دانشگاه شهید بهشتی

دانشکده مهندسی برق و کامپیوتر

عنوان پایان‌نامه

ارائه روشی به منظورنیمه‌خودکارسازی مدل‌رانه ترکیب سرویس‌های وب

پایان‌نامه کارشناسی ارشد مهندسی کامپیوتر
گرایش نرم‌افزار

استاد راهنما:
دکتر فریدون شمس

توسط:
سوده فرخی

زمستان ۱۳۸۹

دانشگاه شهید بهشتی

دانشکده مهندسی برق و کامپیوتر

عنوان پایان‌نامه

ارائه روشی به منظورنیمه‌خودکارسازی مدل‌رانه ترکیب سرویس‌های وب

پایان‌نامه کارشناسی ارشد مهندسی کامپیوتر
گرایش نرم‌افزار

استاد راهنما:
دکتر فریدون شمس

توسط:
سوده فرخی

زمستان ۱۳۸۹

دانشگاه شهید بهشتی

دانشکده مهندسی برق و کامپیوتر

پایان‌نامه کارشناسی ارشد مهندسی کامپیوترگرایش نرم‌افزار

تحت عنوان:

ارائه روشی به منظورنیمه‌خودکارسازی مدل‌رانه ترکیب سرویس‌های وب

در تاریخ ۸۹/۱۱/۱۲ پایان‌نامه دانشجو، سوده فرخی، توسط کمیته تخصصی داوران مورد بررسی و تصویب نهایی قرار گرفت.

۱- استاد راهنما اول:	دکتر فریدون شمس	امضاء
۲-استاد داور (داخلی) .	دکتر حسن حقیقی	امضاء
۳- استاد داور (خارجی)	دکتر رامان رامسین	امضاء
۴- نماینده تحصیلات تکمیلی	دکتر اسلام ناظمی	امضاء

تشکر و قدردانی

سپاسگزارم

از پدر و مادر عزیزم که محبت‌ها و فداکاری‌هایشان را هیچ‌گاه نمی‌توانم جبران کنم

و

از همسرم امیر که صبورانه همراه و مشاورم در تمام مراحل زندگی بوده و هست.

سپاسگزارم

از استاد گرامی، دکتر فریدون شمس که با راهنمایی‌ها و نظرات ارزشمندشان در طول این تحقیق گره‌گشا بوده‌اند

سپاسگزارم

از سید علی یادآور نیک روش و سایر اعضای گروه تحقیقاتی ASER که مرا در انجام این تحقیق یاری دادند.

کلیه حقوق مادی مترتب بر نتایج مطالعات،

ابتکارات و نوآوری‌های ناشی از تحقیق موضوع

این پایان نامه متعلق به دانشگاه شهید بهشتی

می‌باشد.

این پایان‌نامه تحت حمایت مادی و معنوی مرکز تحقیقات مخابرات ایران با شماره قرارداد

۴۲۹۰/۵۰۰ تاریخ ۸۹/۳/۲۹ است

به نام خدا

نام و نام خانوادگی: سوده فرخی

عنوان پایان‌نامه: ارائه روشی به منظورنیمه‌خودکارسازی مدل‌رانه ترکیب سرویس‌های وب

استاد راهنما: دکتر فریدون شمس

فهرست مطالب

فهرست شکل‌ها

فهرست جدول‌ها

چکیده

امروزه تغییرات متداول در نیازمندی‌های مشتریان و محیط کسب و کار به عنوان اصلی‌ترین چالش پیش رو در توسعه سیستم‌های مقیاس وسیع مطرح است. معماری سرویس‌گرا به عنوان یک راه‌حل عملی برای رفع این مشکلات مطرح شده است، از طرف دیگر ترکیب سرویس یکی از اصول اساسی معماری سرویس‌گرا برای پاسخ‌گویی سریع به نیازمندی درخواستی با استفاده از سرویس‌های در دسترس است.

ترکیب سرویس وب این فرصت را برای سازمان‌ها فراهم کرده است که با استفاده از سرویس‌های موجود، قابلیت تطبیق با تغییرات مکرر در نیازمندی کاربران میسر گردد. از جمله نقاط ضعف روش‌ها و دیدگاه‌های ترکیب سرویس کنونی می‌توان به مواردی نظیر عدم پشتیبانی از ترکیب سرویس‌های وب RESTful و مبتنی بر SOAP به طور همزمان در یک سرویس مرکب، عدم پوشش تمامی فازهای توسعه ترکیب سرویس، در نظر نگرفتن نیازمندی‌های غیر عملکردی درخواستی برای ترکیب سرویس، عدم امکان استفاده از دارایی‌های سازمان به همراه سرویس‌های خارج از سازمان در سطح وب اشاره کرد.

هدف اصلی این تحقیق، تلاش برای نیمه خودکار سازی روند ترکیب سرویس‌های وب بر اساس نیازمندی‌های عملکردی و غیر عملکردی مطرح‌شده کاربر به صورت پویا با استفاده از ایده معماری مدل‌رانه در قالب معرفی چارچوب، فازهای فرآیند توسعه ترکیب و مدل‌های مورد استفاده در هر فاز و الگوریتم‌های مربوطه است. سعی بر آن است که در روش ارائه شده، امکان ترکیب سرویس‌های وب RESTful و سرویس‌های وب مبتنی بر SOAP با ارائه مدل متاداده ای برای توصیف همزمان آن‌ها، فراهم گردد. برای رسیدن به این هدف ضمن مطالعه روش‌های موجود، تلاش شده است که نقاط ضعف سایر روش‌ها پوشش داده شود و از نقاط قوت آن‌ها بهره‌مند شویم. بنابراین در این تحقیق روشی با نام " ترکیب مدل‌رانه و پویای سرویس‌های ناهمگن وب" یا به اختصار MDCHeS به همراه چارچوب و فازهای توسعه مربوطه ارائه خواهد شد.

کلمات کلیدی: معماری سرویس‌گرا، ترکیب سرویس وب، معماری مدل‌رانه، سرویس وب RESTful، سرویس وب مبتنی بر SOAP.

۱ فصل اول – کلیات تحقیق

۱.۱ مقدمه

امروزه معماری سرویس‌گرا[۱] به عنوان یکی از شیوه‌های معماری پیشرو در معماری راه‌حل‌های سازمانی مطرح شده و مفهوم سرویس در آن، تطابق نیاز با قابلیت‌های سازمان را به راحتی فراهم کرده است. با افزایش روز افزون نیازمندی‌های کاربران، ترکیب سرویس[۲] به عنوان یکی از مهم‌ترین اصول محاسبات سرویس‌گرا[۳] با هدف پاسخ‌گویی سریع به نیازمندی‌ها، مطرح شده است.

ترکیب سرویس یکی از نقاط قوت معماری سرویس‌گرا برای پاسخ‌گویی به یک نیازمندی جدید مطرح شده با استفاده از ترکیب کردن سرویس‌های[۴] اتمیک در دسترس است. در واقع پس از مطرح شدن نیازمندی جدید توسط کاربر دو راه‌حل برای پاسخگویی به آن نیاز وجود دارد، اول اینکه به آن نیاز بدون در نظر گرفتن سرویس‌های موجود پاسخ داده شود که مستلزم هزینه زیادی است و دوم آنکه با توجه به سرویس‌های موجود و ترکیب و استفاده مجدد از آن سرویس‌ها به سرویس درخواستی با هزینه و زمان کمتری پاسخ داده شود.

به عبارت دیگر به تجمیع[۵] سرویس‌های موجود در سازمان و یا استفاده از سرویس‌های خارج از سازمان به منظور افزایش قابلیت تطبیق سریع با تغییر در نیازمندی‌های تقاضا شده و یا نیازمندی‌های جدید مطرح‌شده، ترکیب سرویس گفته می‌شود. محیط پویا و گسترده‌ی سرویس‌های وب[۶] به همراه افزایش نیازمندی‌های کاربران از یک طرف و پیچیده بودن روند ترکیب انواع مختلف سرویس‌ها از طرف دیگر، نیاز به روشی جامع و نوین با قابلیت پاسخ‌گویی به چالش‌های روش‌های کنونی مطرح شده را در پی داشته است.

---

[۱] Service Oriented Architecture

[۲] Service Composition

[۳] Service Oriented Computing

[۴] در این تحقیق واژه‌های "سرویس" و "سرویس وب" معادل با یکدیگر در نظر گرفته شده‌اند و منظور سرویس‌وب می‌باشد.

[۵] Integrating

[۶] Web Services

۲.۱ طرح مسأله

برای ترکیب سرویس دو سناریوی اصلی شامل " توسعه ترکیب سرویس[۱]" و " مدیریت ترکیب سرویس[۲]" متصور است. در سناریوی توسعه ترکیب سرویس، خروجی نهایی یک فرآیند حرفه[۳] تولید شده با استفاده از ترکیب کردن سرویس‌های در دسترس است. این سناریو با دریافت درخواست سرویس آغاز شده و در نهایت با تولید یک سرویس مرکب[۴] قابل اجرا، به پایان می‌رسد. سناریوی مدیریت ترکیب سرویس زمانی آغاز می‌شود که کاربر قصد اجرا و مدیریت یک سرویس مرکب را داشته باشد، در این سناریو کاربر مدل اجرایی سرویس مرکب را در اختیار دارد [۱]. در این تحقیق ضمن تمرکز بر سناریوی اول یعنی توسعه ترکیب سرویس، سعی شده است بر چالش‌های روش‌های کنونی ترکیب سرویس پاسخ داده شود. با همین هدف، ویژگی‌های زیر برای روش توسعه ترکیب سرویس پیشنهادی در نظر گرفته شده است:

۱. تا حد امکان خودکار باشد :

با افزایش روزافزون سرویس‌ها، توسعه ترکیب سرویس به یک فرآیند پیچیده تبدیل شده است که انجام کلیه فازهای آن به صورت دستی، فرآیندی خسته‌کننده و همراه با خطا بوده است. از طرف دیگر تعامل با کاربر در روند توسعه ترکیب سرویس، منجر به تولید سرویس‌های مرکبی که تطابق بهتری با نیاز درخواستی کاربر دارند، می‌شود، لذا نیمه خود کارسازی این روند به همراه تعاملات مناسب و کنترل شده با کاربر، منجر به افزایش کیفیت سرویس‌های مرکب تولید شده می‌گردد.

