش هاي پيشين براي توليد سيستم هاي توزيع شده هستند. فازهاي اصلي در سرويس گرايي در شكل زير نشان داده شده اند:



**شكل شماره 2 -1: فازهاي SOA[1]**

در شكل فوق دو فاز ابتدايي (service-oriented analysis , service-oriented design ) مسئول تطابق رهيافت در دست ساخت با ويژگي هاي شمرده شده براي سرويس گرايي هستند. در اين دو فاز ايتدايي سرويس ها شناسايي شده و از ميان مجموعه سرويس هاي كانديد، سرويس هايي كه وارد مرحله توسعه سرويس ها مي گردند تعيين مي شوند. به مجموعه اين فعاليت ها شناسايي سرويس ها (*service*

[1].مي گويند (*identification*

**1.3.1 تحليل سرويس گرا**

اهداف اصلي از تحليل سرويس گرا عبارتند از[1]:

* + - * + مشخص كردن مجموعه اصلي عمليات كانديد سرويس ها
        + گروه بندي عمليات كانديد در بسته هاي منطقي. اين بسته ها نشان دهنده سرويس هاي كانديد هستند.
        + مشخص كردن مرزهاي اصلي سرويس هاي كانديد به منظور جلوگيري از بروز رويهم افتادگي سرويس ها
        + مشخص كردن درصد قابليت استفاده مجدد براي هر بسته منطقي
        + حصول اطمينان از وجود تناسب بين بسته هاي منطقي و كاركرد مورد انتظار آنها
        + تعيين تمامي مدل هاي تركيبي ممكن مراحل تحليل سرويس گرا عبارتند از:
        + مرحله 1- تعيين نيازمندي هاي [خودكارسازي] حرفه: با توجه به اينكه نيازمندي هاي حرفه (كه به منظور پوشش آنها دست به طراحي و توسعه سيستم ها مي زنيم) جمع آوري شده اند، مستندات موجود در اين زمينه به عنوان گام آغازين در فرآيند تحليل سرويس گرا استفاده مي شوند. در اين روش تنها نيازمندي هاي مرتبط با رهيافت سرويس گرا ( آنچه مي خواهيم برايش راه حل سرويس گرا ارائه كنيم) در نظر گرفته مي شوند.
        + مرحله 2- مشخص كردن سيستم هاي خودكارسازي كنوني: در اين مرحله منطق هاي كاري موجود در سيستم هاي كنوني مورد بررسي قرار مي گيرند. اطلاعات بدست آمده در اين مرحله به منظور شناسايي سرويس هاي كانديد به كار گرفته مي شوند.
        + مرحله 3- مدلسازي سرويس هاي كانديد: پس از اينكه واحدهاي كاري شناسايي شد و اين واحدها در غالب منطق هاي عملياتي (سرويس ها) بسته بندي گرديد، اين بسته ها به صورت سرويس هاي كانديد در آمده و به منظور نمايش يك مدل ابتدايي از تركيب سرويس ها به منظور پياده سازي سيستم مورد نظر استفاده مي گردند.

**2.3.1 طراحي سرويس گرا**

در ادامه 12 مرحله به عنوان فرآيند مدلسازي سرويس ها مشخص مي گردند.[1]

* + - * مرحله 1- تجزيه فرآيندهاي حرفه

مستندات فرآيندهاي حرفه را گرفته و آن را به مراحل فرآيندها تجزيه كنيد. توجه شود كه تجزيه منطق جريان كاري فرآيندها به ريزترين درجه مراحل فرآيند اهميت زيادي دارد. اين خروجي هاي بدست آمده ممكن است در درجه ريزدانگي با مراحل فرآيندهاي موجود در مستندات متفاوت باشند

* + - * مرحله 2- تعيين عمليات كانديد سرويس هاي حرفه

برخي مراحل در فرآيندهاي حرفه از ماهيتي برخوردارند كه مي توان گفت اين مراحل به هيچ يك از واحدهاي منطقي تعلق نداشته و نمي توان آنها را در قالب سرويس هاي كانديد بسته بندي كرد. به عنوان مثال مراحل پردازش هاي دستي (manual) در فرآيندهاي حرفه كه نيازي به خودكارسازي ندارند و همچنين مراحل فرآيند كه توسط سيستم هاي موروثي انجام مي شوند و نيازي به بسته بندي در قالب سرويس ندارند از اين دسته مي باشند. با جداسازي اين مراحل از مجموعه عمليات كانديد براي سرويس هاي حرفه ، يك گام به فرآيند مدلسازي حرفه نزديك تر مي شويم.

* + - * مرحله 3- تعيين همنواسازي (*orchestration*)

اگر قصد داريد يك لايه همنواسازي براي پروژه SOA خود توليد كنيد، لازم است قسمت هاي فرآيندهاي مختلف كه در اين لايه نمايش داده مي شوند را تعيين كنيد. به عبارت ديگر براي تعيين چگونگي همنواسازي و ارتباط متقابل فرآيندها، قسمت هاي مختلف فرآيندها كه با هم در ارتباط هستند بايد تعيين شوند. توجه شود اگر قصد توليد لايه همنواسازي را نداريد، از اين مرحله عبور كنيد.

* + - * مرحله 4- توليد سرويس هاي كانديد حرفه

در اين مرحله خروجي هاي مراحل پيش را به صورت مجموعه اي از واحد هاي منطقي كار دسته بندي كنيد. اين واحدهاي منطقي كار همان سرويس هاي كانديد هستند.

* + - * مرحله 5- ارزيابي و اعمال شاخصه هاي سرويس گرايي

در اين مرحله واحدهاي عملياتي كه به صورت سرويس ها دسته بندي شده اند را از نظر تطابق با معيارهاي سرويس گرايي بررسي مي كنيم. معيارهاي اصلي سرويس گرايي عبارتند از: قابليت استفاده مجدد، خودمختاري، discoverability ،statelessness . از ميان اين چهار مفهوم دو مفهوا ابتدايي در اين مرحله مورد توجه قرار مي گيرند.

* + - * مرحله 6- تعيين تركيبات سرويس هاي كانديد

تعدادي از سناريوهايي كه به صورت متداول اجرا مي شوند را مشخص كرده و براي هر سناريو مراحل فرآيندهايي كه در مراحل پيش تعيين شده بودند را بررسي كنيد. با اين كار مي توان: ايده خوبي در مورد كيفيت بسته بندي عمليات در سرويس ها بدست آورد، ارتباطات بالقوه ميان لايه سرويس و لايه همنواسازي مشخص مي گردد، تركيبات بالقوه ميان سرويس ها تعيين مي شود، اين كار موجب آشكار شدن منطق هاي جريان كاري و مراحل فرآيندي كه در نظر گرفته نشده بودند مي شود.

* + - * مرحله 7- بازبيني گروه بندي عمليات در سرويس ها

با توجه به نتايج بدست آمده از مرحله 6، گروه بندي عمليات در سرويس ها را بررسي نموده و سازماندهي عمليات در گروه ها را در صورت نياز تغيير دهيد. ممكن است در اين مرحله سرويس هاي جديدي نيز توليد گردند.

* + - * مرحله 8- تحليل نيازمندي هاي پردازشي برنامه هاي كاربردي

تا پايان مرحله 6، سرويس هاي مبتني بر حرفه در لايه سرويس توليد شده اند. مراحل بعدي دلخواه بوده و براي فرآيندهاي پيچيده حرفه و محيط هاي بزرگ مبتني بر سرويس مناسب هستند. در مراحل آينده تمركز بر روي سرويس هاي برنامه كاربردي مي باشد. در اين مرحله تمامي گام هاي فرآيندها به منظور شناسايي نيازمندي هاي پياده سازي و فناوري مورد تحليل قرار مي گيرند

* + - * مرحله 9- تعيين عمليات كانديد سرويس هاي برنامه هاي كاربردي

برنامه هاي كاربردي را به مراحل كوچك تجزيه كنيد. توجه كنيد اين مراحل را طوري نامگذاري كنيد كه ارتباط آنها با توابع شان مشخص باشد.

* + - * مرحله 10-توليد سرويس هاي كانديد برنامه هاي كاربردي

خروجي هاي مرحله پيش را در سرويس هاي برنامه كاربردي گروه بندي كنيد. منطق گروه بندي اينجا ارتباطات منطقي بين مراحل فرآيندها است.

* + - * مرحله 11- بررسي تركيبات سرويس هاي كانديد

سناريوهايي را كه در مرحله 5 مشخص كرده ايد را بررسي نماييد، با اين تفاوت كه در اين مرحله سرويس هاي برنامه هاي كاربردي را نيز دخالت دهيد. در اين مرحله سعي نماييد تناظري بين سرويس هاي برنامه كاربردي و سرويس هاي حرفه ارائه دهيد.

* + - * مرحله 12- بررسي گروه بندي سرويس هاي برنامه هاي كاربردي

پس از بررسي سناريوها در مرحله 11، امكان بروز تغييراتي در گروه بندي سرويس ها وجود دارد.

اين امر به خاطر آشكار شدن مراحل از قلم افتاده در گام هاي پيشين است.

پس از اين مرحله تمامي سرويس هاي كانديد و نحوه تركيب آنها مشخص گرديده اند. در برخي منابع توصيه شده است تا در يك مرحله دلخواه (مرحله 13) سرويس هاي كشف شده در محلي ذخيره سازي گردند تا در صورت نياز در آينده مورد بررسي قرار گيرند.

اكنون كه با فازهاي اصلي معماري سرويس گرا آشنا شديم و فعاليت هاي عمده در اين فازها را مطالعه كرديم، با يكي از مهمترين گام ها در چرخه حيات معماري سرويس گرا يعني گام مدلسازي سرويس ها (service modeling) بيشتر آشنا مي شويم.

## 4.1 اصول طراحی مدل قالب سرویس ها

با توجه به شكل 1 مشاهده مي كنيد مدلسازي سرويس ها اولين قدم در چرخه حيات معماري سرويس گرا است. اصلي ترين هدف از مدلسازي سرويس ها بالا بردن سطح تجريد و كاهش پيچيدگي در كار با سرويس ها مي باشد. براي اين منظور ابتدا در دامنه مورد بررسي، سرويس هاي مورد نياز را شناسايي مي كنيم. اين شناسايي بر اساس اهداف حرفه مورد بررسي و فرآيندهاي آن حرفه انجام مي شود.

سپس ويژگي هاي هريك از اين سرويس ها را به صورت جزئي تر مشخص كرده و نحوه تعامل آنها را با هم تعيين مي كنيم. براي اين كار از نمودارهاي مختلفي مانند نمودار ارتباط سرويس ها، نمودار همكاري بين سرويس ها (collaboration diagram) و ... استفاده مي شود.

در آخر به عنوان آخرين مرحله از مدلسازي سرويس ها، آنها را بر اساس ويژگي ها و خصوصياتشان و هم چنين با توجه به ويژگي هاي حرفه مورد بررسي از نظر چگونگي توليد مورد بررسي قرار مي دهيم. به اين فعاليت عينيت بخشي به سرويس ها گفته مي شود. در اين قدم مشخص مي شود كه مثلا يك سرويس به طور كامل پياده سازي شود و يا با استفاده از سرويس هاي موجود در سيستم هاي موروثي پياده سازي شود و ...

فعاليت هايي كه در بالا به عنوان فعاليت هاي مرحله مدلسازي سرويس ها به آنها اشاره شد، سه فعاليت اصلي در حيطه مدلسازي سرويس ها بوده و عبارتند از مرحله تشخيص سرويس ها (service identification)، مرحله توصيف سرويس ها (service specification) و مرحله عينيت بخشي به سرويس ها (service realization). در اين گزارش سمينار، تمركز روي گام اول يعني تشخيص سرويس ها (service identification) و بررسي اهداف، ويژگي ها و فعاليت هاي پيشنهاد شده براي اجراي اين گتم است. در ادامه پس از آشنا شدن با جايگاه اين فاز و بيان اهميت آن با معيارهاي سنجش تناسب سرويس هاي شناخته شده و روش هاي ارائه شده در اين زمينه آشنا مي شويم. سپس با بررسي كيفيت و توجه به هر يك از فاكتورهاي سرويس ها در روش هاي ارائه شده، به نقد اين روش ها پرداخته و سعي در تعريف راه حلي جديد به منظور پوشش كاستي هاي اين روش ها داريم.

**1.4.1 اهميت گام تشخيص سرويس ها**

مي دانيم كه تشخيص سرويس ها اولين قدم از فاز مدلسازي سرويس ها است و فاز مدلسازي سرويس ها نيز خود اولين فاز از چرخه حيات معماري سرويس گرا مي باشد. بنابراين براي ارائه راه حل سرويس گرا در حيطه يك حرفه يا سازمان اولين فعاليت انجام شده، تشخيص سرويس ها مي باشد. اهميت اين گام از همين جا ناشي مي شود. زيرا اگر در اين گام سرويس هاي نامناسبي را به اشتباه تشخيص دهيم، در ادامه كار به مشكل بر خواهيم خورد. اهميت اين فاز از سه جنيه قابل بررسي است :

1. وابستگي گام هاي بعد به خروجي اين گام: از آنجا كه اين گام اولين گام در فاز مدلسازي سرويس گرا و همچنين در تمام چرخه حيات معماري سرويس گرا مي باشد، خروجي اين گام به عنوان مبنايي براي تمامي گام هاي آينده بوده و خروجي گام هاي بعد بر اساس خروجي اين گام توليد مي شوند. به همين علت اگر در اين گام سرويس هاي نادرستي را تشخيص دهيم، در گام هاي بعد بر اساس اين سرويس هاي نادرست، راه حل و سيستم ناقص و نادرستي را توليد خواهيم كرد.
2. هزينه و زمان مورد نياز براي رفع خطاها: با توجه به اينكه خطاهاي احتمالي در اين فاز در تمامي فازهاي آينده تاثير خواهند گذاشت، در صورتي كه در آينده متوجه بروز خطاها شويم به هزينه و زمان بيشتري جهت رفع اين خطاها نياز خواهيم داشت. به عنوان مثال هنگام پياده سازي سرويس ها در فاز استقرار (deploy) اگر از وجود سرويس هاي نادرست و نابه جا آگاه شويم بايد هزينه تمامي فعاليت هاي فازهاي گردآوري (assemble) و مدلسازي (model) را دوباره پرداخت كنيم و به همين نسبت نيز به زمان بيشتري نياز خواهيم داشت. بنابراين اگر در اين فاز با استفاده از روشي خطاهاي احتمالي را تشخيص دهيم، به ميزان قابل ملاحظه اي بر كيفيت محصول نهايي و كاهش هزينه و زمان توليد محصول تاثير خواهيم گذاشت.
3. تسهيل فرآيند :نگهداري از آنجا كه معماري سرويس گرا يك فرآيند موردي نبوده و همواره در دوره حيات حرفه و سازمان ادامه خواهد داشت، پيش بيني نيازهاي آينده و تشخيص سرويس هاي مورد نياز در آينده بر تسهيل فرآيند توسعه سيستم تاثير مثبت خواهد داشت. به همين علت اگر در فاز تشخيص سرويس ها با دانستن هدف نهايي حرفه و داشتن آگاهي درست نسبت به تغييرات احتمالي در آينده حرفه بتوان سرويس هايي را تشخيص داد كه علاوه بر پوشش نيازهاي اكنون در حيطه سازمان، تا حدي نيازهاي احتمالي آينده سازمان را نيز تحت پوشش خود قرار دهند و همچنين سرويس هايي با امكان پيكربندي آسان طراحي نمود، مي توان تاحد بسيار زيادي فرآيند نگهداري سيستم توليد شده را تسهيل بخشيده و هزينه و زمان مورد نياز براي اعمال تغييرات در فرآيندهاي سيستم و وظيفه مندي هاي آن را به ميزان قابل توجهي كاهش داد.

علاوه بر سه جنبه فوق گام تشخيص سرويس ها از جنبه هاي ديگري نيز اهميت دارد و قابل بررسي است. اما سه جنبه فوق به عنوان اصلي ترين فاكتورهاي اهميت براي اين گام مطرح بوده و اساس توجه به اين گام مي باشند.

**نتيجه گيري**

در اين فصل پس از معرفي مفهوم معماري سرويس گرا، اهداف فازها و چرخه حيات آن با گام هاي موجود در چرخه حيات معماري سرويس گرا تا حدي آشنا شديم. همچنين گام اولين اين چرخه، يعني مدلسازي سرويس گرا را شرح داده و نسبت به فعاليت هاي موجود در اين گام شناخت كلي پيدا نموديم. در ادامه فعاليت تشخيص سرويس ها به عنوان اولين فعاليت در فاز مدلسازي و چرخه حيات معماري سرويس گرا مورد بررسي قرار گرفت و دلايل اهميت اين گام تعيين گرديد.

پس از آشنايي با جايگاه تشخيص سرويس ها (service identification) و دلايل اهميت آن در ادامه تلاش داريم روشي را به منظور انجام اين فعاليت معرفي كنيم. براي اين كار در فصل هاي آينده به معرفي مفاهيم موجود در حيطه تشخيص سرويس ها پرداخته و سپس به معرفي روش هاي موجود در اين زمينه مي پردازيم. همچنين اين روش ها را به منظور تطبيق با فاكتورهاي مورد نظر براي سرويس هاي تشخيص داده شده بررسي نموده و نقاط قوت و ضعف هريك از اين روش ها را بيان مي نماييم. پس از شناخت كاستي هاي موجود در اين روش ها به معرفي روشي جديد جهت رفع اين نواقص مي پردازيم.

# فصل دوم

# ادبيات موضوع

**مقدمه**

در فصل قبل در مورد مفهوم سرويس،معماري سرويس گرا، فازها و چرخه حيات آن توضيحاتي داده شد.

همچنين با فازهاي تشخيص سرويس ها (service identification)، توصيف سرويس ها (service specification) و عينيت بخشي به سرويس ها (service realization) به عنوان فعاليت هاي اصلي در گام مدلسازي سرويس ها (service modeling) از چرخه حيات معماري سرويس گرا آشنا شديم و اهميت فاز تشخيص سرويس ها بيان گرديد.

در اين فصل به منظور چلوگيري از بروز ابهام، براي هر يك از مفاهيم موجود در دامنه مسئله تشخيص سرويس ها، تعريف روشن و شفافي ارائه خواهد شد. همچنين برخي از اصلي ترين فاكتورهاي كيفي سرويس ها بيان خواهند گرديد. اين فاكتورهاي كيفي به عنوان معياري جهت سنجش ويژگي كيفي سرويس هاي تشخيص داده شده حين فرآيندهاي تشخيص سرويس ها ، مورد توجه قرار خواهند گرفت. در آخر نيز فاكتور دانه بندي سرويس ها به عنوان فاكتوري تاثير گذار بر معيارهاي كيفي سرويس ها معرفي شده و نحوه ارتباط آن با ساير فاكتورهاي كيفي تشريح مي گردد.

## 1.2 تعاريف اصلي

همانطور كه پيش از اين عنوان شد در ادامه براي هريك از مفاهيم م.جود در دامنه تشخيص سرويس ها

.تعريفي ارائه خواهد گرديد (service identification)

**سرويس**

كاري كه به وسيله يك سرويس دهنده انجام ميشود كه ممكن است انجام يك درخواست كوچك روي داده مانند دريافت يا ذخيره اطلا عات باشد يا مربوط به انجام كاري پيچيده تر مانند چاپ يك تصوير باشد. در معماري سرويس گرا معمولا سرويس را به صورت خاص تر معرفي مي كنند و به نوعي مضمون آن را با شئ و مؤلفه مرتبط مي دانند. واضح است كه ضرورتي براي استفاده از متدولوژي شئ گرا يا مبتني بر مؤلفه جهت معماري سرويس گرا وجود ندارد و اصلا همان گونه كه بارها گفته شده معماري سرويس گرا مستقل از سكو است، اما مي توان گفت مفهوم و مضمون »سرويس« شباهت هايي با شئ و مؤلفه داشتهاست.

از آنجا كه در حيطه نرم افزارهاي مقياس وسيع در سطح سازمان ها استفاده از سطح تجريد كلاس ها، اشيا و ... به علت گستردگي حيطه نرم افزار باعث افزايش پيچيدگي مي گردد، از مفهوم سرويس به عنوان ابزاري جهت بالا بردن سطح تجريد و در نتيجه كاهش پيچيدگي استفاده مي كنيم. به عبارت ديگر سرويس ها سطح بالايي از تجريد را معرفي مي كنند. در سازمان هاي بزرگ كه از چندين حرفه مرتبط با هم تشكيل شده اند، مي توان براي پاسخ به نيازهاي هر حرفه، سرويسي ارائه نمود و اين سرويس ها را به منظور پوشش تمامي نيازهاي سطح سازمان با هم مجتمع كرد.

از جهت ديگر، سرويس ها را مي توان مصداق تكميل شده IC هاي نرم افزاري دانست. همانطور كه در دنياي سخت افزار با استفاده چينش مختلف از IC ها مي توان مدارهاي مختلف را توليد كرد ايده استفاده از چنين تكنيكي در ذهن مهندسين نرم افزار پديد آمد. به اين منظور ابتدا روش هاي توسعه سيستم هاي نرم افزاري مبتني بر مؤلفه ها (component based development) يا CBD معرفي شدند و در ادامه روش هاي مبتني بر سرويس يا روش هاي سرويس گرا (service oriented) ظهور پيدا كردند.

**مؤلفه**

همانطور كه پيش از اين اشاره شد، مولفه ها نقش ICهاي نرم افزاري را دارند. علاوه بر اين بارزترين ويژگي مولفه مستقل از سكو بودن (platform independent) آنهاست. اين ويژگي به اين معناست كه مولفه ها به پياده سازي خاصي وابسته نيستند و مي توانند روي سكوهاي سخت افزاري ،سيستم عاملي و نرم افزاري متفاوتي اجرا شوند.

اصلي ترين تفاوت ميان مؤلفه ها و سرويس ها در اين است كه سرويس ها از طريق شبكه ارتباطي قابل دستيابي هستند. اين ويژگي باعث افزايش چشمگير قابليت استفاده مجدد سرويس ها مي گردد. به اين ترتيب براي استفاده از سرويس هاي مشخص با كاركرد معين استفاده به پياده سازي آنها نيست بلكه مي توان در صورت يافتن سرويس هايي با قابليت هاي مورد نياز از طريق برقراي تفاهم نامه با ارائه دهنده سرويس از آن استفاده كرد. همچنين سطح تجريد سرويس ها از سطح تجريد مولفه ها بوده و به اين منظور استفاده از سرويس ها در سيستم هايي با دامنه هاي وسيع به علت كاهش پيچيدگي توصيه مي گردد.

**معماري سرويس گرا**

»معماري سرويس گرا« مفهومي جديد نيست و از دهه 90 وجود داشته است ولي آنچه جديد است توانائي اجرا و عينيت بخشيدن به آن است كه به كمك ابزارها و پروتكل هاي مربوطه ميسر شده است. اين معماري توسط دو شركت IBM, Microsoft بوجود آمد، كه هر دو شركت طي سالهاي اخير از حاميان اصلي سرويسهاي وب و عامل بسياري از ابداعات جديد در حيطه سرويسهاي وب، مانند UDDI ,WSE بوده اند. سيستم هاي اطلاعاتي به سرعت در حال رشد هستند؛ سازمان ها نيازمند پاسخگويي سريع به نيازمندي هاي جديد كسب وكار هستند. اين در حالي است كه معماريهاي نرم افزاري موجود به حد نهايي قابليتهاي خود رسيدهاند. معماري مبتني بر سرويس قدم تكاملي بعدي براي كمك به سازمانها جهت مديريت چالشهاي پيچيده است. معماري مبتني بر سرويس حالت بلوغ يافته معماري مبتني بر اجزا، طراحي مبتني بر واسطه (شي گرا) و سيستم هاي توزيع شده است. در معماري مبتني بر اجزا عملكرد كلي به كارهاي كوچك تري تقسيم ميشود كه هر يك در يك جزء بسته بندي خواهند شد. يك سيستم توزيع شده، تعميمي از يك معماري مبتني بر اجزا است كه به اجزائي كه در موقعيتهاي فيزيكي مختلف وجود دارند، اشاره مي كند. مهم ترين مزيت معماري مبتني بر اجزا سهولت در استفاده مجدد و تغيير هدف اجزاي خاص و سهولت در امر نگهداري سيستماست. استفاده مجدد و تغيير هدف معمولاً مهم ترين پيشران هاي كسب و كار جهت استفاده از اين نوع معماري در دهه 90 ميلادي بودهاست. بر اساس منطق معماري مبتني بر سرويس، سيستمهاي نرم افزاري بزرگ مي توانند از گردآوري مجموعههايي از عملكردهاي مستقل و قابل استفاده مجدد تشكيل گردند. برخي از اين عمليات مي تواند از طريق سيستمهاي موجود و يا سيستم هاي ديگر فراهم گردد ولي ساير عمليات لازم بايد پياده سازي شوند. هر سرويس امكان دسترسي به مجموعه خوش تعريفي از عمليات را ميدهد. سيستم به عنوان يك كل به صورت مجموعه اي از تعاملات بين اين سرويس ها طراحي ميشود. معماري مبتني بر سرويس، سرويس هايي را كه سيستم از آنها تشكيل شده را تعريف ميكند و تعاملات لازم بين سرويسها جهت ارائه رفتار مشخص را توصيف ميكند و در نهايت سرويسها را به يك يا چند پياده سازي در تكنولوژي هاي خاص تصوير ميكند. معماري سرويس گرا بر اساس استفاده از اشياء و اجزا توزيع شده است و قدم تكامل بعدي در محيطهاي محاسبه اي است. اين معماري در حال حاضر مدل مرجع استانداردي ندارد؛ اما پياده سازيهاي موجود مفاهيم مشتركي را مورد استفاده قرار ميدهند.[9]

**تفاوت سرويس و معماري سرويس گر**ا

آن گونه كه به نظر ميرسد در مورد ارتباط ميان معماري سرويس گرا و سرويسهاي وب نوعي سردرگمي عمومي وجود دارد. در يكي از گزارش هاي Gartner مورخ آوريل 2003، V. Natis Yefim اين گونه تقاوت ميان آنها را شرح مي دهد: “سرويس هاي وب راجع به مشخصه هاي تكنولوژي هستند، در حالي كه معماري سرويس گرا يك قاعدهٔ طراحي نرم افزار است. شايان ذكر است كه WSDL سرويس هاي وب يك استاندارد تعريف رابط مناسب معماري سرويس گرا است: اين نقطهاي است كه سرويس هاي وب و معماري سرويس گرا اساسا به يكديگر پيوند مي خورند.” اساسا، معماري سرويس گرا يك الگوي معماري است، در حالي كه سرويسهاي وب سرويسهاي پيادهسازي شده توسط مجموعه اي از استانداردها ميباشند؛ سرويس هاي وب يكي از روش هايي است كه شما با استفاده از آن مي توانيد معماري سرويس گرا را پ يادهسازي نماييد. مزيت پيادهسازي معماري سرويس گرا با سرويس هاي وب اين است كه شما به يك رويكرد بي طرفانه نسبت به پلات فرم به منظور دستيابي به سرويس ها و قابليت همكاري بيشتر با ساير قسمت ها دست مي يابيد همچنان كه فروشندگان بيشتر و بيشتري مشخصههاي بيشتر و بيشتري از سرويسهاي وب را پشتيبانيمي نمايند

**چرا از معماري سرويس گرا استفاده ميكنيم؟**

واقعيت موجود در سازمان هاي IT اين است كه زيربنا در ميان سيستم هاي عامل، برنامه هاي كاربردي، نرم افزارهاي سيستمي، و زيربناي كاربردي به صورت ناهمگن است. برخي برنامه هاي كاربردي موجود براي اجراي فرايندهاي فعلي تجارت مورد استفاده قرار گرفته اند، بنابراين آغاز از صفر براي ساختن زيربناي جديد يك رويكرد قابل انتخاب محسوب نميگردد. سازمان ها بايد به شكلي سريع به تغييرات تجاري واكنش نشان دهند؛ از سرمايههاي موجود در برنامههاي كاربردي و زيربناي كاربردي به منظور تمركز بر روي نيازمندي هاي تجاري جديدتر استفاده نمايند؛ كانالهاي جديد تعامل با مشتريان، شركا، و تامين كنندگان را پشتيباني كنند؛ و يك معماري كه تجارت ارگانيك را پشتيباني نمايد به كار گيرند. معماري سرويس گرا با طبيعت اتصال آزادانه خود به سازمان ها امكان بهره گيري از سرويس هاي جديد يا ارتقاي سرويس هاي موجود را به شيوه اي قطعه قطعه به منظور تمركز بر نيازمنديهاي تجاري فراهم مي آورد، امكاني را براي قابل استفاده نمودن سرويس ها در كانال هاي متفاوت فراهم مي سازد، و سازمان موجود و برن امههاي كاربردي نسل قبل را به عنوان سرويس ها ارايه ميكند، در نتيجه سرمايه هاي زيربناي IT موجود را حراست مي نمايد.

**تعريف معماري سرويس گرا**

براي معماري سرويس گرا تعاريف متنوع و بعضا مختلفي ارائه شده كه هر كدام از نگاهي به تبيين خصوصيا ت آن پرداخته اند، براي درك بهتر اين مفهوم و آگاهي از كليه برداشتها و نگاههاي موجود، در ادامه تعدادي از اين تعاريف آورده شده است.

* يك چارچوب راهبردي از فناوري كه به تمام سيستمهاي داخل و خارج اجازه ارائه يا دريافت سرويس هاي خوش تعريف را مي دهد[17].
* روشي براي طراحي و پياده سازي نرم افزارهاي گسترده سازماني به وسيله ارتباط بين سرويس هائيكه داراي خواص اتصال سست، دانه درشتي و قابل استفاده مجدد هستند[18].
* سبكي از معماري كه از اتصال سست سرويسها جهت انعطاف پذيري و تعامل پذيري حرفه و بصورت مستقل از فناوري پشتيباني مي كند و از تركيب مجموعه سرويس هاي مبتني بر حرفه تشكيل شده كه اين سرويس ها انعطاف پذيري و پيكربندي پويا را براي فرآيندها محقق ميكنند[19].
* چارچوبي وسيع و استاندارد كه سرويس ها در آن ساخته، استقرار و مديريت مي شوند و هدفش افزايش چابكي زير ساختهاي فناوري اطلاعات در جهت واكنش سريع به تغييرات در نيازهاي كسب و كار ميباشد[20].

**معماري سرويس گراي مقدماتي**

معماري سرويس گراي مقدماتي، راهي براي تبادل اطلاعات بين عاملهاي نرم افزاري بوسيله پيغام تعريف مي نمايد. اين عامل ها، درخواست كننده سرويس (مشتري) و يا تهيه كننده سرويس مي باشند. علاوه بر اين دو، عامل ديگري بعنوان عامل كشف سرويس نيز وجود دارد. در معماري سرويس گرا معرفي سرويس ها و همچنين نحوه ارتباط اين سه شركت كننده نيز اهميت دارد. اين ارتباطات عبارتند از: منتشر كردن سرويس، پيدا كردن سرويس و متصل شدن به سرويس. در يك سناريو بر پايه سرويس، تهيه كننده، سرويس را پياده سازي كرده و از طريق شبكه به ارائه توضيحات آن سرويس براي درخواست كننده يا عامل كشف سرويس مي پردازد. در خواست كننده معمولا درخواست پيدا كردن سرويس را به عامل كشف سرويس ميدهد تا از طريق آن به توضيحات ارائه شده سرويس و محل آن دسترسي پيدا كند. سپس با بكارگيري اين اطلاعات به تهيه كننده سرويس متصل شده و از سرويس ارائه شده استفاده مي نمايد.

**معماري سرويس گراي توسعه يافته**

معماري مقدماتي در لايه پاييني اين معماري لايه اي قرار گرفته است. لايه تركيب سرويس در معماري توسعه يافته، شامل توابع و نقش هاي لازم براي يكپارچه كردن چند سرويس بعنوان سرويس تركيبي مي باشد. سرويس تركيبي بدست آمده، توسط Aggregator Service بعنوان يك سرويس مقدماتي استفاده ميگردد و ياتوسط درخواست كنندگان سرويس بكارگرفته ميشود. Service Aggregator تهيه كننده سرويسي است كه سرويس هاي ارائه شده توسط ساير تهيه كنندگان را يكپارچه مينمايد تا از آنها سرويس هاي جديد بسازد، همچنين مشخصات و كدهايي را تهيه مي كند تا در مورد سرويس هاي تركيبي عمليات زير را انجام دهد:

* متناسب كردن : كنترل اجراي سرويس هاي تركيب شده و مديريت گردش داده ها در بين آنها و انتقال آن به خروجي.
* كنترل كردن : مجوز دادن به رخدادها و اطلاعات توليد شده توسط سرويسهاي تركيبي جهت به اشتراك گذاشتن و منتشر كردن رخدادهاي تركيبي سطح بالاتر (براي مثال از طريق فيلتر كردن و خلاصه سازي)
* مطابقت دادن : حصول اطمينان از حفظ جامعيت سرويس هاي تركيبي از طريق تطبيق دادن محدوديتها و نوع پارامترهاي سرويس هاي بكار رفته.
* تركيب خواص سرويسها : بكارگيري، مجتمع سازي و دسته بندي ويژگيهاي سرويسهاي تركيب شده جهت بدست آوردن خواص تركيبي جديد كه دربردارنده كارايي، هزينه، امنيت، جامعيت، قياس پذيري، در دسترس بودن و قابليت اطمينان مي باشد.

**مفاهيم اصلي در معماري سرويس گرا**

**(loose coupling) اتصال سست**

يك ويژگي براي سيستمهاي اطلاعاتي است كه در آن واسط هاي بين اجزاء(ماژولها) به گونه اي طراحي مي شوند كه وابستگي بين اين اجزاء حداقل شود و در نتيجه ريسك اثر تغيير يك جزء بر ساير اجزاء كاهش يابد. درمعماري سرويس گرا منظور از اتصال سست قابليت تعامل بين سرويس ها به صورت مستقل از كدنويسي و مكان سرويسها است، بگونه اي كه سرويس ها در زمان اجرا مي توانند تغيير مكان داده، روالهاي داخلي خود را تغيير دهند يا حتي از يك فناوري جديد تر استفاده كنند، بدون اينكه تاثيري منفي بر سرويس گيرندگان گذاشته شود.

**هم نواسازي و هم خواني (orchestration)**

دو واژه پر كاربرد در حوزه كسب و كار و معماري سرويس گرا كه معمولا به جاي هم اشتباه گرفته مي شوند، هم نواسازي و هم خواني نام دارند. هم نواسازي در خصوص ترتيب اجراي سرويس ها در فرآيند بحث مي كند، اركستر اصلي مجموعهاي از سرويس ها را فراخواني مي كند تا نتيجه مورد نظر حاصل شود و فرآيند تكميل گردد، ممكن است سرويسهاي خارج سازمان نيز در اين راستا فراخواني و استفاده شوند، اين كار با كمك موتور فرآيند محقق مي شود. در عوض هم خواني به فرآيندهايي گويند كه بدون موتور فرآيندي(رهبر اركستر) اقدام به تبادل پيام كرده و ترتيب و توالي پيامهاي مبادلاتي را خود بازيگران ثبت و كنترل ميكنند. در هم نواسازي، يك كنترل كنندهٔ مركزي، جريان گردش كارها را بين چندين عامل (سرويس، كارگر، سيستم، ...) تقسيم مي كند.يكي از كاربرهاي اين مفهوم در شكستن فرآيند هاي بزرگ به اجزاي كوچكتر است به طوريكه اين اجزا تحت نظارت هم نواساز اصلي عمل نموده و نتيجه آنها براي همان هم نواساز ارسال شود. اين كار پيچيدگي كار را كاهش ميدهد، بدين ترتيب منطق جريان كار به صورت جداگانه نگهداري ميشود و بسط و تغيير آن ساده تر مي شود. اجزاء(عامل) نبايد دانشي از منطق جريان كار اصلي داشته باشند، آنها فقط به در خواست هاي هم نواساز پاسخ مي دهند و هر جزء يك واحد خود شمول و مستقل به حساب مي آيد.

**مزاياي معماري سرويس گرا**

مولفان و شركت هاي پشتيباني كننده معماري سرويس گرا در خصوص مزاياي استفاده از اين رهيافت دلايل زيادي را مطرح كرده اند كه در ادامه بعضي از آنها تشريح مي شود:

1. سيستم هاي چابك: معماري سرويس گرا شما را قادر ميسازد تا به سرعت سيستم هاي خود را تغيير دهيد. اين چابكي هم از جهت كاركردهاي سيستم و هم از جهت تغيير جغرافيائي يا ارتقاء سكوها و حتي تغيير تامين كننده فناوري مي تواند باشد.
2. يكپارچگي آسان با شركاء داخلي و خارجي : مي توان گفت قابليت يكپارچگي سيستمها و سكوها مهمترين موردي است كه معماري سرويس گرا به آن پرداخته است.
3. استفاده مجدد : استفاده مجدد از كد برنامه يا سيستم ها، از گذشته مورد توجه متدهاي توليد و توسعه نرم افزار بوده است، معماري سرويس گرا قابليت استفاده مجدد را هم در سطح كاركردي(سرويس) و هم در سطح داده ها مهيا ميكند.
4. پشتيباني از محصولات با طول عمر كوتاه : رقابت تجاري در دنيا به شدت افزايش پيدا كرده و نياز به كاهش زمان بازاريابي و توليد براي محصولات جديد ميباشد. معماري سرويس گرا وعده ميدهد كه با وجود سرويس هاي خوش تعريف و قابليت استفاده مجدد از آنها در يك سازمان، پشتيباني سريع از محصولات جديد امكان پذير است.
5. بهبود بازگشت سرمايه: معماري سرويس گرا مجموع هزينه صرف شده براي فناوري اطلاعات و سرويس هاي حرفه را به دو روش كاهش مي دهد. اول با حذف هزينههاي ميان افزارها و فناوريهاي اختصاصي و جايگزين كردن آن با فناوري هاي استاندارد مانند وب سرويس و دوم با تركيب كاركردهاي حرفه در غالب سرويس هائي كه توسط واحدهاي مختلف قابل استفاده باشد.
6. نگاشت مستقيم فرآيندهاي حرفه به فناوري اطلاعات: نقش كليدي معماري سرويس گرا اتصال بين كسب و كار و فناوري اطلاعات است، بدين ترتيب فرآيندها ميبايست از نگاه سرويس گرا ديده شوند و در سطح مديريت حرفه پشتيباني شوند.
7. توسعه و اجراي تدريجي: معماري سرويس گرا يك پروژه عظيم و بزرگ و يكجا نيست بلكه از تكامل و تبديل تدريجي سيستمهاي فعلي و تعريف سرويسهاي جديد بصورت تدريجي ايجاد مي شود.
8. قابليت انعطاف و تغيير آسان از يك ارائه دهنده سرويس به ديگري : موضوع انعطاف در معماري سرويس گرا در هر دو مورد سرويس هاي داخل سازماني و خارجي صدق ميكند.

زباني مبتني بر XML كه جهت توصيف ويژگيهاي عملياتي سرويس هاي وب استفاده ميشود و داراي دو بخش تعريف واسط و پياده سازي است. قسمت واسط براي استفاده متقاضيان سرويس بوده و ممكن است شامل چندين پياده سازي باشد درحاليكه تعريف پياده سازي مشخص ميكند كه چگونه واسط به وسيله يك ارائه دهنده مشخص پياده سازي شده است. اجزاء تشكيل دهنده WSDL

1. نوع(type): پارامترهاي ارسالي و دريافتي را مشخص مي كند.
2. پيام (message): پارامترهاي ورودي و خروجي و نوع آنها را مشخص مي كند، پيام ميتواند شامل چند بخش باشد.
3. عمليات(operation): متدهاي سرويس هاي وب بوده و داراي پيامهاي ورودي و خروجي هستند.
4. نوع درگاه(port type): مجموعه اي از عمليات است.
5. مقيدسازي(binding): مشخص مي كند چگونه عمليات مربوط به نوع درگاه فراخواني مي شود.
6. سرويس(service): مجموعه اي از نقاط انتهائي.