۲. قابلیت پشتیبانی از ترکیب انواع سرویس‌های وب را دارا باشد:

ترکیب سرویس که یکی از مهم‌ترین مزیت‌های معماری سرویس‌گراست، از طریق استاندارد BPEL[۵] برای سرویس‌های مبتنی بر SOAP[۶] و توصیفات WSDL[۷] فراهم می‌گردد. از طرف دیگر با مطرح شدن سبک جدیدی از معماری سرویس‌های وب با نام REST[۸] و رواج استفاده از سرویس‌های RESTful که مبتنی بر قواعد این سبک معماری بوده و مزایایی نظیر سادگی استفاده،

---

[۱] Service Composition Development
[۲] Service Composition Management
[۳] Business Process
[۴] Composite Service
[۵] Business Process Execution Language
[۶] Simple Object Access Protocol
[۷] Web Service Description Language
[۸] REpresentational State Transfer

سبک‌وزن بودن و راحتی توسعه را به همراه دارند، چالشی جدید برای پشتیبانی از ترکیب همزمان این دو نوع سرویس مطرح شده است. از آنجا که استفاده از این نوع سرویس‌ها در سازمان‌ها به دلیل مزایایی نظیر آنچه مطرح شد، رو به افزایش است و از سوی دیگر سرویس‌های موجود در مخازن سازمان‌ها اغلب سرویس‌های مبتنی بر SOAP بوده و به عنوان دارایی‌های سازمان حائز اهمیتاند، لذا نیاز به تمهیدات لازم جهت پاسخگویی به این چالش در روش پیشنهاد شده، ضروری است.

۳. تا حد امکان پویا باشد:

روش پیشنهادی بر روی ایجاد ترکیب سرویس در زمان طراحی با توسعه مدل‌هایی که قابلیت تغییر در عناصر ترکیب در زمان طراحی را دارا هستن، تمرکز دارد. با توجه به محیط پویای سرویس‌های وب، پشتیبانی از پویایی در روال ترکیب برای یک روش ترکیب سرویس از اهمیت بسزایی برخوردار است. که در این روش سعی بر آن است که با استفاده از عناصر انتزاعی در مدل‌های سرویس مرکب، پویایی روش ترکیب تا حد مناسبی تأمین گردد.

۴. قابلیت استفاده در سازمان را داشته باشد:

با توجه به گستردگی محیط سرویس‌های وب، روش ترکیبی باید در نظر گرفته شود که قابلیت کنترل پیچیدگی فرآیند ترکیب با افزایش تعداد سرویس‌های منتشر شده در وب و همچنین سرویس‌های موجود در مخازن سازمان را دارا باشد. برای پشتیبانی از این دو ویژگی از ایده معماری مدل‌رانه[۱] و معرفی مدل‌های مورد استفاده در فرآیند توسعه و الگوریتم‌هایی برای تبدیل مدل‌ها به یکدیگر در هر فاز استفاده شده است.با استفاده از ایده‌ی معماری مدل‌رانه در روش پیشنهادی و افزایش سطح انتزاع و جداسازی منطق ترکیب از جزئیات ترکیب، تسهیل و تسریع توسعه ترکیب سرویس تا حد مناسبی فراهم شده است.

۳.۱ اهداف تحقیق

در این تحقیق تلاش می‌شود تا با ترکیب دو فناوری معماری سرویس‌گرا و معماری مدل‌رانه راه‌حلی برای نیمه خودکار سازی توسعه‌ی سرویس مرکب با قابلیت پشتیبانی پویا از ترکیب انواع سرویس‌های وب ارائه گردد. این مفهوم در شکل ۱-۱ نشان داده شده است. همچنین با معرفی دیدگاهی فازبندی شده برای این روند و پشتیبانی از نیازمندی‌های غیر عملکردی به همراه تعامل مناسب با کاربر، سعی بر افزایش کیفیت سرویس‌های مرکب از نظر تطبیق هر چه بیشتر با سرویس درخواست شده، است. روش

[۱] Model Driven Architecture

ترکیب پیشنهادی در قالب چارچوبی مدل‌رانه با هدف پاسخگویی جامع به تمامی نیازمندی‌های یک روش ترکیب سرویس ارائه می‌شود.

شکل ۱-۱: ترکیب فناروی معماری سرویس گرا و معماری مدل رانه برای توسعه سرویس مرکب

۴.۱ محدوده تحقیق

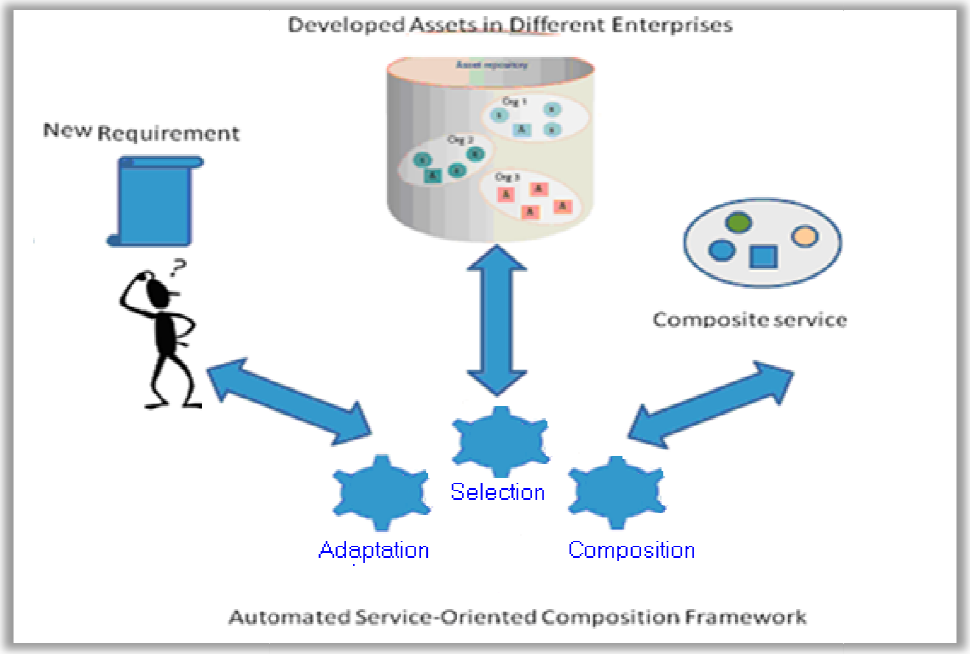
در این تحقیق تنها بر روی بخش توسعه ترکیب سرویس تمرکز شده و به مدیریت و اجرای آن پرداخته نمی‌شود. توسعه ترکیب سرویس نیز خود می‌تواند شامل ارائه دیدگاه‌ها، روش‌ها، ابزارها، زبان‌ها و چارچوب‌ها باشد که در این تحقیق تمرکز بر روی پیشنهاد روشی در قالب چارچوبی جامع برای ترکیب سرویس‌های وب قرار دارد. در این تحقیق بر روی فازهای کشف و ساخت سرویس تمرکز شده و به سایر فازها به صورت مختصر می‌پردازیم و یا از روش‌های کنونی در برخی فازها بهره می‌گیریم. روش ارائه شده از مرحله دریافت درخواست یک سرویس در قالب مدل تعریف شده‌ای در چارچوب، توسط کاربر آغاز شده و با ایجاد یک مدل قابل اجرای سرویس مرکب به صورت نیمه خودکار به پایان می‌رسد. در تمامی فرآیند ترکیب، استفاده از اصول معماری مدل‌رانه مد نظر قرار داده شده است.

لازم به ذکر است که این پروژه بر مبنای یک طرح پژوهشی[۲] با عنوان "ارائه چارچوبی مدل‌رانه جهت تولید خودکار راه‌حل‌های سازمانی مبتنی بر سرویس بر اساس دارایی‌های موجود" با سه هدف زیر با هدایت گروه ASER[۱] آغاز شده است و هدف اول و دوم از این طرح پژوهشی که مربوط به انتخاب و ترکیب سرویس‌ها می‌باشد با این پروژه تکمیل می‌گردد. در شکل ۱-۲ نمایی از اهداف این طرح پژوهشی نمایش داده شده است.

- انتخاب خودکار دارایی موجود بر اساس نیازمندی به وجود آمده (Selection)

[۱] Automated Software Engineering Research Group .http://aser.sbu.ac.ir

- ترکیب نمودن دارایی برای ساخت کاربرد جدید (Composition)
- انطباق و متناسب سازی خودکار دارایی‌های موجود (Adaptation)



شکل ۱-۲: نمایی از طرح پژوهشی [۲]

۵.۱ مراحل انجام تحقیق

مراحل تحقیق همان‌طور که در شکل ۱-۳ مشخص شده است، از مطالعه بر روی معماری سرویس‌گرا آغاز شد. سپس با توجه به اهمیت موضوع ترکیب سرویس در حیطه سرویس گرایی، بر روی این موضوع تمرکز شد. در ادامه همزمان با مطالعه روش‌های ارائه شده در ترکیب سرویس‌ها، به مطالعه انواع سرویس‌های وب پرداخته شد و در کنار آن اصول معماری مدل‌رانه به عنوان یکی از راه‌حل‌های مناسب برای استفاده به عنوان راهبرد[۱] روش پیشنهادی ترکیب سرویس، مطالعه گردید. پس از مطالعه کامل روش‌ها، دیدگاه‌ها و چارچوب‌های پیشنهاد شده برای ترکیب سرویس، دسته‌بندی آن‌ها صورت گرفت و فاکتورهای اصلی یک