**UDDI**

برنامه واسطي است براي انتشار و شناسائي سرويسهاي وب و شامل يك مخزن ميشود كه ارائه دهندگان به انتشار و تبليغ سرويس خود ميپردازند تا ديگران بتوانند آن را شناسائي كنند. از نظر مفهومي مخزن شامل صفحات سفيد(اطلاعات سرويس ها)، صفحات زرد(دسته بندي صنعتي) و صفحات قرمز(اطلاعات فناوري) خواهد بود و داراي اين ويژگي ها است:

* كتابچه اي براي ذخيره اطلاعات مربوط به سرويس هاي وب است
* در آن واسط سرويس هاي وب كه توسط WSDL توصيف شده است، ذخيره ميشود.
* ارتباطات با آن توسط SOAP است.

**BPEL**

زبان اجراي فرآيند هاي حرفه با اين مشخصات:

* زباني مستقل از سكو و مبتني بر XML
* زباني براي توصيف رفتار فرآيند هاي حرفه به كمك سرويس ها
* داراي ساختارهايي براي كنترل جريان و شرطهاي انشعاب است.
* قابليت پوشش به موارد پيچيده تري چون فرآيندهاي تودر تو و الحاق و شكست زير فرآيندها را دارد
* به عنوان يك استاندارد با حق امتياز رايگان ارائه شده
* از WSDLبراي توصيف واسط سرويس ها استفاده ميشود.

**متدولوژي بهبود مداوم براي معماري سرويس گرا**

سيستم هاي سرويس گرا مي بايست بصورت تدريجي و تكاملي تهيه و توسعه داده شوند، ايجاد سازمان مبتني بر سرويس هاي اتوماسيون شده بصورت يكدفعه و در يك پروژه ميسر نيست، بنابراين لازم است توسعه تدريجي مد نظر قرار گيرد، از طرف ديگر پس از چندين مرحله و پياده سازي تمامي سرويس ها در سازمان كار پايان نميپذيرد، خصوصيات سيستم هاي سرويس گرا تغييرات آنها بر طبق نيازهاي حرفهاست، ايجاد سرويس هاي جديد يا تغيير و اصلاح پياده سازي سرويس هاي موجود بايد در نظر گرفته شود و لذا توسعه مداوم بايد در دستور كار مديران فناوري اطلاعات قرار گيرد. در ادامه راهنمائي براي ايجاد معماري سازماني معرفي خواهد شد، اگرچه اين روش توسط مولف آن تحت عنوان متدولوژي بهبود مداوم ارائه شده، وليهمانطور كه گفته شد معماري سازماني هم اكنون داراي يك متدولوژي مدون و مورد قبول نبوده و آنچه هم اكنون تحت عنوان متد وجود دارد درحقيقت راهنمائي ها و تمرين هايي در اين حوزه محسوب ميشوند[7]

**مراحل متدولوژي**

1. تعريف و طراحي سرويس

اين مهمترين مرحله در متدولوژي بوده و نيازمندي هاي مربوط به آن از طريق مشتريان داخلي، صاحبان حرفه و شركاء تجاري بدست مي آيد. براي تعريف سرويس سه روش وجود دارد:

* تعريف وپياده سازي سرويسهاي جديد بر اساس نيازمنديها: در اين روش سرويس هاي مورد نياز بصورت كامل تعريف وپياده سازي ميشوند. اين سرويسها بگونه اي تعريف و طراحي ميشوند كه علاوه بر برآورده نمودن نياز درخواست كنندگان مستقيم آن، توسط ساير سرويسها نيز قابل فراخواني و استفاده باشند.
* تعريف سرويس هاي جديد بر اساس مولفه ها يا سيستم هاي اطلاعاتي موجود: در اين روش واسطي براي استفاده از كاركردها و امكانات سيستمهاي اطلاعاتي موجود تهيه ميشود. اين نوع سرويس ها فاقد منطق حرفه يا كد پياده سازي بوده و صرفا واسطي براي تعامل با سيستم هاي موروثي هستند و حيات آنها وابسته به اين سيستم ها است.
* تعريف سرويسها با تركيب سرويسهاي موجود: در اين روش درصورتيكه واحدهاي سازنده اصلي براي براورده كردن نيازمنديها بصورت آماده موجود باشد، سرويس جديد تنها نياز به فراخواني مجموعه اي از سرويس هاي موجود وپايه دارد. اين روش آسان ترين و مقرون به صرفه ترين روش است و طراحان بايد تلاش كنند حداكثر استفاده را از قابليت استفاده مجدد نمايند البته به شرطي كه ابتدا سرويسهاي پايه(واحدهاي سازنده) تعريف و پياده سازي شده باشند.

در اين راستا طراح سرويس، مدلي را كه شامل تعريف و توصيف سرويسهاي مورد نياز است به كمكزبان هاي موجود مدلسازي مينمايد، وي همچنين بايد طريقه پياده سازي اين سرويس ها را تجزيه و تحليل نموده و چگونگي ايجاد سرويس ها را براساس سه روش گفته شده، مشخص نمايد.

1. طراحي كيفيت سرويس

در اين گام بهترين سكو و فناوري براي پياده سازي و استقرار سرويس ها مد نظر قرار مي گيرد و شاخصهايي مانند قابليت اطمينان، امنيت، در دسترس بودن و كارائي مورد بررسي قرار گرفته و فرآيند طراحي و انتخاب سرويس ها تكميل ميشود. در اين راستا معمار يا طراح كيفيت سرويس جزئيات مقيد سازي سرويس ها و چگونگي پياده سازي و استقرار آنها را مشخص مي كند و روش تحقق شاخصهاي كيفيت سرويس را تعيين مي نمايد.

1. پياده سازي و استقرا سرويس ها

در اين گام بر حسب مدلهاي تهيه شده در گامهاي قبلي، سرويس ها پياده سازي و مستقر مي شوند. در پياده سازي بايست به موضوعاتي چون پيكربندي، مديريت نسخه ها و رضايتمندي از سرويس ها توجه شود. براي گام استقرار طبق قاعده ابتدا يك مرحله پيش ارائه وجود دارد و سپس استقرار نهائي صورت ميپذيرد، برآورده نمودن نيازهاي كيفيت سرويس نظير قابليت اطمينان، امنيت و كارائي نيز از وظايف اين فاز است. اگرچه تست به عنوان يك گام جداگانه در متدولوژي ديده نشده است اما بايد مورد توجه قرار گيرد، فعاليتهاي مربوط به تست و توليد بستههاي تست قبل از استقرار سرويس ها ضروري است.

1. اركستريشن سرويس ها

بعضي سرويس ها را ميتوان بدون پياده سازي و از طريق فراخواني ساير سرويس ها ايجاد نمود، اين فعاليت كاملا با فعاليتهاي مربوط به طراحي و پياده سازي متفاوت است، اولا چگونگي تعريف توالي فراخواني سرويس ها براي ايجاد سرويس جديد مستقل از موارد مربوط به طراحي است و دوما از نظر زماني تا سرويس هاي پايه طراحي و پياده سازي نشوند، تعريف سرويس هاي اركستريشني ممكن نخواهد بود. اگرچه از نظر هزينه و زمان اركستريشن سرويس ها بسيار مقرون بصرفه و مطلوب بوده و يكي از اهداف معماري سرويس گرا نيز ايجاد واحدهاي قابل استفاده مجدد مي باشد ولي اين گونه سرويس هاي تركيبي داراي مسائل خاص خود هستند چرا كه داراي ارتباط و وابستگي محكمي با ساير سرويس ها بوده و مي بايست موارد مهميمد نظر قرار گيرد بطوريكه سرويس گيرنده تفاوتي بين اين سرويسها و سرويس هاي پايه احساس نكند.

1. انتشار سرويس ها

بعد از اينكه طراحي و پياده سازي سرويسها پايان يافت نوبت به انتشار آنها مي رسد، براي اين منظور اين فعاليتها انجام مي شود: •فايل WSDL دراختيار سرويس گيرندگان قرار مي گيرد، در صورتيكه سرويس گيرندگان داخلي باشند اين فايل بصورت محلي در اختيار آنها قرار مي گيرد ولي در صورتيكه سرويس براي استفاده عمومي باشد، فايل ميبايست بر روي اينترنت قرار گيرد. •توصيفات فني و غير فني سرويسها بايد تعيين شود، دسته بندي هاي مختلفي از انواع سرويسها از نظر نوع سرويس و مشخصات فني وجود دارد، اين مشخصات بايد به گونه اي مستند و منتشر شود كه متقاضيان به راحتي بتوانند سرويس مورد نظر خود را شناسائي و فراخواني نمايند.

1. معاهده سطح سرويس

شامل توافق هاي بين منتشركنندگان و دريافت كنندگان سرويس است و جنبههاي مختلفي از جمله قابليت اطمينان، امنيت و كارائي را در بر مي گيرد. قابليت اطمينان: براي هر سرويس مي بايست معيارهاي قابل اندازه گيري براي قابليت اطمينان آن فراهم شود، براي نمونه دردسترس پذيري 99 درصدي سرويس يكي از مقياس هاي مناسب براي اين منظور است، يكي ديگر از اين مقياسها تضمين تحويل(دريافت) يكباره پيام است. امنيت: تصديق هويت و مجوزسنجي دو معيار اساسي در امنيت سرويسها هستند. تصديق هويت جهت بررسي مجوز عاملي است كه درخواست سرويس نموده و معمولا از طريق كنترل شناسه و رمز عبور انجام مي شود. مجوز سنجي كنترل كننده سطوح دسترسي و مجوزهائي است كه عامل تصديق هويت شده ميتواند داشته باشد. كارائي: يكي از معيارهاي كارائي كه اهميت بالائي نيز دارد، زمان پاسخ است و به مدت زماني گفته مي شود كه سرويس دهنده پس از دريافت درخواست، پاسخ مي دهد و معمولا برحسب ثانيه يا ميلي ثانيه است.

يكي ديگر از اين معيارها توان عملياتي است و به تعداد پاسخ گوئي يك سرويس دهنده در يك واحد زماني گفته ميشود، مانند 100 پاسخ در ثانيه.

1. مديريت و ديده باني سرويس ها

اين گام شامل مواردي چون كنترل و ديده باني كارائي سرويس ها، تحليل گزارشات و آمارها و هماهنگ كردنمستمر سرويس ها بر طبق معاهده سطح سرويس است.

1. ميزان كردن سرويس ها

اين گام به تنظيم سرويس ها در راستاي رفع نواقص و ارتقاء شاخص هاي كارائي اختصاص دارد، دادههاي مورد نياز براي اين منظور از گام قبل بدست آورده مي شود.

## مدل تکامل یافته معماری سرویس گرا

چندين مدل براي سنجش بلوغ معماري سرويس گرا منتشر شده كه در اينجا يكي از آنها مورد بررسي قرار مي گيرد، اين مدل شامل 5 سطح مي باشد : [9].

* سطح 1(سرويس هاي اوليه) : در اين مرحله سازمان اولين آشنايي با معماري سرويس گرا را تجربه نموده و مقدمات برپايي آن فراهم ميشود. تعدادي سرويس بصورت آزمايشي پياده سازي و مستقر مي شوند و پروتكل هاي اصلي چون SOAP ،WSDL مورد بررسي و استفاده آزمايشي قرار مي گيرند. هدف اين مرحله تحقيق و آزمايش بوده و در صورت دريافت نتيجه مطلوب مراحل بعدي آغاز مي شود.
* سطح 2(سرويس هاي معماري شده): اين مرحله كارائي و مزاياي پياده سازي كامل و يكپارچه معماري سرويس گرا در كاهش هزينه و زمان و بهبود انعطاف پذيري را اثبات ميكند. استانداردهاي

بيشتري در QoS مانند WS-Security و WS-Reliable Messaging مد نظر قرار ميگيرند، در اين زمان افسر ارشد اطلاعاتي مسئوليت برپاسازي معماري را بصورت رسمي به دست مي گيرد.

* سطح 3(سرويس هاي همكار): در اين مرحله اركستريشن سرويسها و پياه سازي سرويس هاي تركيبي شروع مي شود. انواع فرآيندها و پردازش ها(مبتني بر رويداد، محاسباتي، عملياتي و..) مورد بررسي قرار گرفته و به يكي از دو صورت پياده سازي مستقل يا تركيب از ساير سرويسها ايجاد مي شوند، هدف اين مرحله حداكثر استفاده از سرويسهاي پايه براي تعريف و ايجاد سرويسهاي جديد استو استانداردهائي چون BPEL مورد استفاده قرار ميگيرند.
* سطح 4(سرويس هاي اندازه پذير) : تمركز اين سطح از سطوح بلوغ بر شاخص هاي اندازه گيري كارائي حرفه بوده و ديده باني فعاليت هاي حرفه به عنوان يك اقدام كليدي مي بايست مورد توجه قرار گيرد. به دليل وارد شدن معيارهاي اندازه گيري كارائي در اين مرحله، حضور و نظارت نمايندگاني از مديريت مالي و برنامه ريزي سازمان ضروري خواهد بود.
* سطح 5(بهينه سازي سرويس ها) : در اين مرحله بهينه سازي سرويسهاي سازماني مورد نظر است وهدف آن دستيابي و محقق نمودن تكامل مستمر و تدريجي در سازمان با كمك افسر ارشد اجرائي(CEO) است.

## چرخه حيات معماري سرويس گرا

بر اساس طرح IBM، براي معماري سرويس گرا ميتوان يك چرخه حيات در نظر گرفت. در فاز »مدل« نيازمندي هاي كسب و كار جمع آوري شده و فرايندهاي كسب و كار آنها طراحي مي شود. بعد از بهينه شدن فرايندها، از طريق كنار هم قرار دادن سرويسهاي موجود و سرويسهاي جديد اين فرايندهاي كسب و كار شكل مي گيرد. سپس اين سرمايه ها در يك محيط امن و با قابليت تجميع بالا نصب ميشود. بعد از نصب فرايندهاي كسب و كار، كاربران IBM اين فرايندهاي كسب و كار را هم از منظر فني و هم از منظر فرايندهاي كسب و كار مورد نظارت و مديريت قرار ميدهند. اطلاعات جمع آوري شده در فاز مديريت به چرخه حيات بازخورد خواهد داشت تا بهبود پيوسته فرايندها را امكان پذير سازد. در زير همه اين مراحل در چرخه حيات، حاكميت و فرايندهايي هستند كه رهنمودها و افق هاي آينده را براي پروژه معماري سرويس گرا فراهم ميكنند.

**مرحله مدل سازي**

فاز مدل با جمع آوري و تحليل نيازمندي هاي كسب و كار آغاز مي شود كه بعداً براي مدل كردن، شبيه سازي و بهينه كردن فرايندهاي كسب و كار مورد استفاده قرار ميگيرند. فرايندهاي كسب و كار حاصل براي طراحي سرويس هاي نرم افزاري مرتبط و سطوح سرويس جهت حمايت از اين فرايندها مورد استفاده قرار ميگيرند. در طول اين فاز، مدلي جهت ايجاد درك مشترك بين كسب و كار و فناوري اطلاعات در فرايندهاي كسب و كار، اهداف و خروجي ها استفاده مي شود. به علاوه اين مدل مي تواند اين اطمينان را به وجود آورد كه كاربردهاي حاصل، نيازمندي هاي كسب و كار تعريف شده را براورده ميسازد. اين مدل همچنين ميتواند مبنايي جهت اندازه گيري كارآيي كسب و كار باشد.

**مرحله گردآوري**

در طول فاز گردآوري، كتابخانه سرويسهاي موجود مي تواند جهت يافتن سرويسهاي مورد نظر و موجود در سازمان بررسي شود. در صورتي كه سرويس مورد نظر يافت نشد اين امكان وجود دارد كه يك سرويس جديد ايجاد و پس از تست به مجموعه افزوده شود. هنگامي كه سرويسهاي مورد نياز فراهم شد، سرويس ها جهت پياده سازي فرايندهاي كسب و كار هماهنگ ميگردند.

**مرحله نصب**

در طول فاز پياده سازي، مقياس و محيط زمان اجرا جهت تامين نيازمنديهاي سطوح سرويس به وسيله فرايندهاي كسب وكار پيكربندي ميشود. پس از پيكربندي يك فرايند كسب وكار، امكان پياده سازي آن در يك محيط امن، مطمئن و مقياس پذير سرويسها وجود خواهد داشت. محيط سرويس ها به گونهاي بهينه سازي مي شود كه علاوه بر اجراي مطمئن فرايندهاي كسب وكار، امكان انعطاف پذيري جهت بروز كردن به طور پويا و در صورت تغيير نيازمنديهاي كسب وكار را فراهم مي آورد. اين رويكرد مبتني بر سرويس همچنين هزينه و پيچيدگي نگهداري سيستم را نيز كاهش مي دهد.

**مرحله مديريت**

فاز مديريت شامل نظارت و نگهداري از زمان پاسخ و در دسترس بودن سرويس مي شود. همچنين مديريتمنابع سرويس هاي زيرين در اين فاز انجام مي شود. درك كارايي زمان واقعي فرايندهاي كسب وكار امكان ايجاد بازخورد ضروري به مدل فرايند كسب و كار جهت بهبود دائمي را فراهم مي آورد. اين كار همچنين مديريت و نگهداري كنترل نسخه براي سرويس هاي تشكيل دهنده فرايندهاي كسب و كار را شامل مي شود.

فاز مديريت در نهايت امكان اتخاذ تصميمات كسب و كار بهتر و سريع تر را فراهم مي سازد.

**مرحله حاكميت و فرايندها**

حاكميت و فرايندها جهت موفقيت هر نوع پروژه معماري سرويس گرا ضروري هستند. جهت تخمين موفقيت، ممكن است يك مركز تعالي در كسب وكار، براي پياده سازي سياست هاي حاكميتي و دنبال كردن استانداردهاي حاكميتي بين المللي جهت اهداف كنترلي براي اطلاعات و تكنولوژي مرتبط ايجاد گردد. پياده سازي سياستهاي حاكميتي قوي ميتواند منجر به پروژه هاي معماري سرويس گرا موفق گردد.

**تشخيص سرويس ها**

تشخيص سرويس ها اولين گام در فاز مدلسازي در چرخه حيات معماري سرويس گرا است. فاز شناسايي سرويس ها يكي از بحراني ترين فازها در موفقيت پروژه معماري سرويس گرا است زيرا در اين فاز نيازمندي هاي حرفه شناخته شده و پوشش داده مي شوند.[1] هدف اين فاز توليد مجموعه اي از سرويسهاي كانديد براي پروژه سرويس گرا و مشخص كردن عمليات آنها است.[1] براي شناسايي سرويسهاي كانديد تحليل گر در مورد استفاده از راهبردها و روش هاي مختلف تصميم گيري مي كند.[2]

نقش هاي درگير در اين فار عبارتند از : معمار نرم افزار (نقش اصلي) و تحليل گر حرفه (نقش كمكي). در خلال اين فاز مدل كاري سرويسها توليد مي شود و در پايان فاز، اين مدل به معمار نرم افزار براي انجام

عمليات مربوط به فاز توصيف سرويس ها (Service Specification) تحويل داده مي شود.[1] اصلي ترين ورودي هاي اين فاز سرويس هاي موجودي كه سازمان به آنها (از هر طريقي مانند تملك، خريداري و ...) دسترسي دارد مي باشند. اين سرويس ها معمولا در انباره سرويس هاي سازمان موجود مي باشند. البته بايد توجه داشت كه در سازمان هايي كه اولين تجربه معماري سرويس گراي خود را انجام ميدهند ،طبيعتا اين ورودي وجود ندارد، و سرويس هاي تماما بايد توليد شده و پس از انجام فازهاي مختلف در انباره براي استفاده هاي در پروژه هاي سرويس گرايي آينده قرار بگيرند.[1]

**توصيف سرويس ها**

پس از شناسايي مجموعه سرويس هاي كانديد فاز توصيف سرويس ها آغاز مي گردد. اين فاز توسط معمار نرم افزار به همراه طراحان ( به صروت اختياري ) انجام مي شود.[1] هدف اين فاز توصيف كامل عناصر طراحي معماري سرويس گرا است.

فاز توصيف سرويس ها را مي توان وظيفه مربوط به معماري در فرآيند معماري سرويس گرا دانست. اين ديد اهميت اين فاز را به خوبي نمايان مي سازد.[1]

براي روشن تر شدن تمايز ميان فاز توصيف سرويس ها و شناسايي سرويس ها مي توان فاز شناسايي سرويس ها را (Service Identification) را به عنوان فعاليت هاي مربوط به تحليل مدل سرويس و فاز توصيف سرويس ها (Service Specification) را به عنوان فعاليت هاي مربوط به طراحي مدل سرويس دانست.

در خلال فاز توصيف سرويس ها مدل سرويس به روشني مشخص مي شود و در پايان اين فاز، اين مدل به طراحان براي انجام فاز عينيت بخشيدن به سرويس ها و توسعه دهندگان براي پياده سازي سرويس ها داده مي شود.[1]

وظايف اصلي درون اين فاز عبارتند از: تحليل زيرسيستم ها (Subsystem Analysis)، توصيف مؤلفه ها (Component Specification) و تخصيص سرويس ها (Service Allocation). در اين فاز ويژگي ها و قابليت هاي سرويس ها تشريح مي گردند و به سؤالهايي از جمله : سرويس چه خدمتي ارائه مي دهد؟ منطق داخلي سرويس ها چيست؟ نيازمندي هاي غير وظيفه مندي هر سرويس چيست؟ و... جواب داده مي شود.

**تحليل زيرسيستم ها**

ورودي اين مرحله زيرسيستم هايي (سرويس ها) كه از مرحله تجزيه دامنه بدست آمده اند مي باشند و در اينمرحله وابستگي بين اين زير سيستم ها و جريان هاي بين آنها مشخص مي گردد.[3]

در تحليل زيرسيستم ها مدل هاي اشياء به منظور مشخص كردن نحوه انجام كار در داخل زير سيستم و همچنين نمايش زير سيستم هاي درون زير سيستم در حال بررسي (سلسله مراتب زيرسيستم ها) توليد مي شوند.

ساختار طراحي زير سيستم ها براي توليد ساختار پياده سازي مؤلفه ها – كه خود اساسي براي پياده سازي سرويس ها هستند – استفاده مي شود.[3]

در اين مرحله خصوصيات سرويس ها از جمله : وابستگي بين آنها، نحوه تركيب آنها، پيام هاي بين آنها ، فاكتور هاي كيفيت آنها و... تعيين مي شوند.براي اين كار 6 مرحله زير انجام مي گردند.

1. : *Litmus* تست

اين تست به اين سؤال كه "از ميان سرويس هاي كانديد كداميك براي پياده سازي مناسب است؟" پاسخ مي دهد. بايد توجه داشت تمامي سرويس هايي كه در مرحله تجزيه دامنه مشخص شدند براي پياده سازي مناسب نيستند. براي پيدا كردن زير مجموعه اي از اين سرويس ها در روش تست Litmus براساس موارد مختلف از جمله تطابق سرويس با نيازهاي حرفه، تكراري نبودن سرويس و ... تصميم گيري مي شود. منظور از تكراري نبودن سرويس اين است كه وظايفي كه سرويس در دست بررسي انجام مي دهد توسط سرويس هاي ديگر پوشش داده نشوند.

1. .مشخص كردن وابستگي ميان سرويس ها :بازبيني جزئيات سرويس ها باعث مشخص شدن وابستگي سرويس به سرويس هاي ديگر و نحوه تعامل ميان سرويس ها مي شود.
2. مشخص كردن تركيب سرويسها و جريان ميان آنها: با بررسي حيطه هاي كاري و فرآيندهاي حرفه نحوه توليد سرويس ها از تركيب سرويس هاي ديگر و جريان كاري ميان سرويس ها براي انجام فرآيندهاي حرفه را مشخص مي كند.
3. مشخص كردن نيازهاي غيروظيفه مندي :با استفاده از نيازهاي غير وظيفه مندي ، ويژگي هاي كيفي سرويس ها مشخص مي گردد.
4. مشخص كردن پيام هاي سرويس :در اين قسمت قالب پيام هاي ورودي و خروجي سرويس و محتواي آنها مشخص مي شود.
5. مستندسازي تصميمات مديريتي :برخي مواقع تركيب سرويس ها منجر به برخي تصميمات مديريتي درحالت هاي مختلف كاري سرويس مي شود. اين تصميمات مديريتي در اين مرحله مستندسازي مي شوند.

**تخصيص سرويسها**

پس از مشخص كردن سرويس ها و جزئيات آنها در اين مرحله درباره تخصيص سرويس ها به مؤلفه ها تصميم گيري مي كنيم.[2] و [3] اغلب اين تخصيص يك-به-يك است.

همچنين در اين مرحله نحوه تخصيص سرويس ها و مؤلفه به لايه هاي معماري سرويس گرا مشخص مي شود.[3] براي تصميم گيري در مورد چگونگي تخصيص سرويس ها و مؤلفه ها به لايه هاي معماري علاوه بر توجه به معماري نرم افزار بايد به معماري محيط فني و عملياتي كه سيستم در آن اجرا مي شود توجه نمود.[3]

**توصيف مؤلفه ها**

در مرحله بعد از فاز توصيف سرويس ها ، جزئيات مؤلفه هايي كه سرويس ها را پياده سازي مي كنند مشخص مي شود.[3] اين توصيف دربرگيرنده جزئياتي در مورد داده هاي مؤلفه، قوانين، سرويس ها و نحوه پيكربندي مؤلفه مي باشد. همچنين در اين مرحله پيام هاي مؤلفه ها و خصوصيات رويدادهاي آنها مشخص مي گردد.

[3]

پس از مشخص كردن خصوصيات سرويس ها و نحوه اجراي سرويس ها توسط مؤلفه ها و همچنين تعريف جزئيات مؤلفه ها، بايد در مورد نحوه توليد و در اختيار گرفتن سرويس ها تصميم گيري شود.

**عينيت بخشي به سرويس ها**

در اين فاز در مورد مباحث مربوط به "امكان سنجي فني" پياده سازي يا استخراج سرويس ها از سيستم هاي موروثي تصميم گيري مي كند.[3] و [8] گزينه هاي ديگري كه در اين زمينه وجود دارد عبارتند از تجميع سرويس ها و توليد سرويس مورد نياز، تغيير سرويس هاي قديمي براي توليد سرويس مورد نياز ، سفارشسرويس و ... .[3]

تصميمات ديگري كه در اين فاز گرفته مي شود – علاوه بر تصميمات در مورد وظيفه مندي – عبارتند از :

امنيت، مديريت و نظارت بر سرويس ها. [3]

## 2.2 پارامتر های مهم کیفی

پس از آشنايي با مفاهيم موجود در زمينه معماري سرويس گرا و فاز شناسايي سرويس ها، در اين بخش با معيارهاي كيفي در سرويس ها بيشتر آشنا مي شويم.

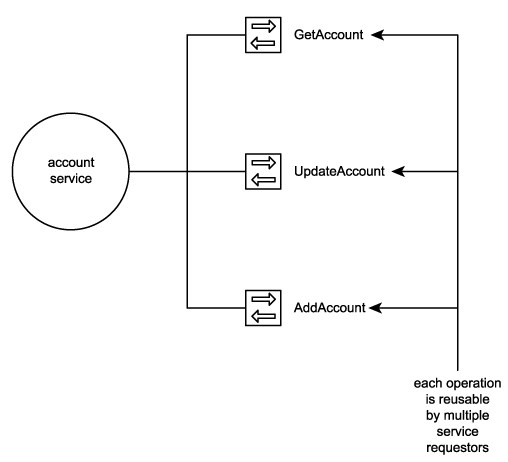
از آنجا كه گام شناسايي سرويس ها يكي از پر اهميت ترين گام ها در چرخه معماري سرويس گرا به شمار مي رود، آشنايي با برخي معيارها در مورد كيفيت سرويس ها و سنجش سرويس هاي معرفي شده طي مراحل اين فاز بر اساس اين معيارها مي تواند تا حدي ديد درستي نسبت به كيفيت فرآيندهاي شناسايي سرويس ها و پروژه سرويس گرايي در اختيار بگذارد. به عبارت ديگر به علت اهميت گام تشخيص سرويس ها، بايد به نحوي از كيفيت خروجي هاي اين گام كه همان سرويس هاي كشف شده در اين گام هستند، شناخت پيدا كرد. استفاده از معيارهاي كيفي به همين منظور انجام مي گيرد.

بايد توجه داشت كه معيارهايي كه در ادامه معرفي مي شوند، تمامي معيارهاي معرفي شده در زمينه سنجش كيفيت سرويس ها نبوده بلكه معيارهاي پر اهميت در اين زمينه مي باشند. همچنين نياز نيست تمامي سرويس هاي كشف شده در فاز تشخيص سرويس ها تمامي معيارهاي زير را به تمامي داشته باشند.

* قابليت استفاده مجدد (reusability)

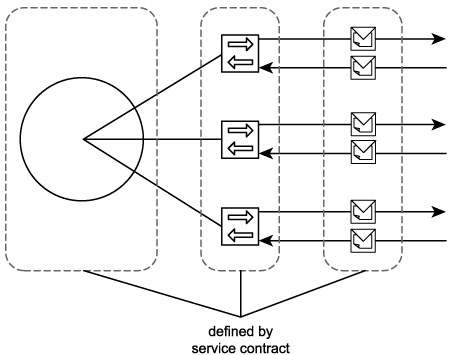
يكي از اصلي تريم معيارها در تشخيص سرويس ها قابليت استفاده مجدد سرويس ها مي باشد. همانطور كه پيش از اين گفته شد، سرويس ها در ادامه ايده IC هاي نرم افزاري بوجود آمدند. اصلي ترين ايده در اين زمينه استفاده مجدد از اين ICهاي نرم افزاري به منظور توليد سيستم هاي پيچيده گوناگون است. همانطور كه با تركيبات مختلف IC هاي سخت افزاري مي توان مدارهاي گوناگون و متنوعي را توليد كرد، استفاده از اين IC هاي نرم افزاري نيز مي توانند توليد سيستم هاي نرم افزاري مختلف و گوناگوني را موجب گردند.

اين قابليت باعث كاهش چشمگير هزينه و زمان توليد سيستم هاي نرم افزاري مي گردد.



**شكل شماره 2 -1 - قابليت استفاده مجدد [1]**

قراردادهاي سرويس مناسب برخي اطلاعات مفهومي در مورد شيوه انجام عمليات توسط سرويس ها را نيز در اختيار قرار مي دهند. اين اطلاعات هنگام توافق ميان فراهم كننده سرويس و متقاضي آن مورد استفاده قرار مي گيرد.



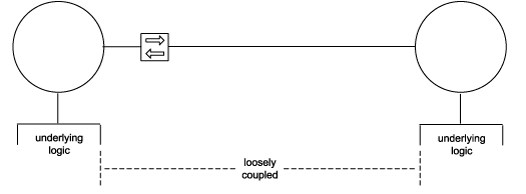
**شكل شماره 2 -2 - قرارداد مشترك [1]**

* وابستگي كم ميان سرويسي (loosely coupled)

به اين علت كه حيطه وظيفه مندي حرفه ها و سازمان ها به صورت پويا تغيير مي كند، و سيستم هاي نرم افزاري اين سازمان ها موظف به پاسخگويي به نيازهاي سازمان هستند، در نتيجه سيستم هاي نرم افزاري بايد پاسخگوي به تغييرات در وظيفه مندي هاي مورد نياز باشند.

به همين علت توسعه سيستم ها به گونه اي كه در برابر تغييرات قابليت انعطاف پذيري بالايي داشته باشند يكي از دغدغه هاي اصلي در حيطه طراحي . توسعه سيستم هاي نرم افزاري است. اين امر باعث كاهش هزينه و زمان مورد نياز جهت اضافه كردن قابليت هاي جديد به سيستم هاي نرم افزاري مي گردد.

يكي از راههاي دستيابي به اين ويژگي در معماري سرويس گرا، توسعه سيستم ها به نحوي كه ارتباط محدود با يكديگر داشته باشند است. به عبارت ديگر در معماري سرويس گرا به منظور دستيابي به انعطاف پذيري بالا سعي مي شود سيستم از سرويس هايي تشكيل شود كه داراي ارتباطات كم و محدود با يكديگر باشند. اين امر سبب مي شود در صورت نياز به تغيير يك سرويس، سرويس هاي كمتري به منظور ارتباط با آن دستخوش تغيير شوند . همين امر انعطاف پذيري سيستم را افزايش داده و هزينه و زمان مورد نياز براي پاسخگويي به تغييرات را كاهش مي دهد.

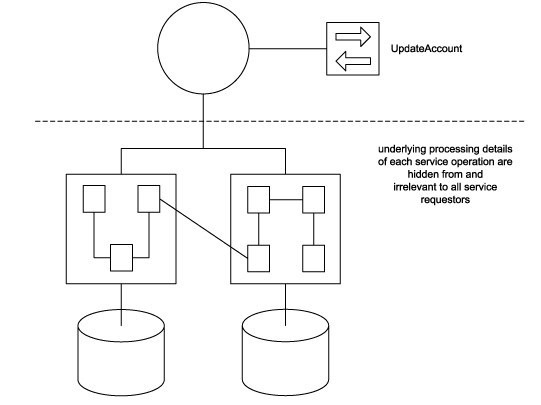


**شكل شماره 2 -3 - وابستگي كم ميان سرويسي [1]**

* (abstraction) تجريد

يكي از دلايلي كه از معماري سرويس گرا عمتدا در توسعه سيستم هاي مقياس وسيع استفاده مي شود، اين است كه در اين سيستم ها به علت گشتردگي حيطه مسئله با موجوديت ها و ارتباطات بسيار زيادي مواجه هستيم. در صورتي كه در توسعه اين سيستم ها بخ.اهيم از روش هاي سنتي استفاده كنيم به علت مواجه با حجم زياد موجوديت ها و ارتباطات دچار سردرگمي خواهيم شد. به همين علت براي كاهش پيچيدگي در اين سيستم ها از موجوديتي به نام سرويس به منظور بالا بردن سطح تجريد و در نتيجه كاهش پيچيدگي استفاده مي شود.

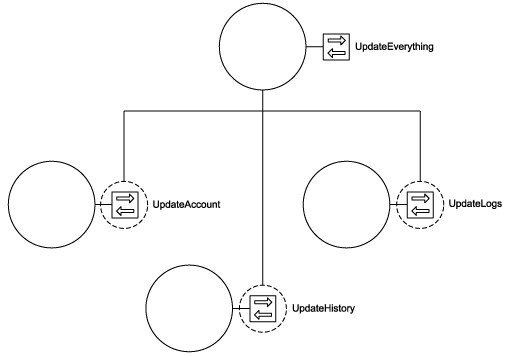
سرويس ها به صورت جعبه سياه هايي عمل مي كنند كه محتويات درونشان را از جهان خارج پنهان مي سازند. اين امر موجب كاهش موجوديت ها و ارتباطات آشكار در حيطه مسئله شده و پيچيدگي را به ميزان چشمگيري كاهش مي دهد. به همين علت توصيه مي گردد سرويس ها از سطح تجريد بالايي برخوردار باشند.



**شكل شماره 2 -4 – تجريد [1]**

* قابليت تركيب (composable)

سرويس ها هر كدام وظيفه مندي خاصي را پوشش مي دهند. يكي از ويژگي هاي اصلي سرويس ها قابليت تركيب مي ياشد. اين امر به اين خاطر است كه گاهي نياز است به وظيفه مندي جديدي كه حاصل از تركيب وظيفه مندي هاي پوشش داده شده توسط دو يا چند سرويس است پاسخ داده شود. به اين منظور مي توان از تركيب اين سرويس ها استفاده كرد. اين ويژگي باعث افزايش قابليت استفاده مجدد سرويس ها شده و هزينه و زمان توسعه سرويس هاي جديد را به صورت قابل ملاحظه اي كاهش مي دهد.

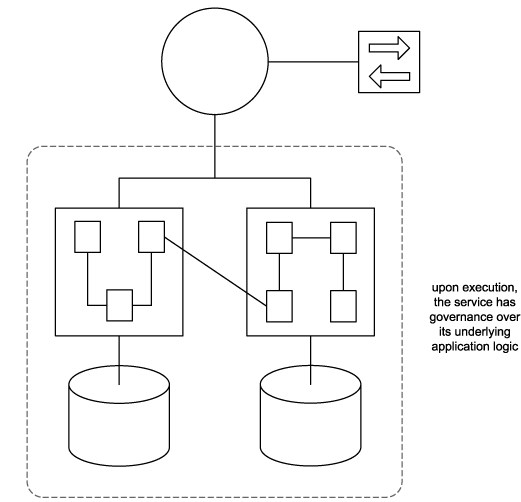


**شكل شماره 2 -5 - قابليت تركيب [1]**

* (autonomous) خودمختاري

خودمختاري سرويس ها نيازمند آن است كه محدوده سرويس ها به خوبي تعريف شده باشد. اين ويژگي باعث كاهش وابستگي سرويس به ديگر سرويس ها شده و سطحي از استقلال را براي سرويس ها فراهم مي آورد. خودمختاري سرويس ها جزو اولين مواردي است كه حين تجزيه دامنه مسئله به سرويس ها بايد در نظر گرفته شود.

البته بايد توجه داشت خودمختاري سرويس به اين معنا نيست كه سرويس در حيطه وظيفه مندي خود اختيار تام دارد بلكه به اين معناست كه سرويس در هنگام اجرا در حيطه وظيفه مندي اش داراي اختيار تام است.

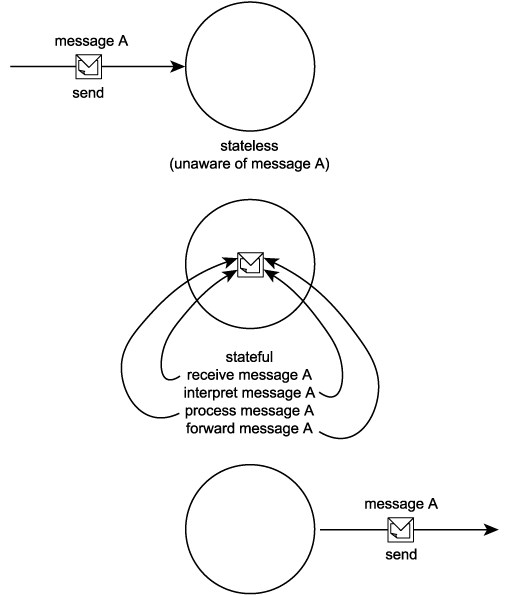


**شكل شماره 2 -6 – خودمختاري [1]**

* نداشتن وضعيت خاص (stateless)

سرويس ها بهتر است ميزان و يا مدت زمان نگهداري اطلاعات حالت خود را تا حد ممكن كاهش دهند.

هنگامي كه به عنوان مثال يك سرويس در حال پردازش يك پيام (message) است، سرويس اصطلاحا داراي حالت مشخص (state full) مي گردد.



**شكل شماره 2 -7 - نداشتن حالت مشخص [1]**

البته ترجيح داده مي شود سرويس ها وضعيت مشخص نداشته باشند (stateless). اين امر ويژگي هاي قابليت استفاده مجدد (reusability) و مقياس پذيري (scalability) سرويس ها را افزايش مي دهد.

## 3.2 بررسي فاكتور دانه بندي سرويس ها

در قسمت قبل با ويژگي هاي اصلي سرويس ها آشنا شديم. اين ويژگي ها در اكثر متون مربوط به سرويس ها آورده شده اند و تمامي روش هاي ارائه شده در زمينه تشخيص سرويس ها (service identification) بر اساس برآورده كردن تعدادي از اين ويژگي ها بنا نهاده شده اند.