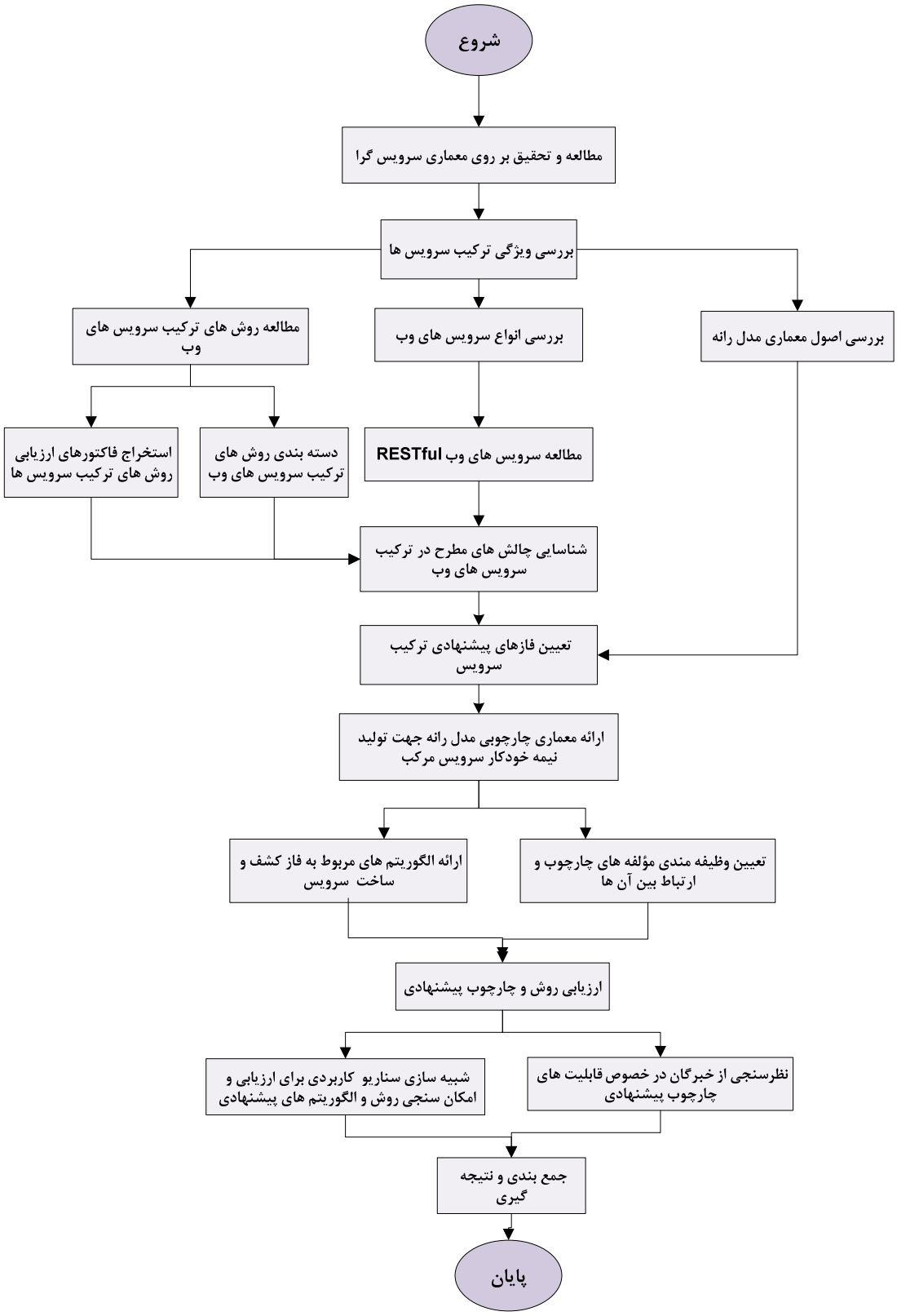
[۱] Strategy

روش ترکیب سرویس استخراج گردید. مطالعه در مورد ویژگی‌های سرویس‌های وب RESTful به عنوان یکی از انواع سرویس‌های وب در مسیر انجام تحقیق صورت گرفت در ادامه با در نظر گرفتن انواع سرویس‌های وب و روش‌های موجود، چالش‌های مطرح در این زمینه استخراج شد. سپس به منظور پاسخ‌گویی به برخی چالش‌های کلیدی نظیر پشتیبانی از انواع سرویس‌های وب، جامع بودن روش ترکیب، پویایی و پشتیبانی از نیازمندی‌های غیر عملکردی، فازهای ترکیب در روش پیشنهادی ارائه و در ادامه چارچوبی مدل‌رانه و نیمه خودکار برای پشتیبانی از ترکیب سرویس پیشنهاد گردید. با در نظر گرفتن محدودیت‌های زمانی برای این تحقیق ضمن تعیین مؤلفه‌های چارچوب و نحوه ارتباط آن‌ها با یکدیگر، بر روی الگوریتم‌های مرتبط با فازهای کشف و ساخت سرویس در این چارچوب، تمرکز گردید. سپس برای ارزیابی الگوریتم و روش پیشنهادی، سناریوی کاربردی طراحی و دنبال شد، علاوه بر این برای ارزیابی چارچوب پیشنهادی و فازهای توسعه دربرگیرنده آن نظرسنجی از خبره در خصوص قابلیت‌های چارچوب بر اساس پرسش‌نامه‌ای استاندارد در مراحل انتهایی تحقیق صورت گرفت. در پایان جمع‌بندی و پیشنهاد کارهای آینده که شامل تکمیل چارچوب پیشنهادی و پیاده‌سازی کامل آن است، انجام شد.

۶.۱ ساختار پایان‌نامه

ساختار این پایان نامه به صورت زیر است:

در فصل دوم، مفاهیم پایه‌ای مورد استفاده در این تحقیق شرح داده می‌شوند و خلاصه‌ای از روش‌های موجود در زمینه ترکیب سرویس بررسی و دسته بندی شده و نقاط قوت و ضعف هر یک از آن‌ها بیان می‌گردد. سپس در فصل سوم، مبانی روش پیشنهادی شامل معرفی چارچوب پیشنهادی به همراه مدل‌های مورد استفاده و مؤلفه‌های درگیر در آن بیان خواهد شد. در فصل چهارم، روش ترکیب پیشنهادی به همراه فازهای توسعه و الگوریتم‌های مورد استفاده در هر فاز با هدف نیمه‌خودکارسازی روند ترکیب انواع سرویس‌های وب با استفاده از معماری مدل‌رانه ارائه می‌شود. در فصل پنجم، به ارزیابی و امکان‌سنجی مدل پیشنهادی می‌پردازیم و در آخر در فصل ششم با جمع آوری و نتیجه‌گیری، تحقیق را به پایان می‌بریم.



شکل ۱-۳: مراحل انجام تحقیق

۲ فصل دوم – مفاهیم پایه و پیشینه تحقیق

۱.۲ مقدمه

در فصل قبل به طرح مسأله مورد مطالعه در این تحقیق پرداخته شد. از آن جا که برای اصطلاحات موجود در حیطه ترکیب‌سرویس‌وب ممکن است تعاریف متعددی مطرح شده باشد، در ابتدای این فصل تعریف مفاهیم اصلی مورد نیاز برای این تحقیق بیان می‌شود. در فصل‌های آینده، استفاده از این مفاهیم بر اساس تعاریف بیان‌شده در این فصل شکل گرفته است، در ادامه فصل مرور مختصری بر برخی کارهای مرتبط با این تحقیق خواهیم داشت. لازم به ذکر است که بررسی مفصل کارهای مرتبط در سمینار [۳] مربوط به این پایان‌نامه صورت گرفته است. در پایان با مقایسه و ارزیابی روش‌های مطرح شده، نقاط قوت و ضعف آن‌ها شناسایی می‌شود و با هدف پوشش برخی نقاط ضعف و بهره‌گیری از نقاط قوت شناخته شده در روش پیشنهادی در فصل سوم معرفی می‌گردد.

۲.۲ مفاهیم پایه

۱.۲.۲ معماری سرویس‌گرا

برای معماری سرویس‌گرا تعاریف مختلفی ارائه شده که هر کدام از نگاهی به تبیین خصوصیات آن پرداخته‌اند که در ادامه تعدادی از این تعاریف آورده شده‌اند:

- روشی برای طراحی و پیاده سازی نرم افزارهای گسترده سازمانی به وسیله ارتباط بین سرویس‌هایی که دارای خواص اتصال سست، دانه درشتی[۱] و قابلیت استفاده مجدد[۲] هستند[۴].
- سبکی از معماری برای ساخت نرم افزارهایی که از سرویس‌های منتشر شده در یک شبکه مانند وب استفاده می‌کند. اتصال سست بین مؤلفه‌های نرم افزاری باعث قابلیت استفاده مجدد از آن‌ها می‌شود و نرم افزارها بر مبنای سرویس ساخته می‌شوند، سرویس در اینجا به معنای پیاده سازی یک کارکرد کسب و کار خوش تعریف[۳] است که می‌تواند در فرآیندها یا نرم افزارهای مختلف مورد استفاده و فراخوانی قرار بگیرد[۵].

---

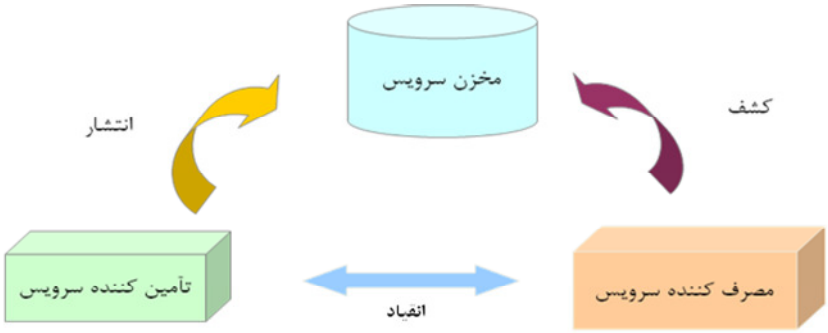
[۱] Coarse Granularity
[۲] Reusability
[۳] Well Defined

به طور کلی معماری سرویس گرا مجموعه‌ای از قواعد، الگوها و معیارها در حیطه تولید و توسعه نرم افزار و فرآیندهای سازمان است که باعث حصول قابلیت استفاده مجدد، واحدمندی[۱]، ارتباط سست و مستقل از سکو بودن[۲] در محصول می‌گردد.

معماری سرویس‌گرا یک فناوری نیست، بلکه مجموعه‌ای از اصول است که به سازمان‌ها کمک می‌کند سیستم‌های مختلف را مجتمع سازند و ساختار منعطف‌تری برای پشتیبانی از فرآیندهای حرفه فراهم می‌آورد[۶].

۱.۱.۲.۲ سرویس وب[۳]

سرویس‌های وب، برنامه‌های کاربردی خود-توصیف[۴] و مستقل از سکو[۵] هستند که می‌توانند از طریق اینترنت با استفاده از یک URL[۶] شناخته شده، فراخوانی شوند. واسط‌های عمومی و انقیادهای آن با استفاده از XML[۷] توصیف و شناسایی می‌شود. (XML زبانی شامل قوانینی برای کدگذاری مستندات به فرمت داده‌ای متنی و قابل فهم برای ماشین است[۷].) سایر برنامه‌ها از طریق روشی که در تعریف آن ارائه شده و با استفاده از پیام‌های مبتنی بر XML از طریق پروتکل‌های اینترنت با سرویس‌های وب تعامل دارند. یک مدل سرویس وب شامل سه موجودیت: تأمین‌کننده سرویس[۸]، مخزن سرویس[۹] و مصرف‌کننده سرویس[۱۰] است. شکل ۲-۱ نشان‌دهنده مدل بیان‌شده می‌باشد.



شکل ۲-۱: مدل سرویس‌وب[۷]

---

[۱] Modularity
[۲] Platform Independent
[۳] Web Service
[۴] Self-descriptive
[۵] Platform-independent
[۶] Uniform Resource Locator
[۷] Extensible Markup Language
[۸] Service Provider
[۹] Service Registry
[۱۰] Service Consumer

تأمین‌کننده سرویس، سرویس را در فرمتی استاندارد که معمولاً XML است و سپس آن را در یک مخزن مرکزی سرویس، انتشار دهد. مخزن سرویس، شامل اطلاعات بیشتری در مورد تأمین‌کننده سرویس، مانند آدرس و نحوه تماس با شرکت تأمین‌کننده و جزئیات فنی در مورد سرویس است. مصرف‌کننده سرویس، اطلاعات را از مخزن بازیابی کرده و از توصیف سرویس بدست آمده برای انقیاد[۱] به آن و یا فراخوانی سرویس‌وب استفاده می‌کند. متدهای انتشار، انقیاد و کشف در شکل ۲-۱ نشان داده شده است.

با بیانی دیگر سرویس‌های وب، نرم افزارهای کاربردی هستند که تحت وب منتشر شده و مورد فراخوانی قرار می‌گیرند، مستقل از سکو و زبان بوده و یک رهیافت کلیدی برای عینیت بخشیدن به معماری سرویس‌گرا به شمار می‌آیند.

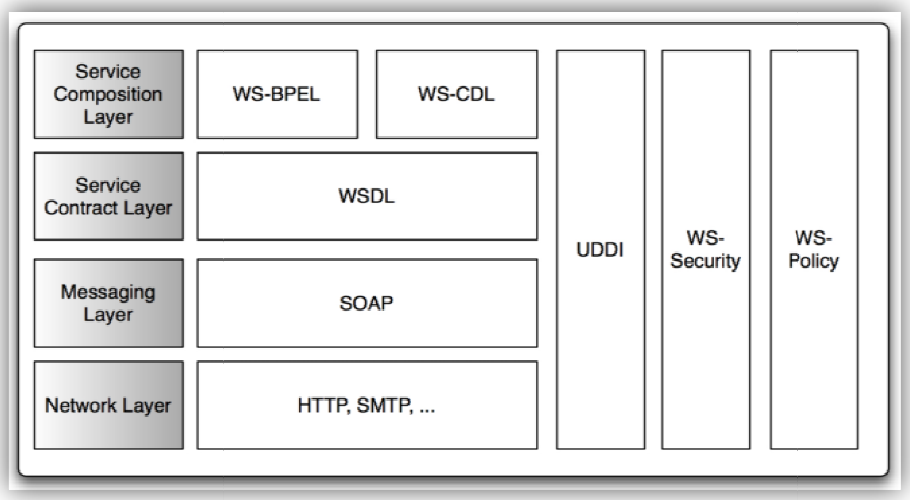
تعریف سرویس وب از نظر W3C[۲]: یک سرویس وب، نوعی سیستم نرم افزاری است که جهت تعامل ماشین با ماشین در سطح شبکه طراحی شده است و دارای یک تعریف قابل پردازش توسط ماشین با نام WSDL است. دیگر سیستم‌ها بر طبق این توصیف، با سرویس دهنده تعامل خواهند داشت، پیام‌ها توسط پروتکل SOAP و یا سایر پروتکل‌های مربوطه منتقل می‌شوند[۸].

در معماری سرویس‌گرا، سرویس توسط تأمین‌کنندگان مختلف توسعه داده می‌شود، از طریق واسط استانداردی نظیر WSDL توصیف می‌شود، در یک مخزن[۳] استاندارد قابل دسترس، نظیر UDDI[۴]، انتشار می‌یابد و می‌تواند توسط پروتکل‌های استانداردی نظیر SOAP کشف و درخواست شود[۹]. ماهیت مستقل از سکوی سرویس‌های وب فرصت توسعه فرآیندهای حرفه با استفاده از ترکیب سرویس‌های وب را برای سازمان‌ها فراهم کرده است [۱].

در شکل ۲-۲ نمایی از پشته استانداردهای سرویس وب، به همراه لایه‌های مختلف و پروتکل‌ها و استانداردهایی که برای هر لایه پیشنهاد شده است، نشان داده شده است. در ادامه با استانداردهای موجود برای هر یک از بخش‌های این پشته آشنا می‌شویم.

---

[۱] Bind
[۲] World Wide Web Consortium
[۳] Registry
[۴] Universal Description Discovery and Integration



شکل ۲-۲: پشته استانداردهای سرویس وب [۱۰]

۱.۱.۱.۲.۲ WSDL

زبانی مبتنی برXML است که به منظور فراهم کردن مدلی برای توصیف سرویس‌های وب مطرح شده است. شامل پارامترهای ورودی و خروجی، نوع داده‌ای و پروتکل‌های انتقال مورد استفاده، می‌باشد. WSDL یک واژه نامه XML برای توصیف سرویس‌های وب، مکانی که سرویس‌ها در آنجا قرار دارند و چگونگی فراخوانی آن‌هاست.

هر سرویس‌وب که بر روی اینترنت قرار می‌گیرد دارای یک فایل WSDL است که حاوی مشخصات، مکان و نحوه استفاده از آن سرویس بوده و بیانگر نوع پیغام‌هایی است که یک آن سرویس می‌فرستد و یا می‌گیرد.

نسخه کنونی WSDL، نسخه ۲.۰ است که توسط W3C پیشنهاد شده است. نسخه‌های مختلفی از WSDL از سال ۲۰۰۰ تا ۲۰۰۷ ارائه شده‌اند که در ادامه به ویژگی‌های اصلی برخی نسخه‌های آن اشاره می‌شود:

- نسخه ۱.۰: در سال ۲۰۰۰ به منظور توصیف سرویس‌های وب برای استفاده در جعبه ابزار[۱]های SOAP ارائه شد.
- نسخه ۱.۱[۲] : این نسخه که در سال ۲۰۰۱ برای توصیف سرویس‌های وب مبتنی بر SOAP ارائه شد، بر مبنای XML است و تنها قادر به پشتیبانی از دو متد get و post از HTTP[۳] است. این نسخه هنوز توسط W3C تصدیق نشده است.

---

[۱] Toolkit
[۲] http://www.w3.org/TR/wsdl
[۳] Hypertext Transfer Protocol

- نسخه ۲.۰[۱] : این نسخه که برای توصیف سرویس‌های وب مبتنی بر SOAP و REST استفاده می‌شود، بر مبنای XML بوده و قادر به پشتیبانی از چهار متد get, post, put, delete از HTTP است. همچنین در این نسخه پیاده‌سازی نیز نسبت به نسخه قبل ساده‌تر شده است. نسخه ۲.۰ نسبت به نسخه ۱.۱ تغییرات زیر را در برداشته است:
  - اضافه کردن وجوه معنایی[۲] بیشتر به زبان توصیف
  - حذف ساختار پیام[۳] از عناصر تشکیل دهنده
  - عدم پشتیبانی از سر بارگذاری عملگرها[۴]
  - تغییر نام عنصر PortTypes به Interface
  - تغییر نام عنصر Ports به Endpoints

هر فایل WSDL بر اساس ساختار نسخه ۲.۰، از عناصر زیر تشکیل شده است:

- Endpoint: تعریف آدرس یا نقطه اتصال به یک سرویس وب را مشخص می‌کند. معمولاً به صورت HTTP URL نمایش داده می‌شود.
- Binding: تعیین واسط[۵]، تعریف سبک انقیاد[۶] و انتقال از طریق این عنصر صورت می‌گیرد.
- Interface: تعریف یک سرویس وب، عملیاتی[۷] که می‌تواند انجام شود و پیام‌هایی که برای انجام آن عملیات مورد استفاده قرار می‌گیرد.
- Operation: به طور کلی یک عملیات با یک فراخوانی متد یا تابع قابل مقایسه است. در اینجا به منظور تعریف اعمال[۸] SOAP و نحوه کد گذاری پیام مورد استفاده قرار می‌گیرد.
- Types: هدف از وجود این عنصر تأمین تعریف نوع داده‌ای با استفاده از شمای XML است [۱۱] و [۱۲].

---

[۱] http://www.w3.org/TR/wsdl20/
[۲] Semantics
[۳] Message
[۴] Operator Overloading
[۵] Interface
[۶] Binding Style
[۷] Operation
[۸] Actions

SOAP ۲.۱.۱.۲.۲

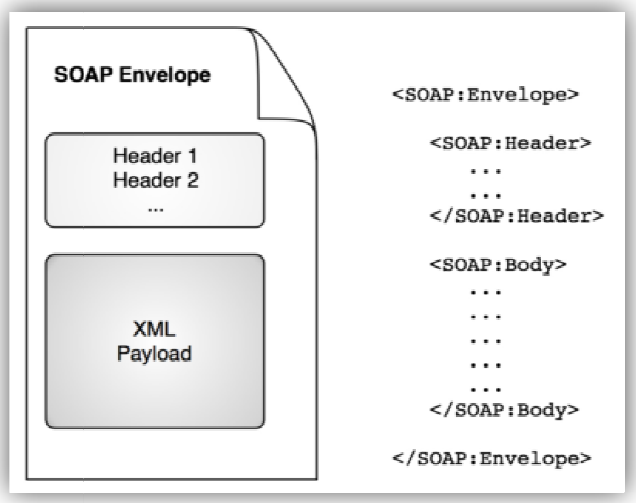
SOAP پروتکلی سبک وزن و مبتنی بر XML، برای تبادل اطلاعات در یک محیط نا متمرکز و توزیع‌شده است [۱۳]. این پروتکل در تعامل سرویس‌های وب استفاده می‌شود. در این پروتکل تبادل پیام‌ها در قالب XML بوده و بین سرویس‌های وب استفاده می‌شود. در این تبادل معمولاً از پروتکل‌هایی نظیر HTTP، HTTPS[۱] و FTP[۲] استفاده می‌شود. همچنین SOAP یک استاندارد مهم در سرویس‌های وب برای توصیف ساختار پیام‌های منتقل شونده در زمان اجرا و فراخوانی است. پیام SOAP حاوی یک مستند XML برای توصیف عملیاتی که باید اجرا شود و پارامترهایی که باید به برنامه‌های کاربردی فرستاده شوند، است [۱۴].