مسئله ديگري كه در معماري سرويس گرا بر كيفيت سيستم توليد شده اثر زيادي مي گذارد و در روش هاي ارائه شده كمتر به آن توجه شده است، فاكتور دانه بندي سرويس ها مي باشد. اين بدان معني است كه اگر سرويس ها بزرگ و يا كوچك باشند بر برخي از ويژگي هاي سيستم نهايي تاثير مستقيم مي گذارد. در زير اثر اين ويژگي بر چهار ويژگي اصلي سيستم هاي نرم افزاري بررسي مي گردد.

* (flexibility) انعطاف پذيري

اگر سرويس ها بزرگ باشند براي تغيير در يكي از جنبه ها يا وظيفه مندي هاي سيستم بايد دامنه گسترده اي از سيستم دستخوش تغيير شود. به همين علت بزرگي اندازه سرويس ها موجب كاهش انعطاف پذيري سيستم مي گردد. در مقابل هر چه سرويس ها اندازه كوچكتري داشته باشند، براي تغيير در قسمتي از سيستم تنها تغيير در حيطه سرويس هدف كفايت مي كند و به همين علت تغيير جنبه اي از سيستم سربار زيادي را ناشي نمي شود و در نتيجه انعطاف پذيري سيستم افزايش مي يابد.

* قابليت استفاده مجدد (reusability)

قابليت استفاده مجدد يكي از بارزترين و اصلي ترين ويژگي ها در معماري سرويس گرا به شمار مي رود.

همانطور كه پيش از اين اشاره شد، قابليت استفاده مجدد باعث كاهش زمان و هزينه توليد سيستم هاي نرم افزاري شده و تا حد ممكن در توسعه سيستم هاي نرم افزاري تلاش در افزايش اين قابليت داريم.

دانه بندي سرويس ها بر اساس حيطه وظيفه مندي سرويس ها تعيين مي شود. به عبارت ديگر هرجه سرويسي حيطه بزرگتري از وظيفه ندي را پوشش دهد ، اصطلاحا ـن سرويس را بزرگتر و هر چه حيطه كوچكتري را پوشش دهد، اصطلاحا سرويس را كوچكتر گويند. در صورتي كه سرويس بزرگ باشد، حيطه بزرگتري از وظيفه مندي هاي حرفه را پوشش مي دهد و اصطلاحا سرويس در حيطه حرفه تنظيم شده و وابستگي بيشتري به حرفه خواهد داشت. همچنين به علت تفاوت در ماهيت حرفه هاي گوناگون امكان استفاده مجدد از آن سرويس در حرفه هاي ديگر كاهش مي يابد و باعث كاهش قابليت استفاده مجدد سرويس مي گردد.

در مقابل هرچه سرويس كوچكتر باشد، وابستگي كم تري به حرفه داشته و به احتمال زياد سرويس مطابق با وظيفه مندي خاصي طراحي شده است. به علت امكان بروز وظيفه مندي هاي مشابه در حيطه حرفه هاي گوناگون امكان استفاده مجدد سرويس افزايش مي يابد.

* (performance) كارآيي

از آنجا كه براي برآورده كردن فرآيندهاي حرفه تعدادي از سرويس ها با هم همكاري مي نمايند، ارتباطات بين سرويس ها به اين منظور نوعي سربار تلقي مي گردند. به عبارت ديگر هنگام فراخواني سرويس ها به منظور برآورده كردن فرآيند حرفه، مقداري از كارآيي سيستم جهت برقراري ارتباط ميان سرويس هاي مختلف هدر مي رود. حال در صورتي كه سرويس ها كوچك باشند، براي برآورده كردن فرآيندهاي حرفه به تعداد بيشتري سرويس نياز است و در نتيجه ارتباطات ميان سرويسي افزايش يافته و سربار بيشتري به سيستم تحميل مي گردد. در نتيجه استفاده از سرويس هاي كوچك باعث كاهش كارآيي سيستم مي گردد.

در مقابل هر چه سرويس ها بزرگتر باشند، براي پوشش نيازمندي هاي يك فرآيند حرفه به تعداد سرويس كمتري نياز بوده و در نتيجه سرباز ناشي از ارتباطات ميان سرويسي كاهش مي يابد. همين امر باعث افزايش كارآيي سيستم نهايي مي گردد.

* (complexibility) پيچيدگي

آخرين فاكتوري كه دانه بندي سرويس ها بر آن اثر مي گذارد، فاكتور پيچيدگي سيستم مي باشد. تاثير دانه بندي سرويس ها بر اين فاكتور تا حدودي شبيه تاثير دانه بندي بر فاكتور كارآيي سيستم مي باشد. در صورتي كه سرويس ها كوچك باشند، به تعداد بيشتري سرويس جهت پوشش دامنه حرفه نياز داريم. طبيعي است كه اين سرويس ها نياز به تعامل با يكديگر دارند. همين امر باعث مي شود در صورت داشتن سرويس هاي كوچك پيچيدگي سيستم به علت تعداد زياد ارتباطات افزايش يابد.

در سمت مقابل هرچه اندازه سرويس ها بزرگ تر باشد، به تعداد سرويس كمتر و در نتيجه ارتباطات كمتر ميان سرويسي نياز خواهد بود. همين امر پيچيدگي سيستم نهايي را كاهش خواهد داد.

با توجه به موارد فوق مشاهده مي شود نمي توان با قاطعيت عنوان كرد سرويس هاي بزرگتر بهتر هستند يا سرويس هاي كوچك تر. بلكه بايد با توجه به نيازمندي هاي حرفه و ويژگي هاي دامنه مسئله با مصالحه بين فاكتورهاي انعطاف پذيري، قابليت استفاده مجدد، كارآيي و پيچيدگي اندازه مناسبي را براي سرويس ها تعيين نمود.

**نتيجه گيري**

به منظور دستيابي به درك يكسان از دامنه مسئله و مفاهيم موجود در آن، در اين فصل براي مفاهيم موجود در حيطه سرويس گرايي و مسائل مربوط به آن از جمله سرويس، مؤلفه، معماري سرويس گرا، متدولوژي هاي سرويس گرا، استانداردها و پروتكل ها، مدل بلوغ معماري سرويس گرا و چرخه حيات معماري سرويس گرا و فازهاي آن تعريفي يكسان ارائه شد. اين امر باعث خواهد شد در ادامه مطلب ابهامي در خصوص مفاهيم مطرح شده وجود نداشته باشد.

در ادامه نيز برخي معيارهاي كيفي به منظور سنجش كيفيت سرويس هاي كشف شده در روش هاي تشخيص سرويس ها معرفي گرديدند و اهميت هر يك ذكر شد. در آخر هم فاكتور اندازه سرويس ها به عنوان فاكتوري كه بر برخي ويژگي هاي سيستم نهايي تاثير گذاشته اما كمتر مورد توجه است مورد بررسي قرار گرفت.

در فصول آينده روش هاي ارائه شده در زمينه تشخيص سرويس ها معرفي شده و ويژگي هاي هريك بيان خواهند شد. همچنين هر كدام از اين روش ها بر اساس معيار هاي معرفي شده در اين بخش سنجيده شده و ارزيابي خواهند گرديد و نقاط قوت و ضعف آن ها بيان خواهند شد.

سپس با توجه به نقاط ضعف موجود در راه حل هاي كنوني، اصول راه حلي جهت پوشش نقاط ضعف موجود بيان شده و ويژگي هاي اين راه حل عنوان خواهند گرديد.

# فصل سوممعرفي روش هاي گذشته

**مقدمه**

با توجه به گسترش چشمگير استفاده از معماري سرويس گرا در توسعه سيستم هاي نرم افزاري مقياس وسيع و همچنين اهميت فاز مدلسازي در چرخه حيات معماري سرويس گرا، روش هاي مختلفي به اين منظور ارائه شده است. ار آنجا كه گام تشخيص سرويس ها به عنوان اولين گام در مدلسازي سرويس گرا و چرخه حيات معماري سرويس گرا مطرح است و وابستگي ساير گام ها به خروجي اين گام، گام تشخيص سرويس ها (service identification) از اهميت ويژه اي برخوردار است.

در فصل اول با اهميت گام تسخيص سرويس ها و جايگاه آن در چرخه حيات معماري سرويس گرا آشنا شديم. بديهي است با توجه به اهميت اين گام روش هاي گوناگوني جهت انجام آن ارائه شده باشد. در فصل دوم نيز پس از آشنايي با مفاهيم موجود در زمينه معماري سرويس گرا، با برخي فاكتورهاي كيفي جهت سنجش كيفيت سرويس ها آشنا شديم. در اين فصل قصد داريم با روش هاي ارائه شده در زمينه تشخيص سرويس ها آشنا شده و با توجه به فاكتورهاي ارائه شده در فصل دوم به ارزيابي اين روش ها بپردازيم. در آخر نيز نقاط قوت و ضعف هر يك از اين روش ها را تا حد امكان شرح خواهيم داد.

## 1.3 دسته بندي روش هاي موجود

در مقاله اي كه در سال 2007 توسط Art Ligthart ،Jan-Willem Hubbers و Linda Terlouw در SOA Magazine ارائه شد [4] روش هاي تشخيص سرويس ها (service identification) در 10 رده كلي دسته بندي شدند. اين رده ها بر اساس ويژگي هاي كلي اين روش ها تعيين شده و همچنين نقاط قوت و ضعف هر يك از رده ها به صورت كلي بيان شده است. در جدول زير اين گروه ها به اختصار بيان گرديده اند:

|  |  |  |
| --- | --- | --- |
| **معايب** | **مزايا** | **روش** |
| كاهش استفاده مجدد، بروز افزونگي | تطابق با نيازهاي حرفه | تجزيه فرآيندهاي حرفه |
| كاهش استفاده مجدد | جلوگيري از افزونگي | وظيفه هاي حرفه |
| نياز به تحليل مجدد براي كشف تمامي اشياء و سرويس ها | كاهش تغييرات طراحي در آينده | اشياء موجوديتي حرفه |
| روش وابسته | شفافيت مالكيت سرويس ها | مالكيت و مسئوليت |
| كاهش استفاده مجدد، بروز افزونگي | تطابق با سياست هاي حرفه | هدف گرايي |
| مشكلات ناشي از تفاوت در ماهيت مولفه و سرويس | سادگي، تشخيص سرويس هاي جديد براي اهداف ناشناخته | مبتني بر مولفه |
| مشكلات ناشي از برنامه هاي با | كاهش زمان، كاهش هزينه | دارايي موجود (پايين به بالا) |
| طراحي ضعيف |  |  |
| مشكلات ناشي از برنامه هاي با طراحي ضعيف | سرعت، استفاده مجدد | Front-Office Application  Usage Analysis |
| كاهش تجريد | كاهش هزينه ها | زيرساخت |
| روش وابسته | توجه به نيازهاي كيفي | نيازهاي غيروظيفه اي |

**جدول شماره 3-1 - رده بندي روش هاي ارائه شده [4]**

در زير هر يك از روش هاي فوق به اختصار توضيح داده شده اند.

* تجزيه فرآيندهاي حرفه

در روش هايي كه از اين ايده استفاده مي كنند، ابتدا تمامي فرآيندهاي حرفه شناسايي شده و سپس به تجزيه هر يك از اين فرآيندها پرداخته مي شود. در اين روش پس از تجزيه فرآيندهاي حرفه به زير فرآيندها و وظايف (task) پس از رسيدن به سطح قابل قبولي از دانه بندي، سرويس ها شناسايي مي گردند.

مزيت اصلي اين روش اين است كه به علت تمركز بر فرآيندهاي حرفه، سرويس هاي شناسايي شده با فرآيندهاي درون حرفه تطابق دارند.

نقطه ضعف اين روش در اين است كه به علت تمركز بر فرآيندهاي يك حرفه خاص، امكان استفاده مجدد از سرويس هاي شناسايي شده، به علت تفاوت در ماهيت حرفه ها، كاهش مي يابد. همچنين به اين علت كه در روش هايي كه از اين منطق استفاده مي كنند فرآيندهاي حرفه به صورت جداگانه مورد بررسي قرار مي گيرند، امكان بروز افزونگي بوجود مي آيد. زيرا ممكن است يك وظيفه مندي خاص در يك فرآيند حرفه ديده شده باشد و در سرويسي پوشش داده شده باشد، اما در فرآيند ديگري نيز آن را در سرويس ديگري پوشش دادهباشيم. اين مسئله سبب مي شود امكان رويهم افتادگي در وظيفه مندي سرويس ها بوجود آيد و افزونگي بروزكند.

* وظيفه هاي حرفه

اين روش تا حد زيادي شبيه روش تجزيه فرآيندهاي حرفه است اما تفاوت اين دو روش در اين است كه در اين روش فرآيندهاي حرفه به صورت جداگانه بررسي نمي شوند بلكه ابتدا مدلي از فرآيندهاي حرفه بوجود مي آيد كه در آن تمامي فرآيندهاي حرفه ديده مي شوند سپس مدل فرآيندهاي حرفه به منظور يافتن سرويس ها مورد تجزيه و تحليل قرار مي گيرد.

اين روش علاوه بر مزيت روش تجزيه فرآيندهاي حرفه، از بروز افزونگي و رويهم افتادگي سرويس ها جلوگيري مي كند و با استفاده از مدل فرآيندهاي حرفه تا حد زيادي مديريت وظيفه مندي سرويس ها را مديريت مي نمايد. البته بايد توجه داشت مزيت روش تجزيه فرآيندهاي حرفه بر اين روش سادگي و سربار كمتر است زيرا در اين روش براي شناسايي سرويس ها يك مرحله اضافي نسبت به روش پيش يعني توليد مدل فرآيندهاي حرفه را بايد انجام داد كه خود موجب بروز افزونگي در روال شناسايي سرويس ها مي گردد.

نقطه ضعف اين روش نيز مانند روش قبل كاهش استفاده مجدد سرويس هاست كه به علت تمركز بر روي فرآيندهاي يك حرفه خاص بوجود مي آيد.

* اشياء موجوديتي حرفه

در اين روش با استفاده از روش هايي مانند استفاده از ماتريس CRUD و اشياء موجوديتي حرفه(business entity)، مي توان سرويس هاي مورد نياز را شناسايي نمود در اين روش با استفاده از اين ماتريس ها، اشياء موجوديتي حرفه و فرآيندهاي مقدماتي حرفه (elementary business elements) مي توان سرويس هاي كانديد را بر اساس روابط مشخص ميان اي اشياء و فرآيندها تشخيص داد. در [3] از اين روش به منظور تشخيص سرويس ها استفاده شده است.

ازمزاياي اين روش اين است كه با توجه به موجوديت هاي دامنه حرفه، امكان بروز تغييرات در سرويس ها در آينده كاهش مي يابد زيرا معمولا در دامنه حرفه هاي مختلف موجوديت هاي جديدي بوجود نمي آيند وهمين امر پايداري مجموعه سرويس هاي شناسايي شده با اين روش را افزايش مي دهد.

البته در اين روش در صورتي كه يكي از اشياء موجوديتي حرفه با ساير موجوديت ها ارتباطي نداشته باشد، درنظر گرفته نمي شود و لذا در اين روش به يك مرحله بازبيني جهت يافتن اينگونه اشياء نياز است. همين امر به عنوان عيب اصلي اين روش ها ذكز شده است.

* مالكيت و مسئوليت

در اين روش براي جلوگيري از بروز افزونگي و رويهم افتادگي، پس از اينكه با استفاده از يكي از روش هاي ديگر برخي سرويس هايي كانديد شناسايي شدند، حيطه هر يك از سرويس ها به دقت تعيين مي گردد.

همانطور كه از تعريف اين روش مشخص است حسن آن در تعيين دقيق مرز سرويس ها و وظيفه مندي آنهاست اما عيب اين روش وابسته بودن به ساير روش هاست. زيرا همانطور كه پيش از اين گفته شد، در اين روش سرويس هاي كانديدي كه توسط ساير روش ها شناسايي شده بودند را به منظور تعيين دقيق حيطه وظيفه مندي پالايش مي كنيم و حال اگر در روش اوليه نقص و كاستي وجود داشته باشد، اين خطا در خروجي اين روش نيز بروز خواهد كرد.

* هدف گرايي

در اين روش برخلاف روش تجزيه فرآيندهاي حرفه، به فرآيندهاي حرفه توجه نمي كنيم بلكه به اهداف مورد نظر اين فرآيندها ويا به عبارت ديگر به اهداف مورد نظر حرفه توجه مي نماييم. روش كار در اين رده از تكنيك ها در اين است كه ابتدا اهداف حرفه شناسايي شده و سپس اين اهداف به منظور رسيدن به دانه بندي مناسب به زير اهداف تجزيه مي گردند. پس از رسيدن به دانه بندي مناسب، هر كدام از زير اهداف به يك سرويس جهت برآورده كردن آن هدف متناظر مي گردند.

مزيت اين روش تطبيق كامل با سياست هاي حرفه است. اين بدان معني است كه چون در شناسايي سرويس ها اهداف و سياست هاي حرفه را مد نظر قرار داده ايم، خروجي هاي اين روش بر سياست هاي حرفه كاملا تطبيق خواهند داشت و سياست هاي حرفه را به طور كامل پوشش مي دهند.

نقطه ضعف اين روش نيز مانند روش تجزيه فرآيندهاي حرفه و ساير روش هاي متمركز بر ماهيت حرفه، كاهش قابليت استفاده مجدد است و اين امر به علت تفاوت در ماهيت حرفه هاي گوناگون با يكديگر ميباشد. همچنين به علت تمركز بر اهداف و عدم توجه به وظيفه مندي هاي مورد نياز امكان بروز افزونگي ورويهم افتادگي ميان سرويس ها رد اين روش وجود دارد.

* مبتني بر مؤلفه

در اين روش از روش هاي شناسايي مؤلفه ها استفاده مي شود. با توجه به بلوغ نسبي در روش هاي شناسايي مؤلفه ها در اين روش ابتدا با يكي از اين روش ها مؤلفه هاي موجود در دامنه حرفه شناسايي مي شوند و سپس اين مؤلفه ها براي تبديل به سرويس ها مورد پالايش قرار مي گيرند.

اصلي ترين مزيت اين روش سادگي آن است. زيرا با يكي از روش هاي شناسايي مؤلفه ها مي توان مؤلفه ها را به آساني شناسايي نمود و سپس اين مؤلفه هاي شناسايي شده را به سرويس ها تبديل نمود. همچنين در اين روش به علت اينكه دامنه مسئله دو بار مورد پالايش قرار مي گيرد، برخي سرويس هاي ناشناخته را نيز مي توان كشف نمود و اين امر نيز به عنوان يكي ديگر از مزيت هاي اين روش مطرح است.

نقطه ضعف اصلي اين روش نيز از تفاوت در ماهيت سرويس ها و مؤلفه ها ناشي مي شود. زيرا فاكتورهاي مهم در ارزيابي سرويس ها با فاكتورهاي ارزيابي مؤلفه ها متفاوت است و در صورت استفاده از اين روش ها، سرويس هايي شناسايي مي شوند كه مطابق با فاكتورهاي مناسب براي مؤلفه ها مي باشند . فاكتورهاي لازم براي سرويس ها را ندارند.

همچنين اين روش به منظور بررسي دوباره دامنه مسئله، سربار روال شناسايي سرويس ها را افزايش مي دهد.

اين رده از روش ها بر استفاده از دارايي هاي موجود تاكيد دارند به همين علت به اين روش ها، روش هاي پايين به بالا نيز مي گويند. در اين روش ها سيستم موجود و موروثي سازمان جهت كشف سرويس ها مورد بررسي، ترزيابي و پالايش قرار مي گيرند و در صورت تطبيق قسمتي از سيستم هاي موجود با فاكتورهاي سرويس، از روي آن سرويس مورد نياز توليد مي گردد.

تصلي ترين مزيت اين روش كاهش هزينه و زمان در توليد سرويس هاست زيرا در اين روش سرويس ها از درون سيستم هاي موروثي استخراج شده و نياز به هزينه و زمان زياد براي توليد ندارند.

نقطه ضعف اصلي اين روش نيز در استفاده از سيستم هاي موروثي ناشي مي شود. زيرا در صورتي كه اينسيستم هاي موروثي طراحي ضعيفي داشته باشند و يا در منطق آنها اشتباهي وجود داشته باشد اين مورد در سرويس هاي آينده نيز تكثير شده و اثر گذار خواهد بود.