این پروتکل از سه بخش زیر تشکیل شده است:

- یک بسته[۳] که تعریف‌کننده یک چارچوب برای توصیف اینکه چه چیزی در یک پیام است و چگونه باید پردازش گردد، است. ساختار این بسته در شکل ۲-۳ نشان داده شده است.
- مجموعه‌ای از قوانین رمز گذاری برای بیان نمونه‌ای از نوع داده‌های تعریف شده‌ی برنامه‌های کاربردی.
- یک قرارداد[۴] برای نمایش فراخوانی روال از راه دور[۵] و پاسخ‌های آن[۱۳].

---

[۱] HTTP Secure
[۲] File Transfer Protocol
[۳] Envelope
[۴] Convention
[۵] Remote Procedure Call



شکل ۲-۳: ساختار پیام در SOAP[۱۰]

قدرت SOAP به عنوان بسته بندی کننده کد، این است که پیام SOAP به صورت فایل متنی است و به صورت عمـومی از طریـق پروتکل‌های HTTP و HTTPS فرستاده می‌شود، به همین علت می‌تواند از دیوار آتش[۱] عبور کند. این مزیـت در روش‌هـایی ماننـد CORBA[۲]، DCOM[۳] و RPC وجود ندارد [۱۴].

- سرویس مبتنی بر SOAP

منظور از سرویس وب مبتنی بر SOAP در این تحقیق، سرویس وبی است که بر اساس اصول پروتکل SOAP پیـاده سـازی شـده است. این سرویس با استفاده از WSDL توصیف می‌گردد.

۳.۱.۱.۲.۲ UDDI

UDDI مخزنی بر اساس XML و مستقل از سکو است که به منظور انتشار سرویس‌ها در اینترنت استفاده می‌شـود بـه طـوری کـه حرفه‌ها[۴] را قادر به انتشار لیست سرویس‌ها، کشف یکدیگر و تعریف چگونگی تعامل برنامه‌های کاربردی نرم‌افزاری یـا سـرویس‌هـا بـا یکدیگر در دنیای اینترنت می‌کند. یک موضوع ثبت‌شده در UDDI شامل سه بخش زیر است :

- صفحه‌های سفید[۵] : آدرس، تماس و شناسه شناخته‌شده.
- صفحه‌های زرد[۱] : طبقه صنعتی بر اساس طبقه‌بندی‌های استاندارد.

---

[۱] Firewall
[۲] Common Object Request Broker Architecture
[۳] Distributed Component Object Model
[۴] Businesses
[۵] White Pages

- صفحه‌های سبز[۲]: اطلاعات تکنیکی در مورد سرویس‌های پیشنهادی حرفه، به طور مثال نحوه دسترسی به یک سرویس وب، اطلاعات یک سرویس وب شامل آدرس، پارامترها و واسطها.

مخزن UDDI شامل متاداده[۳]هایی است که از آن‌ها برای جستجوی سرویس‌ها از روی نام، شناسه، رده، نوع و غیره استفاده می‌شود. همچنین UDDI به منظور یکپارچه‌سازی با پیام‌های SOAP و تأمین دسترسی به مستندات WSDL طراحی شده است. مخزن‌های UDDI می‌توانند عمومی یا خصوصی باشند، این مخزن پایگاه‌های داده ساختار یافته‌ای از سازمان‌ها، سرویس‌های ارائه شده و توصیف فنی آن‌ها به همراه مستندات WSDL برای آن سرویس‌ها می‌باشد. تأمین‌کنندگان سرویس‌ها، WSDL سرویس‌ها را در UDDI منتشر می‌کنند و درخواست‌کنندگان می‌توانند به آن‌ها با استفاده از SOAP متصل[۴] شوند. می‌توان UDDI را نوعی DNS[۵] برای سرویس‌های وب دانست [۱۱].

۲.۱.۲.۲ سبک معماری REST

سبک معماری REST در سال ۲۰۰۰ توسط آقای Roy Fielding[۶] به عنوان تز دکتری برای توصیف یک سبک معماری برای سیستم‌های شبکه‌ای ارائه شد[۱۵].

تعریف ارائه شده از REST در این تز بدین شرح است:

"REST قصد دارد که چگونگی رفتار یک برنامه کاربری تحت وب و خوش طراحی شده[۷] را نشان دهد. شبکه‌ای از صفحات وب (یک ماشین حالت مجازی) که در آن کاربر از طریق برنامه کاربردی پیش می‌رود و با انتخاب لینک‌ها (انتقال حالت) ، و نتیجه صفحه بعدی (به نمایندگی از حالت بعدی برنامه کاربردی) به کاربر منتقل شده و برای استفاده از آن به کاربر ارائه می‌گردد [۱۶]."

اصول REST به توسعه وب گسترده جهانی[۸] (WWW) بر می‌گردد. این اصول عبارتند از: ۱. موجودیت‌ها و وظیفه‌مندی‌ها به صورت منابع مدل می‌شوند و توسط URL شناسایی می‌شوند. ۲. دسترسی و دست‌کاری منابع از طریق عملیات شناخته شده و

---

[۱] Yellow Pages
[۲] Green Pages
[۳] Meta Data
[۴] Join
[۵] Domain Name System
[۶] http://roy.gbiv.com/
[۷] Well-designed
[۸] World Wide Web

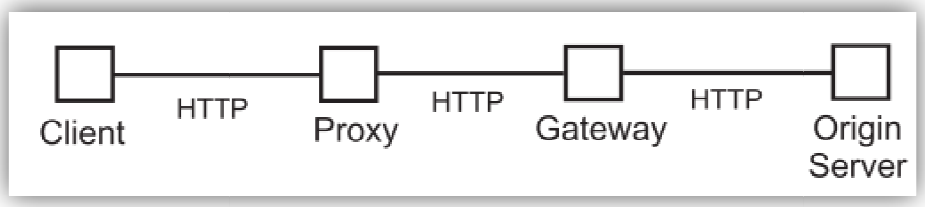
استاندارد HTTP (GET, POST, PUT, DELETE) است. ۳. مؤلفه‌های سیستم از طریق عملیات واسط استاندارد معرفی شده و تبادل نمایش[۱] منابع با یکدیگر تعامل دارند[۱۷].

همان‌طور که گفته شد یکی از اصول مرکزی در طراحی مبتنی بر REST مجموعه‌ای از عملیات تبدیل حالت است که در هر سیستم ذخیره و بازیابی اطلاعات وجود دارد. این عملیات عبارتند از" ایجاد"،" خواندن"،" به هنگام سازی "و "حذف" که به این مجموعه عملیات به طور مختصر CRUD[۲] گویند. جامعه وب ۲.۰ یک نگاشت غیررسمی از عملیات CRUD را روی دستورهای فراهم شده توسط HTTP یا همان متدهای HTTP انجام داده است که در آن،" ایجاد "معادل متد POST، "خواندن" معادل متد GET، "به هنگام سازی "معادل متد PUT و "حذف" معادل متد DELETE از پروتکل HTTP است. این متدها، عملیات CRUD درخواست شده روی یک منبع را که توسط URL آن تعریف شده است، مشخص می‌کنند [۱۸]. تعداد زیادی از شرکت‌های اینترنتی نظیر گوگل، یاهو، آمازون و غیره در حال توسعه سرویس‌های RESTful هستند که دسترسی برنامه‌های کاربردی به داده‌ها و منابع را فراهم می‌کند. برخلاف سرویس‌های مبتنی بر SOAP، این نوع سرویس‌ها معمولاً با استفاده از مستندات متنی غیر رسمی نظیر HTML، PDF و غیره توصیف می‌شوند [۱۹].

شکل ۲-۴ عناصر معماری و نحوه ارتباط آن‌ها در سبک معماری REST را نشان می‌دهد. در شکل زیر عناصر واسط نظیر proxy و دروازه[۳] اختیاری‌اند و معمولاً به منظور انجام کنترل دسترسی، ذخیره[۴] و یا انواع ترجمه پروتکلی به سبک معماری REST افزوده می‌شوند. به طور مثال دروازه می‌تواند در زمانی که چندین کار سپار[۵] به طور همزمان به کارگزار[۶] متصل شده‌اند در بین راه قرار بگیرد و درخواست‌ها را در خود نگه دارد و یا با قرار دادن proxy در این حالت اعتبار سنجی کارسپارها انجام می‌گیرد[۲۰].

---

[۱] Representation
[۲] Create-Read-Update-Delete
[۳] Gateway
[۴] Caching
[۵] Client
[۶] Server



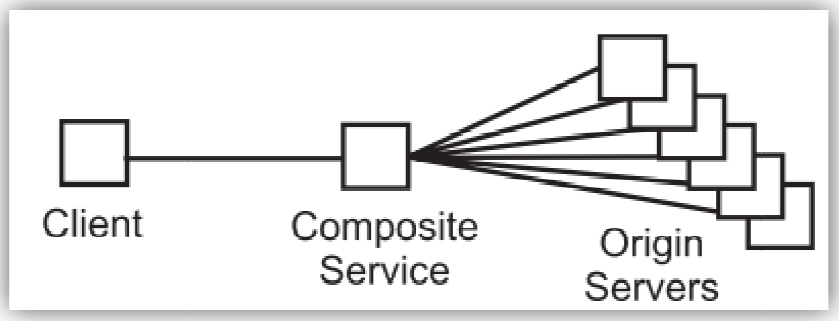
شکل ۲-۴: عناصر معماری و نحوه ارتباطشان در REST[۲۰]

- سرویس وب RESTful

سرویس وبی که بر اساس اصول معماری REST پیاده‌سازی شده است، سرویس مبتنی بر REST یا RESTful نامیده می‌شود . این نوع از سرویس وب از یک URL برای هر منبع بهره می‌برند و تنها از متدهای HTTP استفاده می‌کنند و چون از هیچ پروتکل اضافی برای تعامل با مشتریان استفاده نمی‌کند بسیار سبک وزن‌تر از سرویس‌های مبتنی بر RPC است. سادگی قوانین طراحی و راحتی تبعیت از آن‌ها، سبک وزن بودن REST و به همراه تطابق ذاتی آن با HTTP باعث گسترش روزافزون استفاده از این نوع سرویس‌وب شده است.

- سرویس مرکب RESTful

همان‌طور که در شکل ۲-۵ نمایش داده شده است، یک سرویس مرکب RESTful نوعی خاصی از عنصر واسط است که بر خلاف دروازه و proxy، درخواست‌ها را به سادگی به سمت کارگزار ارسال نمی‌کند بلکه ممکن است یک درخواست را تجزیه کند به طوری که با فراخوانی بیش از یک کارگزار رسیدگی گردد.