در اين روش برنامه ها به دو قسمت پيش زمينه و پس زمينه تقسيم مي گردند. پس از توليد قسمت پيش زمينه به منظور كشف سرويس ها، پرسوجوهايي (query) طراحي شده و با استفاده از آنها سرويس هاي مورد تياز كشف مي گردند.

مزيت اين روش در سرعت كشف سرويس ها و قابليت استفاده مجدد از آنهاست. زيرا با استفاده از اين پرسوجوها به سرعت مي توان سرويس هاي مورد نياز را شناسايي نمود و با توجه به وابسته نبودن اين سرويس ها به دامنه حرفه اي خاص، امكان استفاده مجدد از آنها افزايش مي يابد.

اما نقطه ضعف اين روش نيز از طراحي ضعيف برنامه هاي پيش زمينه و پرس و جوها ناشي مي گردد زيرا در صورتي كه اين برنامه ها از كيفيت پاييني برخوردار باشند، اين امر بر كيفيت سرويسهاي شناسايي شده نيز اثر خواهد گذاشت.

در اين روش، با توجه به اينكه در صورت اجراي سرويس ها بر روي يك سكوي سخت افزاري كارآيي انها افزايش مي بابد، يك سري عمليات به منظور اجراي سرويس ها بر سكوهاي سخت افزاري مشخص انجام مي گيرد.

اين امر باعث كاهش هزينه در توسعه سرويس ها مي گردد كه همين مسئله به عنوان اصلي ترين مزيت اين روش ها مطرح مي باشد.

اما نقطه ضعف اصلي اين روش ها كه با اصلي ترين ويژگي سرويس ها در تعارض است اين است كه در صورت توجه به مسائل مربوط به سكوهاي سخت افزاري سرويس ها ، موجب كاهش سطح تجريد سرويس ها شده و يكي از اصلي تريم اصول در طراحي سرويس ها را نقض مي نماييم.

در تمامي روش هاي قبل تمركز بر نيازهاي وظيفه مندي سرويس ها مي باشد. در اين روش برخلاف تمامياين روش ها، تمركز بر ويژگي هاي غير وظيفه مندي و كيفي سرويس ها قرار مي گيرد. در اين روش پس از شناسايي سرويس ها توسط يكي از روش هاي قبل، طراحي سرويس را به منظور دستيابي به ويژگي هاي كيفي مورد انتظار تغيير مي دهيم.

همانطور كه واضح است نقطه قوت اين روش توجه و تمركز بر ويژگي هاي كيفي سرويس هاست كه مي تواند تاثير زيادي بر نحوه استفاده از سرويس ها داشته باشد.

نقطه ضعف اصلي اين روش نيز در وابسته بودن آن است. اين بدان معناست كه ابتدا بايد مجموعه سرويس هاي كانديد توسط يكي از روش هاي فوق شناسايي شوند و سپس از اين روش استفاده نماييم. همچنين در صورت وجود نقص در خروجي روش هاي فوق اين خطا در خروجي اين روش نيز تكثير پيدا مي كند.

پس از آشنايي با رده بندي كلي روش ها و قوت و ضعف هر يك از آنها در ادامه به معرفي برخي روش هاي ارائه شده در اين زمينه مي پردازيم. بايد توجه داشت روش هايي كه در ادامه معرفي مي گردند، روش هاي شاخص در اين حسطه هستند كه بيشتر از ساير روش ها مورد استقبال قرار گرفته اند و مجموعه زير حاوي تمامي روش هاي ارائه شده در اين زمينه نمي باشد.

## Zimmermann روش اول 2.3

در اين روش به اين نكته كه روش هاي گذشته مانند روش تحليل و طراحي شيء گرا (OOAD)، معماري سازماني (EA) و مدلسازي فرآيندهاي حرفه (BPM) الگوهاي معماري سرويس گرا را پوشش نمي دهند اشاره شده است و همچنين با توجه به اين موارد نتيجه گيري شده است كه به روشي جديد جهت تحليل و طراحي در زمينه سرويس گرايي نياز است.

روش هاي EA ،OOAD و BPM به صورت جداگانه خروجي هاي مفيدي را توليد نمي كنند. در اين مقاله روش Service Oriented Analysis and Design) SOAD) يا تحليل و سراحي سرويس گرا به عنوان روشي تركيبي از روش هاي قبل معرفي مي شود.

نيازمندي هاي روش SOAD عبارتند از:

* فرآيندها و يادداشت ها (notations) بايد به صورت رسمي يا غير رسمي تعريف شده باشند.
* با روش OOAD اشياء و كلاس ها مشخص شده و BPM مدل هاي مبتني بر رويداد (event driven) را توليد مي كند و SOAD اين خروجي ها را در كنار هم قرار مي دهد.
* بايد توجه داشت كه اين روش مبتني بر موارد كاربري (use case driven) نيست و موارد كاربري مي توانند در مرحله دوم استفاده شوند.

در اين مقاله جهت تشخيص سرويس ها (service identification) از روش هاي زير استفاده مي شود:

* تحليل مستقيم و غير مستقيم حرفه
* (domain decomposition) تجزيه دامنه حرفه
* در نظر گرفتن دامنه بندي سرويس ها
* قواعد نامگذاري سرويس ها

معايب اين روش عبارتند از:

* در اين مقاله يك روش عملياتي معرفي نشده است به عبارت ديگر يك روش قابل استفاده و كاربردي در اين مقاله ذكر نشده است.
* در اين روش به اهداف حرفه توجهي نشده است. به عبارت ديگر در هيچ يك از مراحل كشف سرويس ها اين امر كه تمامي اهداف حرفه توسط روش ذكر شده پوشش داده مي شوند يا خير، توجه نشده است.

## Zimmermann روش دوم 3.3

Analysis and Design Techniques for Service-oriented Development اين مقاله با عنوان ارائه شده M. Pestel و G. Waller ،N. Schlimm ،O. Zimmermann توسط and Integration [3] .است

در اين مقاله پس از بررسي روش Service Oriented Modeling and Architecture) SOMA) روشي جهت تبديل مدل هاي موارد كاربري و فرآيندهاي حرفه پيشنهاد گرديده اما روشي مشخص جهت اين امر ارائه نشده است.

همچنين در اين روش مسائلي مانند دانه بندي سرويس ها و ساير موارد كيفي سرويس ها در نظر گرفته نشده اند.

## Zhang روش 4.3

Service Identification and packaging in Service-oriented اين مقاله با عنوان

[16] .ارائه شده است H. Yang و R. Liu ،Zh. Zhang توسط Reengineering

در اين مقاله مراحل تشخيص سرويس ها (service identification) به صورت زير تعيين شده اند:

* تشخيص سرويس ها در دامنه مسئله . خروجي اين فرآيند مدل دامنه (domain model) مسئله خواهد بود.
* تشخيص سرويس ها در سيستم هاي موروثي. اين گام با استفاده از روش هاي خوشه بندي

(clustering) انجام مي گردد.

* تطبيق وظيفه مندي هاي كشف شده در سيستم هاي موروثي با وظيفه مندي هاي حرفه

(functions) در سرويس هاي منطقي مشخص شده. با اين كار در مورد استفاده از سرويس هاي موروثي براي پوشش نيازمندي ها و يا ساخت سرويس هاي جديد تصميم گيري مي شود.

اشكال عمده اين روش عدم توجه به فاكتور دانه بندي سرويس ها است. در اين روش همانطور كه در بالا اشاره شد، ابتدا سرويس هاي موجود در دامنه مسئله شناخته شده و سپس سرويس هاي موجود در سيستم هاي موروثي كشف مي شوند و در مورد استفاده يا عدم استفاده از اين سرويس هاي موروثي تصميم گيري مي شود و در هيچ مرحله اي دانه بندي سرويس ها به عنوان فاكتوري مهم و تاثير گذار بر ويژگي سيستم نهايي در نظر گرفته نمي شود.

**Amsden روش 5.3**  ارائه شده J. Amsden توسط Modeling SOA: Part 1. Service identification اين مقاله با عنوان [15] .است

در اين مقاله بر موارد زير تاكيد شده است:

* تلاش براي ارائه سرويس هايي به منظور پوشش دهي اهداف حرفه
* توجه روي مفهوم جداسازي دغدغه ها (separation of concerns)
* توجه روي مفهوم اتصال سست (loose coupling)

در اين مقاله روش كار به اين ترتيب است كه ابتدا با تمركز روي اهداف حرفه شروع به كار كرده سپس فرآيندهاي حرفه كه به اين اهداف پاسخ مي دهند را شناسايي مي كند. پس از شناسايي نيازهاي حرفه، سرويس ها را به منظور پوشش آنها تشخيص مي دهد.

در اين مقاله از روش تجزيه دامنه (domain decomposition) براي شناسايي سرويس هاي كانديد از روي مدل حرفه (business model) اشتفاده مي شود.

اشكال عمده روش هاي ارائه شده در اين مقاله نيز در نظر نگرفتن فاكتور دانه بندي سرويس هاست. اين روش توجه خوبي بر پوشش دهي اهداف حرفه و فرآيندهاي آن دارد اما فاكتور دانه بندي سرويس ها را به عنوان فاكتوري تاثير گذار بر ويژگي سيستم نهايي در نظر نمي گيرد.

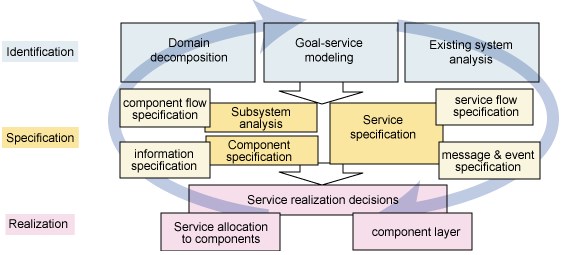
## SOMA روش 6.3

Ali توسط ش Service-Oriented Modeling and Architecture (SOMA) اين مقاله با عنوان

Arsanjani ارائه شده است. [11] در اين مقاله مفاهيم زير آورده شده اند:

* روش هاي OOAD معمول مفاهيمي چون سرويس، جريان و مؤلفه ها را پوشش نمي دهند.
* روش SOMA شامل فعاليت هاي مدلسازي، تحليل و طراحي و فعاليت هايي به منظور تعريف مؤلفه هاي موجود در تمامي لايه هاي SOA مي باشد.
* در اين مقاله براي تشخيص سرويس ها ابتدا از روش بالا به پايين به منظور يافتن سرويس هايكانديد و سپس از روش مدلسازي هدف سرويس به منظور بررسي پوشش دهي اهداف توسطسرويس هاي شناخته شده اشتفاده مي شود. در آخر نيز از روش پايين به بالا استفاده مي شود.
* در روش بالا به پايين از روش domain decomposition استفاده مي شود. در اين روش دامنه حرفه به نواحي مختلفي از نظر وظيفه مندي تجزيه مي شود. همچنين جريان ها يا فرآيندها به زير فرآيندها و موارد كاربري سطح بالا تجزيه مي شوند. اين موارد كاربري اغلب كانديدهاي خوبي براي سرويس هاي حرفه هستند.
* در روش پايين به بالابا استفاده از روش تحليل دارايي هاي موجود، سيستم هاي موروثي به منظور يافتن مؤلفه هايي جهت پوشش وظيفه مندي هاي شناخته شده تحليل مي شوند.
* در روش مياني از روش مدلسازي هدف-سرويس به منظور كشف سرويس هايي كه تا به حال شناخته نشده بودند استفاده مي شود. در اين روش سرويس ها را به اهداف، زير اهداف و KPIها متناظر مي كنيم.

در شكل زير فعاليت هاي روش SOMA در تمامي مراحل مدلسازي سرويس ها آمده است:



**شكل شماره 3 -1 - فعاليت هاي SOMA[11]**

در زير فازها و فعاليت هاي روش SOMA را به اختصار شرح مي دهيم.

**فاز شناسايي سرويسها  *Identification***

فاز شناسايي سرويسها يكي از بحراني ترين فازها در موفقيت پروژه معماري سرويس گرا است زيرا در اين فاز نيازمندي هاي حرفه شناخته شده و پوشش داده مي شوند. هدف اين فاز توليد مجموعه اي از سرويس هاي كانديد براي پروژه سرويس گرا و مشخص كردن عمليات آنها است. براي شناسايي سرويسهاي كانديد تحليل گر در مورد استفاده از راهبردها و روش هاي مختلف تصميم گيري مي كند.

نقش هاي درگير در اين فار عبارتند از : معمار نرم افزار (نقش اصلي) و تحليل گر حرفه (نقش كمكي). در خلال اين فاز مدل كاري سرويس ها توليد مي شود و در پايان فاز، اين مدل به معمار نرم افزار براي انجام عمليات مربوط به فاز توصيف سرويس ها (Service Specification) تحويل داده مي شود.

اصلي ترين ورودي هاي اين فاز سرويس هاي موجودي كه سازمان به آنها (از هر طريقي مانند تملك، خريداري و ...) دسترسي دارد مي باشند. اين سرويس ها معمولا در انباره سرويس هاي سازمان موجود مي باشند. البته بايد توجه داشت كه در سازمان هايي كه اولين تجربه معماري سرويس گراي خود را انجام مي دهند ،طبيعتا اين ورودي وجود ندارد، و سرويس هاي تماما بايد توليد شده و پس از انجام فازهاي مختلف در انباره براي استفاده هاي در پروژه هاي سرويس گرايي آينده قرار بگيرند.

همانطور كه در شكل شماره 10 مشاهده مي كنيد ،فعاليت هاي درون اين فاز عبارتند از: تجزيه دامنه حرفه (Domain Decomposition)، مدلسازي سرويس-هدف (Goal-Service Modeling) و تحليل دارايي هاي موجود (Existing Asset Analysis). به طور كل مي توان گفت اين فاز داراي تركيبي از تحليل هاي بالا به پايين (تجزيه دامنه حرفه)، پايين به بالا(تحليل دارايي هاي موجود) و تحليل مياني(مدلسازي سرويس- هدف) است.[11].در روش بالا به پايين با بررسي نيازمندي هاي حرفه، سرويس ها استخراج مي شوند، سپس با استفاده از تحليل پايين به بالا سرويس هاي قابل استفاده در سيستم هاي موروثي شناسايي مي گردند.

**تجزيه دامنه حرفه**

در اين مرحله دامنه حرفه به فرآيندهاي حرفه، زير فرآيندها و موارد كاربري تجزيه مي گردد. از ديد حرفهدامنه مجموعه اي حيطه هاي وظيفه مندي است. در اين مرحله اين حيطه ها را بر اساس دو پرسش كليدي زيربر حسب نيازمندي ها تجزيه مي كنيم :

* آيا اين وظيفه مندي در حيطه وظايف سازمان قرار مي گيرد؟ آيا اين وظيفه مندي توسط يك واحد سازمان انجام مي شود يا توسط بيش از يك واحد؟
* اگر اين نيازمندي فراتر از حيطه سازمان است ، چه سازمان هاي ديگري در انجام آن با سازمان همكاري دارند؟

با استفاده از دو پرسش فوق دامنه مسئله را به فرآيندهاي سازمان، زير فرآيندها و موارد كاربري تجزيه مي كنيم. اين موارد كاربري (سطح سوم تجزيه) معمولا كانديدهاي خوبي براي سرويس ها مي باشند.

**تحليل دارايي هاي موجود**

پس از شناخت سرويس هاي كانديد مورد نياز براي پاسخگويي به نيازهاي سازمان (تحليل بالا به پايين)، سيستم هاي موروثي سازمان به منظور شناسايي سرويس هايي كه مي توان از آنها در پروژه استفاده كرد تحليل مي شوند (تحليل پايين به بالا). اين كار باعث كاهش هزينه و زمان پروژه مي گردد. به اين فعاليت تحليل دارايي هاي موجود مي گويند. البته بايد توجه داشت كه گاهي نياز است تا دارايي هاي موجود (سيستم هاي موروثي) به منظور مطابقت با نيازهاي جديد، واحدبندي مجدد شوند.

**مدلسازي سرويس- هدف**

همانطور كه گفته شد، موارد كاربري كه در تجزيه دامنه حرفه شناسايي شدند كانديد هاي خوبي براي سرويس ها مي باشند. در اين مرحله مدلي به نام مدل سرويس-هدف به منظور بررسي كامل بودن مجموعه سرويس هاي كانديد توليد مي شود.

از طريق برگزاري جلسه با مالكان حرفه و پرسش از آنها در مورد اهداف موجود در حيطه پروژه، مي توان زير هدف هاي پروژه كه پيش نياز رسيدن به اهداف كلان پروژه و سازمان هستند را شناسايي كرد. هر زير هدف را نيز مي توان به زير هدف هاي كوچكتر تجزيه نمود. اين عمل تا زماني كه سرويس هايي كه برآورده كننده زيراهداف هستند به روشني مشخص شوند، ادامه پيدا مي كند. به درخت حاصل از اين تجزيه، درخت هدف -سرويس گفته مي شود.

با در دست داشتن مدل هدف-سرويس و شناسايي سرويس هاي مربوط به هر هدف مي توان كامل بودن مجموعه سرويس هاي كانديد را بررسي كرد.

پس از انجام اين مرحله مجموعه اي از سرويس هاي كانديد براي پياده سازي به منظور پوشش نيازهاي سازمان و مجموعه اي از سرويس هاي موجود از طريق سيستم هاي موروثي كه براي پاسخگويي به برخي از نيازمندي هاي سازمان قابل استفاده هستند توليد شده است. همچنين با استفاده از درخت هدف-سرويس كامل بودن مجموعه سرويس هاي كانديد مورد بررسي قرار گرفته است.

**فاز توصيف سرويس ها *Specification***

پس از شناسايي مجموعه سرويس هاي كانديد فاز توصيف سرويس ها آغاز مي گردد. اين فاز توسط معمار نرم افزار به همراه طراحان ( به صروت اختياري ) انجام مي شود. هدف اين فاز توصيف كامل عناصر طراحي معماري سرويس گرا است.

فاز توصيف سرويس ها را مي توان وظيفه مربوط به معماري در فرآيند معماري سرويس گرا دانست. اين ديد اهميت اين فاز را به خوبي نمايان مي سازد.

براي روشن تر شدن تمايز ميان فاز توصيف سرويس ها و شناسايي سرويس ها مي توان فاز شناسايي سرويس ها را (Service Identification) را به عنوان فعاليت هاي مربوط به تحليل مدل سرويس و فاز توصيف سرويس ها (Service Specification) را به عنوان فعاليت هاي مربوط به طراحي مدل سرويس دانست.

در خلال فاز توصيف سرويس ها مدل سرويس به روشني مشخص مي شود و در پايان اين فاز، اين مدل به طراحان براي انجام فاز عينيت بخشيدن به سرويس ها و توسعه دهندگان براي پياده سازي سرويس ها داده مي شود.

همانطور كه در شكل شماره 10 مشاهده مي كنيد ، وظايف اصلي درون اين فاز عبارتند از: تحليل زيرسيستم ها (Subsystem Analysis) ، توصيف مؤلفه ها (Component Specification) و تخصيص سرويس ها (Service Allocation). در اين فاز ويژگي ها و قابليت هاي سرويس ها تشريح مي گردند و به سؤالهايياز جمله : سرويس چه خدمتي ارائه مي دهد؟ منطق داخلي سرويس ها چيست؟ نيازمندي هاي غير وظيفهمندي هر سرويس چيست؟ و... جواب داده مي شود.

**تحليل زيرسيستم ها**

ورودي اين مرحله زيرسيستم هايي (سرويس ها) كه از مرحله تجزيه دامنه بدست آمده اند مي باشند و در اين مرحله وابستگي بين اين زير سيستم ها و جريان هاي بين آنها مشخص مي گردد.

در تحليل زيرسيستم ها مدل هاي اشياء به منظور مشخص كردن نحوه انجام كار در داخل زير سيستم و همچنين نمايش زير سيستم هاي درون زير سيستم در حال بررسي (سلسله مراتب زيرسيستم ها) توليد مي شوند.

ساختار طراحي زير سيستم ها براي توليد ساختار پياده سازي مؤلفه ها – كه خود اساسي براي پياده سازي سرويس ها هستند – استفاده مي شود.

در اين مرحله خصوصيات سرويس ها از جمله : وابستگي بين آنها، نحوه تركيب آنها، پيام هاي بين آنها ، فاكتور هاي كيفيت آنها و... تعيين مي شوند.براي اين كار 6 مرحله زير انجام مي گردند.

1. تست *Litmus* : اين تست به اين سؤال كه "از ميان سرويس هاي كانديد كداميك براي پياده سازي مناسب است؟" پاسخ مي دهد. بايد توجه داشت تمامي سرويس هايي كه در مرحله تجزيه دامنه مشخص شدند براي پياده سازي مناسب نيستند. براي پيدا كردن زير مجموعه اي از اين سرويس ها در روش تست Litmus براساس موارد مختلف از جمله تطابق سرويس با نيازهاي حرفه، تكراري نبودن سرويس و ... تصميم گيري مي شود. منظور از تكراري نبودن سرويس اين است كه وظايفي كه سرويس در دست بررسي انجام مي دهد توسط سرويس هاي ديگر پوشش داده نشوند.
2. .مشخص كردن وابستگي ميان سرويس ها :بازبيني جزئيات سرويس ها باعث مشخص شدن وابستگي سرويس به سرويس هاي ديگر و نحوه تعامل ميان سرويس ها مي شود.
3. مشخص كردن تركيب سرويسها و جريان ميان آنها :با بررسي حيطه هاي كاري و فرآيندهاي حرفه نحوهتوليد سرويس ها از تركيب سرويس هاي ديگر و جريان كاري ميان سرويس ها براي انجام فرآيندهاي حرفه را مشخص مي كند.
4. مشخص كردن نيازهاي غيروظيفه مندي :با استفاده از نيازهاي غير وظيفه مندي ، ويژگي هاي كيفي سرويس ها مشخص مي گردد.
5. مشخص كردن پيام هاي سرويس :در اين قسمت قالب پيام هاي ورودي و خروجي سرويس و محتواي آنها مشخص مي شود.
6. مستندسازي تصميمات مديريتي :برخي مواقع تركيب سرويس ها منجر به برخي تصميمات مديريتي در حالت هاي مختلف كاري سرويس مي شود. اين تصميمات مديريتي در اين مرحله مستندسازي مس شوند.

**تخصيص سرويسها**

پس از مشخص كردن سرويس ها و جزئيات آنها در اين مرحله درباره تخصيص سرويس ها به مؤلفه ها تصميم گيري مي كنيم. اغلب اين تخصيص يك-به-يك است.

همچنين در اين مرحله نحوه تخصيص سرويس ها و مؤلفه به لايه هاي معماري سرويس گرا مشخص مي شود. براي تصميم گيري در مورد چگونگي تخصيص سرويس ها و مؤلفه ها به لايه هاي معماري علاوه بر توجه به معماري نرم افزار بايد به معماري محيط فني و عملياتي كه سيستم در آن اجرا مي شود توجه نمود.

**توصيف مؤلفه ها**

در مرحله بعد از فاز توصيف سرويس ها ، جزئيات مؤلفه هايي كه سرويس ها را پياده سازي مي كنند مشخص مي شود. اين توصيف دربرگيرنده جزئياتي در مورد داده هاي مؤلفه، قوانين، سرويس ها و نحوه پيكربندي مؤلفه مي باشد. همچنين در اين مرحله پيام هاي مؤلفه ها و خصوصيات رويدادهاي آنها مشخص مي گردد.

پس از مشخص كردن خصوصيات سرويس ها و نحوه اجراي سرويس ها توسط مؤلفه ها و همچنين تعريف جزئيات مؤلفه ها، بايد در مورد نحوه توليد و در اختيار گرفتن سرويس ها تصميم گيري شود.

**فاز عينيت بخشيدن به سرويسها *Realization***

در اين فاز در مورد مباحث مربوط به "امكان سنجي فني" پياده سازي يا استخراج سرويس ها از سيستم هاي موروثي تصميم گيري مي كند.[11] گزينه هاي ديگري كه در اين زمينه وجود دارد عبارتند از تجميع سرويس ها و توليد سرويس مورد نياز ، تغيير سرويس هاي قديمي براي توليد سرويس مورد نياز ، سفارش سرويس و

تصميمات ديگري كه در اين فاز گرفته مي شود – علاوه بر تصميمات در مورد وظيفه مندي – عبارتند از :

امنيت، مديريت و نظارت بر سرويس ها.

با توجه به فازهاي سه گانه متدولوژي SOMA و چرخه حيات معماري سرويس گرا (شكل شماره 1) مي توان سه فاز فوق را به شكل زير در مراحل چرخه حيات معماري سرويس گرا گنجاند:

فازهاي شناسايي و توصيف سرويس ها در مرحله مدل، فاز عينيت بخشيدن به سرويس ها در مراحل گردآوري، نصب و مديريت.

## Portier روش 7.3

توسط SOA terminology overview, Part 3: Analysis and design اين مقاله با عنوان

B.Portier ارائه شده است. [12] در اين مقاله نكات زير آورده شده اند:

* تحليل به اين علت براي SOA حياتي است كه تطابق فرآيندهاي نرم افزاري و نيازمندي هاي حرفه در فاز شناخت سرويس ها (service identification) انجام مي شود.
* شناخت سرويس ها (service identification) فعاليت مركزي در تحليل سرويس گراست. هدف service identification شناخت گروه سرويس هاي مفهومي و عمليات آنهاست.
* Service identification توسط تحليل گر حرفه كه به كعكار نرم افزار كمك مي كند انجام مي گردد.
* تحليل بالا به پايين : در اين روش فرآيندهاي حرفه و موارد كاربري تجزيه مي شوند. مزيت اين روش اين است كه از تطابق ميان سرويس ها با نيازهاي حرفه اطمينان داريم.
* تحليل پايين به بالا : اين روش روي تحليل دارايي هاي فناوري اطلاعات موجود تمركز دارد. مزيت اين امر استفاده مجدد است.
* تحليل مياني (meet-in-the-middle) : اين روش روي تطبيق دادن نيازمندي ها (سرويس هاي شناخته شده توسط روش بالا به پايين) و آنچه توسط فناوري اطلاعات موجود است (خروجي هاي مرحله پايين به بالا ) تمركز دارد.

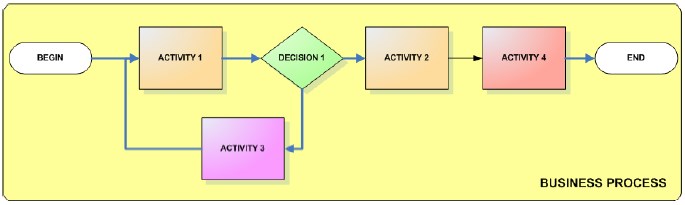
همانگونه كه از مطالب فوق مشخص است اين روش نيز به فاكتور دانه بندي سرويس ها توجهي ندارد.

## Inganti روش 8.3

و S. Inaganti توسط Service Identification: BPM and SOA Handshake اين مقاله با عنوان

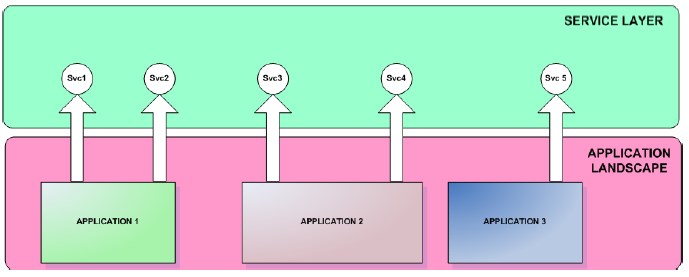
G.K. Behara ارائه شده است. [14] در اين مقاله موارد زير ذكر گرديده اند:

* دلايل اهميت شناسايي سرويس ها (service identification) : كم بود مستندات فرآيند حرفه، تخصص در تحليل، استفاده مجدد، كم بود درگيري اجرايي به همراه بروز اختلال در تيم هاي پروژه • اشتباهاتي كه در شناسايي سرويس ها (service identification) رخ مي دهند ممكن است در طراحي و پياده سازي ادامه پيدا كنند.
* روش هاي كلي در گام شناسايي سرويس ها عبارتند از: 1. روش هاي بالا به پايين 2. روش هاي پايين به بالا
* در روش هاي بالا به پايين ممكن است از روش هاي مبتني بر فرآيندهاي حرفه و يا روش هاي مبتني بر موارد كاربري استفاده شود. تصميم گيري در مورد هر يك از روش ها بستگي به گستردگي دامنه مسئله دارد.
* در اين مقاله عنوان شده است كه بهتر است دو تيم داشته باشيم. تيم سرويس ها و تيم برنامه هاي كاربردي. تيم سرويس ها دنبال سرويس ها و نيازمندي براي هر سرويس مي گردد و تيم برنامه هاي كاربردي همزمان نيازمندي هر زير سيستم را شناخته و به دنبال سرويس براي مطابقت با زيرسيستم است.
* در اين روش ابتدا فرآيندهاي حرفه به صورت مجموعه اي از فعاليت ها (activity) كه با هم در ارتباط هستند مشخص مي شوند. سپس وظيفه مندي هاي برنامه هاي كاربردي مشخص شده و هر كدام به صورت يك سرويس معرفي مي گردند. سپس عمل تطبيق بين اين سرويس ها و فعاليت ها انجام مي شود. اگر فعاليتي وجود داشت كه سرويسي آن را پوشش نمي داد، فعاليت به زير فعاليت ها شكسته شده و براي زير فعاليت ها سرويس جديد بوجود مي آوريم. در شكل هاي زير اين موارد نشان داده شده اند:



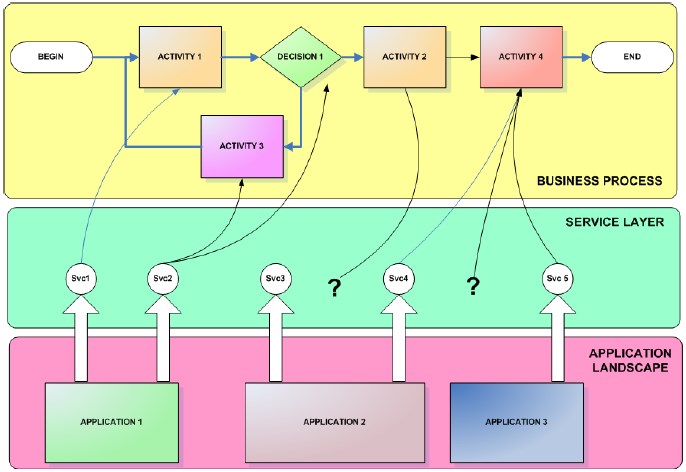
**شكل شماره 3 -2 - بالا به پايين [14]**

در شكل شماره 11، توسط روش بالا به پايين يك فرآيند حرفه به تعدادي فعاليت (activity) تجزيه شده است.



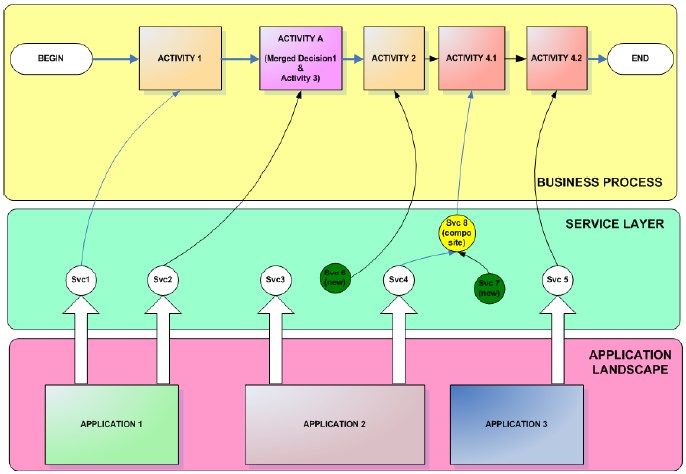
**شكل شماره 3 -3 - روش پايين به بالا [14]**

در شكل شماره 12، از روي وظيفه مندي هاي برنامه هاي كاربردي، سرويس ها تعيين مي گردند. به اين ترتيب كه هر يك از واسط هاي برنامه هاي كاربردي به يك سرويس متناظر مي گردند.



**شكل شماره 3 -4 - تناظر فعاليت ها و سرويسها [14]**

در شكل شماره 13، سرويس هاي شناخته شده به فعاليت ها متناظر مي گردند. مشاهده مي كنيد فعاليت activity 2 توسط هيچ سرويسي پوشش داده نمي شود. براي رفع اين مشكل مطابق شكل شماره 14 فعاليت ها تجزيه مي گردند.



**شكل شماره 3 -5 - تجزيه فعاليت ها [14]**

در شكل فوق براي حل مشكل، سرويس جديدي به منظور پوشش فعاليت activity2 توليد شده است.

همچنين سرويس جديدي نيز از تركيب سرويس 4 و يك سرويس جديد به منظور پوشش فعاليت Activity4 توليد گرديده است.

اشكال روش اين مقاله اين است كه در آن فرآيندها به صورت جداگانه در نظر گرفته مي شوند، لذا امكان بروز افزونگي و رويهم افتادگي سرويس ها در آن وجود دارد.

**نتيجه گيري**

در اين فصل با روش هاي ارائه شده در زمينه تشخيص سرويس ها آشنا شديم. همانطور كه در اين فصل بيان شد در كل 10 رده از تكنيك ها به منظور كشف سرويس ها وجود دارد . اين رده ها تعريف شده و مزايا و معايب هر يك بيان گرديد.

همچنين در ادامه با 7 روش مختلف كه در هفت مقاله مطرح ارائه شده بودند آشنا شديم و مزايا و معايب اين روش ها را مشاهده نموديم.

در فصول آينده با شناخت نسبت به حيطه مسئله و ويژگي هاي سرويس هاي مورد انتظار و همچنين ويژگي هاي روش هاي ارائه شده در اين زمينهقصد داريم روشي جديد ارائه نماييم كه نقاط ضعف روش هاي موجود را پوشش داده و از نقاط قوت اين روش ها نيز بهره ببرد.

# فصل چهارم ويژگي هاي راه حل مورد انتظار

**مقدمه**

در فصول قبل با مفهوم گام تشخيص سرويس ها و جايگاه آن در چرخه حيات معماري سرويس گرا و فاز مدلسازي سرويس گرا آشنا شديم. همچنين روش هاي پركاربرد در حيطه تشخيص سرويس ها معرفي شده و مورد تجزيه و تحليل قرار گرفتند.

در اين فصل قصد داريم تا با شناخت فاكتورهاي لازم به منظور ارزيابي كيفيت سرويس هاي نرم افزاري و همچنين آگاهي از نقاط ضعف روش هاي موجود در زمينه تشخيص سرويس هاي نرم افزاري، به بيان چاچوبي جهت ارائه روشي به منظور تشخيص سرويس ها كه پوشش دهنده نقاط ضعف ديگر روش ها بوده و در عين حال از نقاط قوت آنها نيز سود ببرد بپردازيم. به اين منظور در اين فصل ابتدا مروري كوتاه بر ويژگي هاي مورد انتظار سرويس هاي نرم افزاري كرده، سپس نقاط ضعف عمده در ساير روش ها عنوان مي گردد.

در ادامه نيز با توجه به اين دو فاكتور اصول روش مورد انتظار بيان خواهد شد.

## 1.4 نتایج مورد انتظار از سرویس ها

در فصل دوم و در بخش 2.2 معيارهاي كيفي سرويس ها بيان شدند، همچنين در بخش 3.2 فاكتور دانه بندي سرويس ها به عنوان عاملي تاثير گذار بر برخي جنبه هاي سيستم نهايي مطرح شد. در اين بخش قصد داريم تا با مروري كوتاه بر مطالب بخش هاي فوق ويژگي هاي مورد انتظار از خروجي هاي روش تشخيص سرويس ها كه همان سرويس هاي نرم افزاري هستند را متذكر شويم.

در منبع [1] توماس ارل، ويژگي هاي زير را به عنوان فاكتورهاي كيفي سرويس ها بيان كرده است:

* **سرويس ها قابليت استفاده مجدد دارند** به اين معناست كه سرويس ها به گونه اي طراحي مي گردند كه پتانسيل استفاده مجدد را دارا باشند. البته ممكن است سرويسي هيچگاه مورد استفاده مجدد قرار نگيرد اما داراي اين خصوصيت باشد.
* **سرويس ها يك قرارداد** **رسمي را به اشتراك مي گذارند** به اين معناست كه سرويس ها به منظور برقراري ارتباط با ساير سرويس ها تنها نياز به اشتراك گذاري قرارداد رسمي خود دارند كه در اين قرارداد سرويس به معرفي ويژگي هاي اصلي خود، وظايفي كه ارائه مي دهد و ويژگي هايي كه طرف استفاده كننده براي استفاده از سرويس بايد داشته باشد مي پردازد.
* **سرويس ها اتصال سست با يكديگر دارند** به اين معناست كه سرويس ها بايد به گونه اي طراحي شوند كه ارتباطشان با ساير سرويس ها تا حد امكان كم باشد. در صورتي كه سرويس ها اين ويژگي را داشته باشند، امكان توسعه، تست و اجراي مستقل سرويس ها بوجود مي آيد همچنين در صورت وجود اين ويژگي در سرويس هاي مختلف در سيستم نهايي، قابليت نگهداشت سيستم نهايي به صورت چشمگيري افزايش پيدا مي كند زيرا در صورت نياز به تغيير در ماهيت يك سرويس (نياز به تغيير در واسط هاي سرويس) تعداد سرويس كمي كه با آن در ارتباط هستند نياز به تغيير دارند.
* **سرويس ها منطق دروني خود را پنهان مي كنند** بدان معناست كه تنها قسمتي از سرويس كه از طريق جهان خارج قابل مشاهده است، قرارداد رسمي سرويس مي باشد. اين امر سبب مي شود منطق دروني سرويس و نحوه عينيت بخشي به تعهدات قرارداد رسمي سرويس از ديد جهان خارج و متقاضي سرويس پنهان بماند. از جمله مزيت هاي اين امر افزايش امنيت سرويس مي باشد. زيرا در صورتي كه راههاي ارتباطي با سرويس تنها به واسط هاي از پيش تعيين شده آن محدود گردد امكان نفوذ به درون سرويس و ايجاد اختلال در كار آن كاهش يافته و تشخيص موارد نفوذ آسان مي گردد.علاوه بر اين اين ويژگي تا حد زيادي سبب جلوگيري از پيدايش پيچيدگي در توسعه سيستممي شود. زيرا اگر سرويس ها منطق دروني خود را پوشش نمي دادند براي توسعه سيستم بايد مواردمربوط به موجوديت هاي دروني سرويس ها را نيز در نظر مي گرفتيم كه اين امر سبب افزايش موجوديت هاي مورد مطالعه و روابط ميان آنها شده و به شدت باعث افزايش پيچيدگي در امر طراحي و توسعه سيستم هاي نرم افزاري مي شود.
* **سرويس ها قابليت تركيب دارند** به اين معناست كه سرويسي ها ممكن است از تركيب چند سرويس ديگر توليد شوند. اين امر باعث مي گردد منطق حرفه در دانه بندي هاي متفاوت پوشش داده شده و باعث افزايش قابليت استفاده مجدد و افزايش سطح انتزاع (abstraction) مي شود.
* **سرويس ها خودمختار هستند (autonomous)** به اين معناست كه منطق و وظيفه مندي هاي درون سرويس با مرز آشكاري مشخص مي شوند. سرويس در اين مرز كنترل فعاليت ها را در دست دارد و به سرويس هاي ديگر براي اجرا وابسته نمي باشد.
* **سرويس ها وضعيت خود را نگهداري نمي كنند (stateless)** به اين معناست كه سرويس ها لازم نيست اطلاعات مربوط به وضعيت را مديريت نمايند زيرا داده هاي مربوط به مديريت وضعيت ممكن است روي قابليت اتصال سست تاثير منفي بگذارد. به همين علت توصيه مي شود سرويس ها به گونه اي طراحي گردند كه كمترين اطلاعات وضعيتي را ثبت كنند.
* **سرويس ها قابل كشف هستند** به اين معناست كه سرويس ها بايد اجازه دهند توصيفشان توسط افراد و متقاضيان سرويس درك و كشف شده تا توسط آنها مورد استفاده قرار گيرند. به عبارت ديگر بايد مكانيسمي وجود داشته باشد كه نام سرويس ها و ويژگي هاي آنها را در اختيار متقاضيان قرار دهد به طوري كه متقاضيان با استفاده از اين مكانيسم سرويسي كه نيازمندي هايشان را پوشش مي دهد كشف نموده و از آن استفاده كنند.