شکل ۲-۵: سرویس مرکب RESTful [۲۰]

۱.۲.۱.۲.۲ ویژگی‌های سرویس‌های RESTful

ویژگی‌های سرویس‌های RESTful عبارتند از:

- کار سپار-کارگزار: همان‌طور که درشکل ۲-۴ نشان داده شده است، سرویس‌های وب RESTful بر اساس ارتباط کار سپار-کارگزار پیاده‌سازی شده‌اند.
- بدون حالت: بدین معنی که هر درخواست از کار سپار به کارگزار باید شامل تمامی اطلاعات لازم برای درک درخواست باشد و نمی‌تواند از هیچ سابقه‌ی ذخیره‌شده در کارگزار بهره برد. حالت یک کار سپار یا به طور کامل توسط خود کار سپار مدیریت می‌شود و یا در منابع محصور سازی[۱] می‌گردد.
- آدرس پذیری: منابع از طریق URL قابل آدرس دهی هستند. URL هر سرویس باید تمامی اطلاعات لازم به منظور شناسایی یک منبع یکتا را شامل باشد.
- واسط یکنواخت: همه منابع با یک واسط عمومی قابل دسترس هستند[۱۶] از طریق چهار متد HTTP که هر کدام مشابه یکی از عملیات CRUD هستند و بر روی همه منابع با مفهومی یکسان اعمال می‌شوند:
  - POST: یک منبع جدید تحت URL داده شده ایجاد می‌کند. (معادل create)
  - GET: بازیابی نمایش حالت کنونی یک منبع، بدون هیچ اثر جانبی می‌تواند بارها تکرار شود (معادل read)
  - PUT: بروز رسانی حالت منبع موجود (و یا مقدار دهی اولیه به حالت یک منبع جدید در صورتی که با URL داده شده قبلاً یافت نشده باشد) (معادلupdate)
  - DELETE: آزاد کردن حالت اختصاص داده شده به یک منبع و باطل کردن URL متناظر آن منبع. (معادلdelete)[۲۱]

هر یک از متدهای بالا، بر روی یک منبع با استفاده از درخواست-پاسخ همگام HTTP فراخوانی می‌شوند. بنابراین نیازی به الگوهای نا همگام تبادل پیام که توسط BPEL/WSDL پشتیبانی می‌گردد، نیست. همچنین، از آنجایی که مجموعه عملیات تنها به چهار متد معرفی شده در بالا، محدود می‌گردد، لذا نیازی به بر شمردن صریح آن‌ها از طریق یک زبان توصیف واسط نیست [۲۲].

---

[۱] Encapsulate

۲.۲.۱.۲.۲ مزایای سرویس‌های RESTful

با توجه به ویژگی‌های سرویس‌های RESTful می‌توان موارد زیر را به عنوان مزایای اصلی این سرویس‌ها به شمار آورد:

۱. سبک‌وزن بودن به دلیل بهره‌گیری مستقیم تنها از متدهای HTTP بدون نیاز به XML های غیرضروری و محصور سازی برنامه‌های کاربردی ورودی و خروجی‌ها.

۲. وابستگی کمتر به نرم‌افزارها و مکانیزم‌هایی که لایه‌های اضافی SOAP را بر روی HTTP پیاده‌سازی می‌کنند.

۳. بالاتر بودن کارایی این نوع سرویس‌ها به دلیل طبیعت سبک‌وزن بودن و از اینرو مناسب بودن برای وب ۲.۰.

۴. راحتی توسعه و استفاده.

۵. راحتی دسترس‌پذیری به دلیل استفاده از URL.

۶. مقیاس‌پذیری به دلیل پشتیبانی از ذخیره (پاسخ درخواست GET قابل ذخیره می‌باشد) و توازی بر روی URLها.

۷. تأمین یک راهکار ساده برای پشتیبانی از توازن بار بر اساس بخش‌پذیری URL.

۸. داشتن ماهیت اعلانی در مقابل ماهیت دستوری [۱۷].

۳.۲.۱.۲.۲ دسته‌بندی انواع سرویس‌های وب RESTful

سرویس‌های وب RESTful به سه دسته کلی تقسیم می‌شوند که عبارتند از:

۱. **سرویس مجموعه‌ی منابع**[۱]: این نوع سرویس، به مجموعه‌ای از منابع دامنه نگاشت شده‌اند. اگر سناریوی خرید اینترنتی را در نظر بگیریم، منابع مرتبط با یک مجموعه از مشتریان، یا یک مجموعه از سفارشات در این نوع سرویس قرار می‌گیرند. این نوع از سرویس‌ها از تمامی چهار عملیات HTTP پشتیبانی می‌کنند.

۲. **سرویس منبع انفرادی**[۲]: منابع انفرادی دامنه، می‌توانند با این نوع سرویس مدل شوند. در سناریوی خرید اینترنتی منبع مرتبط با یک مشتری یا یک خرید به این دسته از سرویس تعلق می‌گیرد. این دسته از سه عملیات HTTP (GET, PUT وDELETE) پشتیبانی می‌کند. عملیات POST از آنجا که URL مربوط به این منبع منفرد قبلاً ایجاد شده است، در این نوع سرویس کاربرد ندارد.

---

[۱] Resource Set Service

[۲] Individual Resource Service

۳. **سرویس انتقالی**[۱]: برخلاف دو دسته معرفی شده در بالا، سرویس‌های کمی به این دسته تعلق می‌گیرند. برخی از سرویس‌ها که ماهیت گذرا یا تبدیلی دارند مربوط به این دسته هستند. وظیفه‌مندی این نوع سرویس‌ها کمتر به عنوان سرویسی که با منبعی سروکار دارد، معرفی می‌شود. به عنوان مثال در سناریوی خرید اینترنتی ثبت پرداخت[۲] یا ارسال سفارش[۳] از این نوع سرویس است. این نوع سرویس تنها از عملیات POST پشتیبانی می‌کند [۱۷].

۴.۲.۱.۲.۲ مقایسه سرویس‌های وب مبتنی بر SOAP با RESTful

از آنجا که در این تحقیق هر دو نوع سرویس مورد استفاده قرار می‌گیرند، در این بخش مقایسه‌ای از ویژگی‌های این دو نوع سرویس ارائه می‌گردد:

- عدم وجود استاندارد رسمی برای سرویس‌های وب RESTful برخلاف سرویس‌های وب مبتنی بر SOAP. زبانی که برای توصیف سرویس‌های RESTful استفاده می‌شود WADL[۴] نام دارد که در این فایل تعریف می‌شود که در یک سرویس چه منابعی وجود دارد، چگونه نمایش داده شده‌اند، چگونه به هم متصل شده‌اند و چه متدهایی از HTTP برای آن‌ها قابل استفاده است.
- REST از تمامی متدهای HTTP بهره می‌برد در حالی که SOAP تنها از متد POST برای انتقال پیام‌های SOAP استفاده می‌کند.
- در مقابل ماهیت دستوری و جنبه عملیات-محور[۵] سرویس‌های مبتنی بر SOAP، سرویس‌های RESTful ماهیت اعلانی دارند و منبع-محور[۶] اند [۱۹]. ماهیت اعلانی بدین معناست که سرویس‌های RESTful بر روی توصیف منابع نسبت به چگونگی انجام توابع تمرکز دارند. به بیانی دیگر در زمان ساخت سرویس برای یک سیستم معین، دیدگاه اعلانی بر این تمرکز دارد که چه منابعی مورد نیاز است و چگونه این منابع باید نمایش داده شوند، در حالی که دیدگاه دستوری بر این تمرکز دارد که چه عملیاتی مورد نیاز است و این عملیات بر روی چه ورودی خروجی اعمال می‌شوند. دیدگاه اعلانی گزینه بهتری برای ساخت یک سیستم بر اساس معماری سرویس گرای مقیاس پذیر است.

---

[۱] Transitional Service
[۲] Submit Payment
[۳] Ship Order
[۴] Web Application Description Language
[۵] Operation-Centric
[۶] Resource-Centric

- سرویس‌های وب RESTful اغلب بهتر از سرویس‌های وب مبتنی بر SOAP با HTTP و جستجو گرهای وب مجتمع می شوند.
- سرویس‌های وب RESTful به پیام‌های XML یا تعریف WSDL نیازی ندارند.
- بر خلاف سرویس‌های مبتنی بر SOAP، BPEL از سرویس‌های RESTful پشتیبانی نمی‌کند[۱۷].

۲.۲.۲ ترکیب سرویس

به فرآیند توسعه یک سرویس مرکب از ترکیب سرویس‌های موجود ترکیب سرویس گفته می‌شود.

این فرآیند معمولاً به طور کلی شامل مراحل زیر است:

۱. بررسی درخواست یک سرویس مرکب از طرف کاربر و تحلیل نیازمندی‌های کاربر
۲. کشف سرویس‌های قابل استفاده برای پاسخگویی به نیازهای عملکردی، این سرویس‌ها می‌توانند هم از منابع داخل سازمان که در مخزن سرویس موجود است و یا از منابع سایر سازمان‌ها از طریق اینترنت کشف شوند.
۳. انتخاب سرویس مورد نظر از بین سرویس‌های مرکب کاندیدا بر اساس نیازهای غیر عملکردی نظیر کارایی، دقت و کیفیت
۴. تولید توصیف سرویس مرکب
۵. اجرای سرویس مرکب

به عنوان مثالی از سرویس مرکب می‌توان رزرواسیون پرواز، اتومبیل و هتل را در قالب یک سرویس مرکب نام برد. از مزایای ترکیب سرویس می‌توان به توسعه سریع برنامه‌های کاربردی، استفاده مجدد از سرویس و تکمیل سرویس‌های پیچیده اشاره گرد [۲۳]. به علت وجود فضای وسیع جستجوی سرویس‌ها، انواع مختلف سرویس‌ها برای مقایسه و تطبیق و راه‌های مختلف ساخت سرویس، فرآیند ترکیب سرویس بسیار پیچیده و پویا است به طوری که نمی‌توان آن را به صورت دستی انجام داد [۱]. اما از چالش‌های مطرح در خودکار سازی فرآیند ترکیب سرویس‌ها، ایجاد خودکار کنترل جریان برای ترکیب و رسیدگی کردن به موضوع ناهمگنی احتمالی داده در فرآیند ترکیب است [۱۹].

۱.۲.۲.۲ سرویس مرکب

سرویس مرکب به سرویسی اتلاق می‌شود که از ترکیب چندین سرویس ایجاد شده است و این سرویس‌ها بر اساس یک شمای ترکیب با یکدیگر تعامل دارند. ممکن است سرویس‌های موجود در سرویس مرکب، در مکان‌های مختلف پیاده سازی شده باشند و در زمینه‌های مختلف اجرا شوند، اما لازم است همگی برای دست‌یابی به یک هدف، با یکدیگر تعامل کنند[۲۴].

به عبارت دیگر یک سرویس مرکب، می‌تواند به عنوان ترکیبی از فعالیت‌ها (که خود می‌توانند سرویس اتمیک یا سرویس مرکب) باشند، در نظر گرفته شود که بر اساس یک ترتیب از پیش تعریف شده فراخوانی می‌شوند و تماماً اجرا می‌شوند. به این صورت یک سرویس پیچیده در واقع مشابه یک فرآیند حرفه رفتار می‌کند[۲۵].

BPEL ۲.۲.۲.۲

BPEL یک زبان مبتنی بر XML است که در سال ۲۰۰۲ برای توصیف منطق هماهنگی و کنترل سرویس‌های وب در یک فرآیند حرفه ارائه شد [۲۶]. به بیان دیگر BPEL زبانی برای ترکیب فرایند گرا[۱]ی سرویس‌های وب است [۲۳].

BPEL دارای مشخصات زیر است:

- بر اساس قوانین معماری سرویس‌گرا بوده و از WSDL برای توصیف واسط سرویس‌ها استفاده می‌کند[۲۶].
- شامل ساختارهای کنترل جریان و شرط‌های انشعاب است.
- زبانی مستقل از سکو و مبتنی برXML است.
- قابلیت پوشش مواردی نظیر فرآیندهای تو در تو[۲]، الحاق و شکست زیر فرآیندها [۲۷] را داراست.

BPEL دارای چندین گروه از عناصر سازنده است که عناصر اصلی آن عبارتند از:

- <process>: برای آغاز یک فرآیند.
- <partnerLink>: برای تعریف سرویس‌های شرکت‌کننده در ترکیب.
- <invoke>...<receive>: برای فراخوانی همگام و غیر همگام.
- <variable>, <assign>, <copy>: برای دست‌کاری متغیرها و نتایج میانی.
- <scope>, <faultHandlers>: برای اداره کردن خطاها.
- <sequence>, <flow>: برای اجرای موازی یا ترتیبی.

---

[۱] Process-Oriented
[۲] Nested

- <switch>: برای کنترل منطق [۲۳].

۳.۲.۲.۲ مشاپ[۱]های وب

مشاپ‌های وب از کاربردهای مهم وب ۲.۰ هستند که امروزه در بین افرادی که به ایجاد و فراهم آوردن سرویس‌های وب روی وب، علا قمند هستند رواج یافته است. اصطلاح مشاپ از موسیقی پاپ سرمنشأ گرفته است و به عمل تولید یک آهنگ جدید با سبکی جدید از ترکیب دو یا چند قطعه موسیقی موجود (عموماً در سبک‌های مختلف) اتلاق می‌شود. در زمینه اینترنت مشاپ یک برنامه کاربردی تحت وب است که داده‌های دو یا چند منبع را ترکیب و به صورت یک ابزار یکپارچه ساده در می‌آورد. از این رو به مشاپ‌ها برنامه‌های وب مرکب نیز گفته می‌شود. مشاپ‌ها دارای ویژگی‌های زیر هستند:

- کاربران هدف محدود هستند.
- گذرا هستند و طول عمر کوتاهی دارند.
- نیاز به یک راه حل آنی و نسبتاً خوب دارند.
- توسط کاربران نهایی خود توسعه داده می‌شوند[۱۸] و [۹].

در حال حاضر، مشاپ‌ها به عنوان یک سکوی قدرتمند برای توسعه برنامه‌های کاربردی تحت وب مطرح شده‌اند که چندین مجموعه جریان داده را به صورت یک تجربه کاربری یکپارچه ترکیب می‌کنند [۱۸].

اگر بخواهیم مقایسه‌ای بین مشاپ و ترکیب سرویس‌های RESTful داشته باشیم، می‌توان به موارد زیر اشاره کرد:

- یک مشاپ در سطح مجتمع سازی سرویس‌های داده‌ای محدود است و اکثر استفاده از سرویس‌های RESTful در مشاپ‌ها به واکشی کردن داده از منابع دور از دسترس محدود می‌شود.
- مشاپ‌ها معمولاً در بروز رسانی و دست‌کاری داده‌های منابع دور از دسترس یا منابع دیگر درگیر نمی‌شوند.

---

[۱] Mashup

- **تفاوت مشاپ با سرویس وب مرکب RESTful**

مشاپ می‌تواند به عنوانی ترکیبی که در لایه واسط کاربری اعمال می‌شود، در نظر گرفته شود. مشاپ کردن سرویس‌های RESTful تأکیدی بر قابلیت استفاده از ترکیب را ندارد بلکه تنها سادگی ایجاد را مد نظر قرار می‌دهد. بر خلاف آن سرویس وب مرکب RESTful با هدف استفاده مجدد ایجاد شده و لزومی به تأمین یک واسط کاربری برای ترکیب خود ندارد. تفاوت دیگر در این است که اکثر مشاپ‌ها تنها مجتمع سازی قابل خواندنی از داده‌هایی با منابع مختلف را انجام می‌دهند در حالی که سرویس‌های وب RESTful اجازه دسترسی و انتقال حالت از کار سپار برای بروز رسانی منابع ترکیب شده به همراه انتقال حالت از میان کارگزارهای ترکیبی را می‌دهند [۲۰].

از آنجا که هدف روش ترکیب پیشنهادی ایجاد سرویس مرکبی است که ضمن قابلیت استفاده مجدد، بتواند هر دو نوع از سرویس‌های وب مبتنی بر SOAP و RESTful را شامل شود، لذا به طور کلی هدف این تحقیق با هدف توسعه مشاپ‌های موجود متفاوت است.

۳.۲.۲ معماری مدل‌رانه

از آنجا که روش پیشنهادی در این تحقیق از اصول معماری مدل‌رانه بهره برده است، لازم است در این بخش مفاهیم اصلی مرتبط با مفهوم معرفی گردد. معماری مدل‌رانه دیدگاهی پیشنهاد شده توسط OMG[۱] برای توسعه نرم‌افزار و نوشتن توصیف برنامه‌های کاربردی است. معماری مدل‌رانه خود نوعی از توسعه‌ی مدل رانه به شمار می‌آید که در آن تبدیل مدل‌ها[۲] با ارائه فرآیندی خودکار صورت می‌گیرد. در شکل ۲-۶ نمایی از تبدیل مدل‌ها در توسعه مدل‌رانه نمایش داده شده است. دیدگاه معماری مدل‌رانه توصیف عملیات یک سیستم را از جزئیاتی که سیستم از قابلیت‌های سکوهایش استفاده می‌کند، جدا می‌سازد. این دیدگاه از مدل‌های انتزاعی برای مشخص کردن منطق برنامه‌های کاربردی استفاده می‌کند. این مدل‌ها، ایجادکننده مدل‌های جدیدی هستند که نیازمندی‌های سیستم را در یک سکوی خاص، بیان می‌کنند [۲۸]. معماری مدل‌رانه بر روی کارایی و کیفیت توسعه نرم‌افزار تمرکز دارد. این معماری مشخص می‌کند که در طی فرآیند توسعه لازم است سه مدل CIM[۳]، PIM[۴] و PSM[۵] (که در

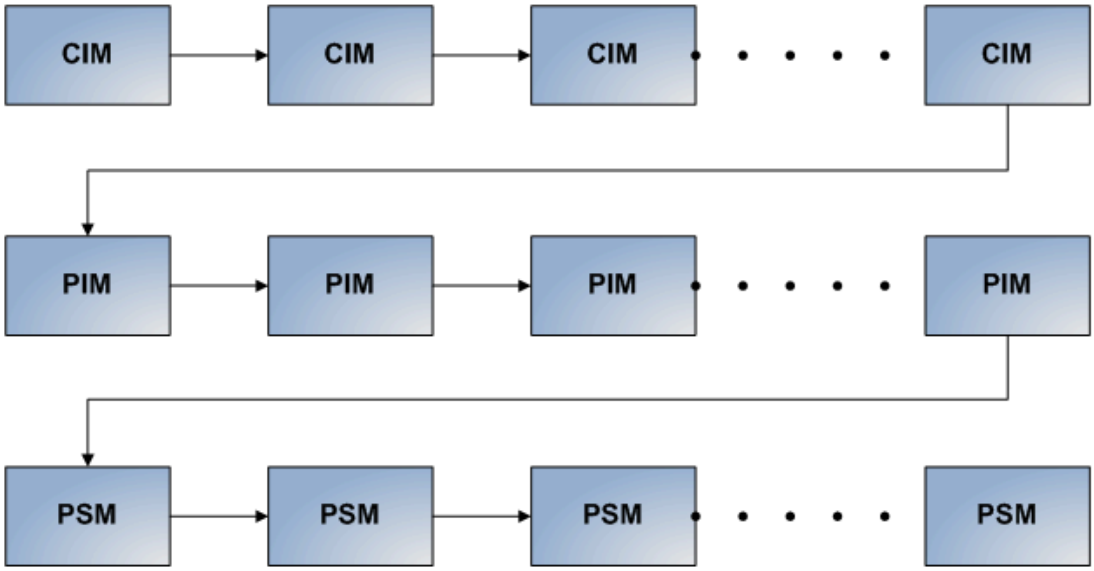
---

[۱] Object Management Group
[۲] Models Transformation
[۳] Computation Independent Model
[۴] Platform-Independent Model
[۵] Platform-Specific Model

ادامه توصیف می‌شوند) ایجاد گردند. الگوی اصلی در این معماری تبدیل مدل‌ها از مدل مستقل از محاسبه به مدل مستقل از سکو (PIM) و سپس به مدل خاص سکو (PSM) است [۲۹]. شکل ۲-۶ فرآیند اصلی تبدیل مدل‌ها در معماری مدل رانه را نشان می‌دهد.