در بالا با هشت ويژگي اصلي سرويس ها آشنا شديم. توصيه مي شود در سرويس هايي كه توسط فرآيندهاي تشخيص سرويس ها شناسايي مي شوند تا حد امكان اين ويژگي ها رعايت شده باشند البته اين به اين معني نيست كه در صورتي كه سرويسي يكي از ويژگي هاي فوق را نداشت لزوما از آن سرويس استفاده نمي كنيم بلكه اين فاكتورها را مي توان به عنوان معياري براي سنجش كيفيت سرويس هاي شناخته شده و در نهايتارزيابي كيفيت روش هاي استفاده شده براي شناسايي سرويس ها استفاده نمود.

علاوه بر موارد فوق، در بخش 3.2 با فاكتور اندازه سرويس ها آشنا شديم و ديديم اين فاكتور بر چهار ويژگي مهم در سيستم نهايي تاثير مي گذارد. اين ويژگي ها عبارتند از:

* (flexibility) انعطاف پذيري
* قابليت استفاده مجدد (reusability)
* (performance) كارآيي
* (complexibility) پيچيدگي

به عنوان مرور بر موارد فوق مي توان گفت:

اگر سرويس ها بزرگ باشند براي تغيير در يكي از جنبه ها يا وظيفه مندي هاي سيستم بايد دامنه گسترده اي از سيستم دستخوش تغيير شود. به همين علت بزرگي اندازه سرويس ها موجب كاهش انعطاف پذيري سيستم مي گردد. در مقابل هر چه سرويس ها اندازه كوچكتري داشته باشند، براي تغيير در قسمتي از سيستم تنها تغيير در حيطه سرويس هدف كفايت مي كند و به همين علت تغيير جنبه اي از سيستم سربار زيادي را ناشي نمي شود و در نتيجه انعطاف پذيري سيستم افزايش مي يابد.

قابليت استفاده مجدد يكي از بارزترين و اصلي ترين ويژگي ها در معماري سرويس گرا به شمار مي رود.

همانطور كه پيش از اين اشاره شد، قابليت استفاده مجدد باعث كاهش زمان و هزينه توليد سيستم هاي نرم افزاري شده و تا حد ممكن در توسعه سيستم هاي نرم افزاري تلاش در افزايش اين قابليت داريم.

دانه بندي سرويس ها بر اساس حيطه وظيفه مندي سرويس ها تعيين مي شود. به عبارت ديگر هرجه سرويسي حيطه بزرگتري از وظيفه ندي را پوشش دهد ، اصطلاحا ـن سرويس را بزرگتر و هر چه حيطه كوچكتري را پوشش دهد، اصطلاحا سرويس را كوچكتر گويند. در صورتي كه سرويس بزرگ باشد، حيطه بزرگتري از وظيفه مندي هاي حرفه را پوشش مي دهد و اصطلاحا سرويس در حيطه حرفه تنظيم شده و وابستگي بيشتري به حرفه خواهد داشت. همچنين به علت تفاوت در ماهيت حرفه هاي گوناگون امكان استفاده مجدد از آن سرويس در حرفه هاي ديگر كاهش مي يابد و باعث كاهش قابليت استفاده مجدد سرويس مي گردد.

در مقابل هرچه سرويس كوچكتر باشد، وابستگي كم تري به حرفه داشته و به احتمال زياد سرويس مطابق باوظيفه مندي خاصي طراحي شده است. به علت امكان بروز وظيفه مندي هاي مشابه در حيطه حرفه هايگوناگون امكان استفاده مجدد سرويس افزايش مي يابد

از آنجا كه براي برآورده كردن فرآيندهاي حرفه تعدادي از سرويس ها با هم همكاري مي نمايند، ارتباطات بين سرويس ها به اين منظور نوعي سربار تلقي مي گردند. به عبارت ديگر هنگام فراخواني سرويس ها به منظور برآورده كردن فرآيند حرفه، مقداري از كارآيي سيستم جهت برقراري ارتباط ميان سرويس هاي مختلف هدر مي رود. حال در صورتي كه سرويس ها كوچك باشند، براي برآورده كردن فرآيندهاي حرفه به تعداد بيشتري سرويس نياز است و در نتيجه ارتباطات ميان سرويسي افزايش يافته و سربار بيشتري به سيستم تحميل مي گردد. در نتيجه استفاده از سرويس هاي كوچك باعث كاهش كارآيي سيستم مي گردد.

در مقابل هر چه سرويس ها بزرگتر باشند، براي پوشش نيازمندي هاي يك فرآيند حرفه به تعداد سرويس كمتري نياز بوده و در نتيجه سرباز ناشي از ارتباطات ميان سرويسي كاهش مي يابد. همين امر باعث افزايش كارآيي سيستم نهايي مي گردد.

آخرين فاكتوري كه دانه بندي سرويس ها بر آن اثر مي گذارد، فاكتور پيچيدگي سيستم مي باشد. تاثير دانه بندي سرويس ها بر اين فاكتور تا حدودي شبيه تاثير دانه بندي بر فاكتور كارآيي سيستم مي باشد. در صورتي كه سرويس ها كوچك باشند، به تعداد بيشتري سرويس جهت پوشش دامنه حرفه نياز داريم. طبيعي است كه اين سرويس ها نياز به تعامل با يكديگر دارند. همين امر باعث مي شود در صورت داشتن سرويس هاي كوچك پيچيدگي سيستم به علت تعداد زياد ارتباطات افزايش يابد.

در سمت مقابل هرچه اندازه سرويس ها بزرگ تر باشد، به تعداد سرويس كمتر و در نتيجه ارتباطات كمتر ميان سرويسي نياز خواهد بود. همين امر پيچيدگي سيستم نهايي را كاهش خواهد داد.

**2**.4 نقاط ضعف

در فصل سوم با روش هاي معمول در زمينه تشخيص سرويس ها (service identification) آشنا شديم. همانطور كه گفته شد در كل 10 رده روش در اين زمينه تا به حال معرفي شده است و با نقاط قوت و ضعف اين گروه ها نيز تا حدي آشنا گرديديم. همچنين برخي از روش هاي پر استفاده در اين زمينه معرفي شده و به اختصار نقاط قوت و ضعف آنها مورد بررسي قرار گرفت.

به منظور ارائه روشي جديد كه نقاط ضعف و قوت روش هاي موجود را پوشش دهد نياز به شناسايي دقيقروش هاي گذشته است. همانطور كه در فصل پيش ديده شد، اشكالات زير به صورت عمده در روش هايگذشته به چشم مي خورد:

1. عدم ارائه روش مشخص و يا وابسته بودن روش معرفي شده:

در برخي از روش هاي ارائه شده، گام هاي مدون و مشخصي به منظور رسيدن به مجموعه سرويس هاي كانديد بيان نشده است. به عبارت ديگر در برخي از روش ها تنها به خروجي ها و ويژگي هاي خروجي ها اشاره شده گاهي تنها مراحل رسيدن به خروجي ها كه همان سرويس ها هستند بيان شده است در صورتي كه توضيحي در مورد اين مراحل، نحوه تقدم و تاخر آنها و ساير موارد داده نشده است.

1. در نظر نگرفتن فاكتورهاي كيفي سرويس ها :

در برخي از روش هاي ارائه شده، تنها برخي فعاليت ها به منظور رسيدن به مجموعه سرويس هاي كانديد بيان شده و در اين مراحل گام هايي به منظور رعايت و رسيدن به ويژگي هاي كيفي گنجانده نشده است. به عبارت ديگر ايندسته روش ها تنها به پوشش نيازهاي وظيفه مندي و نيازهاي دامنه مسئله بسنده كرده اند و به ويژگي هاي كيفي مورد انتظار از سرويس ها توجهي نشده است.

1. در نظر نگرفتن فاكتور اندازه سرويس:

در بسياري از روش ها به فاكتور اندازه سرويس توجهي نشده است. همانطور كه پيش از اين عنوان گرديد فاكتور دانه بندي سرويس ها يكي از فاكتورهاي تاثير گذار بر چهار ويژگي كيفي سيستم نهايي مي باشد كه عدم توجه به آن ممكن است به سيستم نهايي ضربه بزند. در مقاله *Inganti* [14] به فاكتور دانه بندي سرويس ها تا حدي توجه شده است. البته بايد در نظر داشت كه در اين مقاله هم دانه بندي سرويس ها بر اساس برنامه هاي كاربردي (application) موجود در زير ساخت حرفه و سيستم هاي موروثي تنظيم مي شود. به اينصورت كه اگر فعاليتي توسط واسطي از برنامه پوشش داده شود به صورت سرويس شناخته شده و اگر توسط يك واسط پوشش داده نشود، به زير فعاليت ها تجزيه مي گردد. حال اگر در سازماني سيستم هاي موروثي پاسخگوي فعاليت ها نباشند و يا سازمان سيستم هاي موروثي نداشته باشد نمي توان دانه بندي مناسبي را براي سرويس ها تعيين نمود.

1. امكان بروز افزونگي:

برخي از روش ها به اين علت كه به صورت غير متمركز به شناخت سرويس ها مي پردازند ممكناست سرويس هايي با رويهم افتادگي وظيفه مندي توليد كنند به اين صورت كه يك وظيفه منديخاص توسط دو يا چند سرويس مختلف برآورده شود كه اين امر باعث صرف زمان و هزينه بيهوده جهت اين سرويس ها مي گردد.

1. عدم تعريف دقيق محدوده سرويس ها:

برخي روش ها به تعيين دقيق وظيفه مندي هاي سرويس ها نمي پردازند. به عبارت ديگر در بسياري از روش ها گامي جهت تعيين دقيق حيطه وظيفه مندي سرويس ها وجود ندارد. اين امر علاوه بر ايجاد ابهام در ادامه فعاليت هاي فاز مدلسازي سرويس گرا خود مي تواند يكي از دلايل عمده در بروز افزونگي در سرويس ها باشد.

1. عدم كنترل پوشش دهي نيازهاي حرفه توسط سرويس هاي ارائه شده:

در بسياري از روش ها گامي جهت بررسي اين امر كه آيا مجموعه سرويس هاي شناخته شده تمامي اهداف حرفه را پاسخ مي دهند يا خير وجود ندارد. در صورتي كه سرويس هاي شناخته شده كامل نباشند يا به عبارت ديگر سرويس هاي شناخته شده تمامي اهداف را پوشش ندهند در مراحل بعد ممكن است سيستمي ناقص توليد گردد و يا در صورت تشخيص اين امر در مراحل بعد هزينه و زمان زيادي جهت رفع اين مشكل مورد نياز است. به همين علت بهتر است گامي در روش مورد نظر به منظور بررسي مجموعه سرويس هاي كانديد از نظر كامل بودن گنجانده شود.

1. عدم انجام كنترل كيفي سرويس ها:

در عمده روش ها گامي به منظور سنجش و ارزيابي كيفيت سرويس هاي شناخته شده وجود ندارد.

در صورتي كه چنين گامي در روش وجود داشته باشد علاوه بر كسب آگاهي نسبت به كيفيت سرويس هاي شناخته شده، مي توان كيفيت سيستم نهايي را بهبود بخشيد و همچنين از توسعه سرويس هايي با كيفيت پايين جلوگيري كرد.

البته بايد توجه داشت موارد فوق در تمامي روش ها وجود ندارد و اين موارد به عنوان موارد با اهميت كه در تعدادي از روش ها رعايت نشده اند قابل ذكر هستند.

**3.4 ويژگي هاي روش مورد انتظار**

پس از آشنايي با ويژگي هاي مورد انتظار سرويس ها و نقاط ضعف موجود در روش هاي جاري در اين بخش به ترسيم چارچوبي جهت ارائه روشي به منظور فرآيند تشخيص سرويس ها در حيطه معماري سرويس گرا مي پردازيم.

همانطور كه در بخش 2.4 عنوان شد، تعدادي معيار كيفي جهت ارزيابي سرويس ها وجود دارد. فرآيند تشخيص سرويس ها بايد به گونه اي بطراحي شود كه به صورت ضمني سرويس هايي با اين معيارها را توليد نمايد. به عبارت ديگر نمي توان با استفاده از روشي سرويس ها را شناسايي كرده و سپس طي مراحلي اين مجموعه سرويس ها را جهت تطبيق با اين معيارها پيكربندي نمود، بلكه فرآيند تشخيص سرويس ها بايد به گونه اي طراحي سود كه خود سرويس هايي را شناسايي كند كه داراي اين ويژگي ها باشند.

البته بايد توجه داشت كه برخي از معيارهاي كيفي سرويس ها را نمي توان طي فرآيند تشخيص سرويس ها لحاظ نمود. به عنوان مثال قابليت كشف سرويس ها در حيطه تشخيص سرويس ها نمي گنجد و براي رسيدن به اين ويژگي به اقداماتي در گام توصيف سرويس ها (service specification) نياز است. با بررسي هايي از اين دست مجموعه اي از فاكتورهاي كيفي كه در مرحله تشخيص سرويس ها قابل توجه اند بدست مي آيد كه به شرح زير هستند:

* قابليت استفاده مجدد
* قابليت تركيب
* اتصال سست
* خودمختاري
* دانه بندي مناسب براي سرويس ها

البته در مورد ويژگي خودمختاري مي توان گفت در فاز تشخيص سرويس ها بايد سرويس هايي را شناسايي كرد كه داراي حيطه مشخصي از وظيفه مندي باشند. همچنين در مورد سطح تجريد بالا (پنهان سازي اطلاعات دروني سرويس) و stateless بودن سرويس ها بايد گفت اين ويژگي ها طي فعاليت هايي در گام توصيف سرويس ها (service specification) قابل دستيابي مي باشند.

براي رسيدن به قابليت استفاده مجدد، بايد سرويس ها به گونه اي طراحي شوند كه در سيستم هاي ديگر قابل استفاده باشند. به اين منظور بايد تا حد امكان سرويس ها را طبق فرآيندهاي استاندارد حرفه تهيه كرد. به عبارت ديگر براي رسيدن به قابليت استفاده مجدد، مي توان گامي را به منظور استاندارد سازي فرآيندهاي حرفه به مجموعه گام هاي فعاليت تشخيص سرويس ها اضافه نمود. به اين ترتيب سرويس هاي شناسايي شده در حيطه حرفه هايي كه از اين فرآيند استاندارد پيروي مي كنند، قابل استفاده خواهند بود.

براي رسيدن به قابليت تركيب، سرويس ها براي ارتباط با يكديگر بايد از واسط هاي استانداردي طبعيت كنند كه اين امر نيز در گام توصيف سرويس ها بايد تامين گردد. البته در فاز تشخيص سرويس ها مي توان سرويس ها را به گونه اي شناسايي كرد كه از لحاظ منطقي قابليت تركيب با برخي سرويس هاي ديگر را داشته باسند كه با توجه به اينكه روش اصلي شناخت سرويس ها بر اساس فرآيندهاي حرفه است، استاندارد سازي فرآيندهاي حرفه به اين امر كمك شاياني مي نمايد.

يكي از ويژگي هاي اصلي سرويس ها اتصال سست آنهاست. براي رسيدن به اين ويژگي مي توان از روشي به اين شرح استفاده كرد:

ابتدا فرآيندهاي حرفه به فعاليت ها، زير فعاليت ها و .. تجزيه مي گردد. اين تجزيه تا سطحي ادامه مي كند كه در آن هر بخش تنها يك وظيفه مشخص و قابل تعريف را انجام دهد. سپس گرافي با استفاده از اين بخش ها توليد مي گردد. به اين ترتيب كه هر بخش به عنوان يگ گره در گراف نمايش داده مي شود و در صورتي كه بخش با بخش ديگري در ارتباط باشد، بين آن دو گره در گراف يالي رسم مي شود. سپس گراف توليد شده ار مي توان با استفاده از الگوريتم هايي به چندين پارتيشن تقسيم كرد به طوري كه ارتباط درون پارتيشني ماكسيسمم و ارتباط بين پارتيشني مينيمم باشد. در آخر هر كدام از اين پارتيشن ها را مي توان به صورت يك سرويس در نظر گرفت. به اين ترتيب سرويس هايي با اتصال سست شناسايي مي شوند كه داراي دانه بندي مناسبي مي باشند.

علاوه بر گام هاي پيشنهادي فوق مي توان گام هايي را جهت ارزيابي كامل بودن مجموعه سرويس هاي كانديد و همچنين كنترل كيفي سرويس ها بر اساس فرمول هاي مشخص در فرآيند تشخيص سرويس قرار داد.

**نتيجه گيري**

در اين فصل به اختصار به تبيين چارچوبي جهت راه حل پيشنهادي در زمينه تشخيص سرويس هاي نرم افزاري پرداختيم. به اين منظور ابتدا فاكتورهاي كيفي مورد انتظار سرويس ها شناسايي شده و سپس فاكتورهايي كه در گام تشخيص سرويس ها قابل دستيابي هستند مشخص گرديد. همچنين نقاط ضعف روش هاي جاري در اين زمينه تعيين شده و مورد تجزيه و تحليل قرار گرفت. در آخر نيز گام هايي جهت دستيابي به ويژگي هاي كيفي مورد نظر و پوشش نقاط ضعف ساير روش ها معرفي شد.

در ادامه اين گزارش به تعريف مسئله اصلي مي پردازيم. به عبارت ديگر به تعريف راه حل پيشنهادي و گام هاي رسيدن به آن جهت ارائه روشي به منظور تشخيص سرويس هاي نرم افزاري خواهيم پرداخت.

**فهرست منابع**

1. Thomas Erl, Service Oriented Architecture: Concepts, Technology, and Design, Prentice Hall, 2019
2. O. Zimmermann, N. Schlimm, G. Waller, and M. Pestel “Analysis and Design Techniques for Service-oriented Development and Integration,” IBM Deutschland [3] Pooyan Jamshidi, Saeed Mansour, ASIM: Toward Automatic Transformation of Enterprise Business Model to Service Model, IEEE Transactions on service computing
3. Jan-Willem Hubbers, Art Lightart, Linda Terlouw, Ten Ways to Identify Services, The SOA Magazine, 2007
4. Solmaz Boroumand, Working with SOA and RUP, The SOA Magazine, March 2008
5. Mark Endrei, Jenny Ang and Ali Arsanjani, Patterns: Service Oriented

Architecture and Web Services, IBM Corp, April 2004

1. Bernhard Borges, Service Oriented Architecture, www.searchwebservices.com
2. Iran’s Information Architecture committee: www.esoa.ir
3. Wikipedia: www.wikipedia.org
4. O. Zimmermann, P. Krogdahl, and C. Gee, “Elements of Service-Oriented Analysis and Design,” IBM® developerWorks
5. A. Arsanjani, “Service-Oriented Modeling and Architecture (SOMA)”, IBM® developer Works
6. B. Portier, “SOA terminology overview, Part 3: Analysis and design,” IBM® developer Works®
7. S. Johnson, “Modeling service-oriented solutions,” IBM® developer Works
8. S. Inaganti, and G.K. Behara, “Service Identification: BPM and SOA Handshake,” Technical Report. Business Process Trends
9. J. Amsden, “Modeling SOA: Part 1. Service identification,” IBM® developer Works®, October 2007
10. Zh. Zhang, R. Liu, and H. Yang, “Service Identification and packaging in

Service-oriented Reengineering,”

1. Linthicum, “What Level Is Your SOA?”, 2004
2. Oasis: SOA Adoption Blueprint, 2006, Available: www.oasisopen.org
3. Borges, Holly and Arsanjani, “Service Oriented Architecture”, 2004,

Available: http://searchwebservices.techtarget.com/originalContent/

1. Knorr, Rist, “10 Steps to SOA”, 2004, Available:

http://weblog.infoworld.com/article/05/11/07/45FEsoastep1\_1.htm