شکل ۲-۶: فرآیند اصلی در معماری مدل رانه [۲۸]

لازم به ذکر است که ممکن است در یک سیستم پیچیده فرآیند تبدیل مدل‌ها از شکل ۲-۶ پیچیده‌تر باشد، به طور مثال شکل ۲-۷ فرآیند معماری مدل‌رانه در یک سیستم پیچیده را نشان می‌دهد. در این حالت هر مدل می‌تواند سطوح انتزاع متفاوتی داشته باشد و در این مراحل به طور مثال یک PIM با سطح انتزاع بالاتر به یک PIM با سطح انتزاع پایین‌تر و با توصیفات جزئی‌تر تبدیل شود و همچنین برای PSM که مرحله به مرحله به مدلی با توصیفات دقیق‌تر مرتبط با سکوی مورد نظر غنی‌تر[۱] گردد[۲۸].



شکل ۲-۷: فرآیند تبدیل مدل‌های یک سیستم پیچیده در معماری مدل رانه [۲۸]

[۱] Enrich

با استفاده از معماری مدل‌رانه، طراحی سرویس‌وب و فرآیند حرفه می‌تواند با تمرکز بر منطق حرفه بدون در نظر گرفتن فناوری‌های سکوهای زیرین[۱]، با PIM انجام گیرد و پیاده‌سازی آن‌ها به عنوان PSM ارائه شود که مرتبط با سکوی اجرایی یا زبان توصیف فرآیند حرفه است [۳۰]. یکی از مهم‌ترین مزیت‌های معماری مدل‌رانه توانایی تبدیل PIM به مدلی است که قابلیت اجرا بر روی فناوری‌های متعددی را دارد، معماری مدل‌رانه با در نظر گرفتن این فرض که فناوری بسیار ناپایدار[۲] است، لذا با تبدیل خودکار PIM به PSM می‌توان در زمان و هزینه در کنار بهبود کارایی در مراحل پیاده‌سازی، مجتمع سازی، نگهداری و تست صرفه جویی کرد[۳۱].

از نقطه نظر ترکیب سرویس، استفاده از دیدگاه مدل‌رانه مزایایی نظیر بهبود بهره‌وری[۳]، پشتیبانی از مهندسی تصمیم‌گیری[۴] و استفاده مجدد از دانش را به دنبال دارد. همچنین با تمرکز بر روی مدل از فناوری‌ها و پیاده‌سازی‌های سطح پایین دوری می‌کنیم و بر روی مفهوم دامنه، شرایط مسأله، بهبودها مورد انتظار و سیستم مورد نظر تمرکز کرده و تصمیمات بهتری اتخاذ می‌کنیم در نتیجه مشکلات محیط زودتر و بهتر شناخته شده و سیستم با تحلیل و طراحی کامل بدست آید و در این صورت فرآیندهای مستعد خطا را می‌توان به صورت سیستماتیک خود کارکرد [۲۷]. در ادامه با مدل‌های مورد استفاده در این معماری آشنا می‌شویم.

CIM ۱.۳.۲.۲

این مدل انتزاعی‌ترین مدل مورد استفاده در معماری مدل‌رانه به شمار می‌آید. مدل مستقل از محاسبه، بر روی محیط و نیازمندی‌های سیستم تمرکز دارد به همین دلیل، مدل دامنه نیز نامیده می‌شود [۳۱].

کاربری که از CIM استفاده می‌کند نیازی به داشتن دانش در زمینه‌ی جزئیات ساختار سیستم ندارد. در حقیقت CIM، نقش مهمی را در پر کردن فاصله بین متخصص دامنه یا متخصص حرفه با طراح سیستم بر عهده دارد [۲۸]. در این مدل راه حل مسأله بیان نمی‌شود و تنها بر روی توصیف نیازمندی‌ها توسط کاربر در فضای مسأله در ارتباط است و وارد فضای جواب مسأله نخواهد شد. در واقع در این مدل جزئیات ساختاری و فرآیند سیستم پنهان است و یا هنوز مشخص نشده است [۲۸].

---

[۱] Underlying Platform Technology
[۲] Volatile
[۳] Productivity
[۴] Decision Making

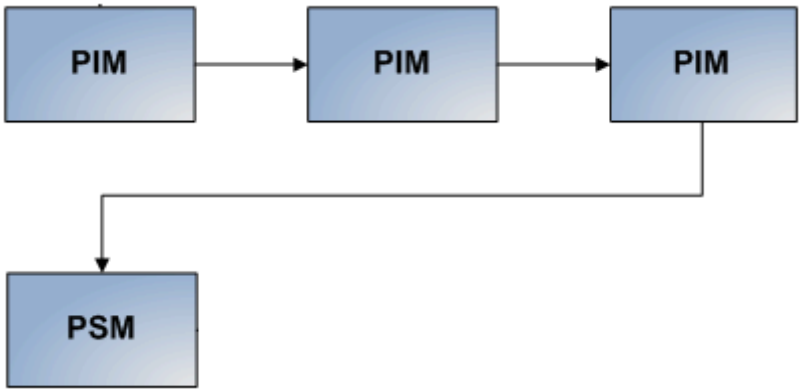
PIM ۲.۳.۲.۲

مدل مستقل از سکو، توصیفی از سیستم را از نقطه نظر مستقل از سکو، ارائـه مـی‌دهـد. ایـن مـدل بـر روی عملکردهـای سیسـتم متمرکز شده و بخش‌هایی از توصیف کامل سیستم را که از سکویی به سکوی دیگر تغییـر نخواهـد کـرد نشـان مـی‌دهـد [۳۱]. بـا توجه به ویژگی مستقل از سکوی این مدل، قابلیت استفاده با چندین سکوی مختلف در یک زمینه را دارد [۲۸] و [۲۹].

PSM ۳.۳.۲.۲

مدل خاص سکو، سیستم را با ترکیب توصیفات ارائه شده در PIM به جزئیـات مـرتبط بـا سـکویی کـه سیسـتم بـر روی آن اجـرا می‌شود، توصیف می‌کند[۳۱]. منظور از سکو در این تحقیق، فناروی و زبـان مـورد اسـتفاده بـرای توصـیف سـرویس مرکـب نظیـر BPEL است.

پس از آشنایی با مفاهیم معماری مدل‌رانه و معرفی مدل‌های مورد استفاده در آن، اگر بخـواهیم نمـایی از فرآینـد مـورد اسـتفاده در این تحقیق و روند تبدیل مدل‌ها در روش پیشنهادی را به طور مختصر معرفی کنـیم، ایـن رونـد در شـکل ۲-۸ نمـایش داده شـده است. طبق شکل سطوح انتزاع مختلفی از PIM مورد استفاده قرار گرفته و در نهایت بـه PSM تبـدیل مـی‌گـردد. در فصـل‌هـای آینده این فرآیند، مدل‌های مورد استفاده و نحوه تبدیل آن‌ها به یکدیگر به طور کامل تشریح می‌گردد.



شکل ۲-۸: فرآیند پیشنهادی این تحقیق در تبدیل مدل‌ها

۳.۲ بررسی کارهای مرتبط

پس از آشنایی با مفاهیم مرتبط با معماری سرویس گرا، ترکیب سرویس، معماری مدل‌رانه و سبک معماری REST، در ادامه به معرفی و بررسی روش‌های ارائه شده به منظور ترکیب سرویس‌ها می‌پردازیم. لازم به ذکر است از آنجا که مرور روش‌های مرتبط، در گزارش سمینار[۳] به طور کامل و مفصل بیان شده است، در اینجا تنها خلاصه‌ای از هر روش به ترتیب سال ارائه روش، آورده شده است. در ادامه فصل شاخص‌های مهم در تحلیل و ارزیابی روش‌ها را بیان کرده و سپس با در نظر گرفتن آن‌ها، روش‌های ترکیب سرویس را مقایسه می‌کنیم.

۱.۳.۲ روش‌های ترکیب سرویس

۱.۱.۳.۲ چارچوب eFlow

eFlow [۳۲] سکویی است که در سال ۲۰۰۲ جهت پشتیبانی از توصیف، تصویب[۱]، نظارت و اجرای سرویس‌های وب مرکب معرفی شد. سرویس‌های مرکب در آن به عنوان فرآیندهای حرفه مدل می‌شوند. ترکیب سرویس‌وب در eFlow توسط یک گراف که ترتیب اجرای بین گره[۲]های فرآیند را تعریف می‌کند، مدل می‌شود. این گراف به صورت دستی ایجاد می‌شود اما به طور خودکار بروز می‌گردد. در حقیقت این روش تنها انقیاد سرویس‌ها به وظیفه‌مندی های درخواستی را انجام می‌دهد و قادر به ترکیب به معنای ایجاد سرویس مرکب برای یک وظیفه‌مندی نیست. همچنین مکانیزمی جهت پشتیبانی از انواع سرویس‌های وب در آن وجود ندارد. این روش قابلیت مدل‌سازی ویژگی‌های کیفی سرویس[۳] را ندارد. با توجه به مراحل دستی زیادی که این روش در طول روال ترکیب سرویس دارد اما هم‌اکنون به عنوان روشی در صنعت مورد استفاده قرار گرفته است.

---

[۱] Enactment
[۲] Node
[۳] QoS: Quality of Service

۲.۱.۳.۲ دیدگاه SELF-SERV

در دیدگاه SELF-SERV [۳۳] که در سال ۲۰۰۲ معرفی شد، سرویس‌های وب به صورت اعلانی ساخته شده و در یک محیط همتا به همتا[۱] اجرا می‌شوند. SELF-SERV سه نوع سرویس معرفی می‌کند: سرویس ابتدایی، سرویس مرکب و سرویس انجمنی[۲].

سرویس انجمنی می‌تواند به عنوان محفظه‌ای برای سرویس‌های پیشنهادی دیده شود. ترکیب سرویس بر اساس نمودار حالت می‌باشد که پارامترهای ورودی و خروجی عملیات و رخدادهای تولیدشده را به یکدیگر متصل می‌کند. این روش ابزاری را برای مشخص کردن سرویس‌های مرکب، قوانین تبدیل داده‌ها و سیاست‌های انتخاب تأمین کننده فراهم می‌کند. این دیدگاه می‌تواند به عنوان یک جعبه‌ابزار برای ترکیب دستی سرویس‌های وب تا یک دیدگاه ترکیب سرویس وب خودکار، طبقه‌بندی شود.

۳.۱.۳.۲ دیدگاه OntoMat-Service

این دیدگاه [۳۴] در سال ۲۰۰۳ توسط Zeng مطرح شد که در آن چارچوبی تحت عنوان OntoMat-Service توصیف شده بود. دیدگاه OntoMat-Service با یک توصیف سرویس WSDL توسط تأمین‌کننده سرویس، آغاز می‌شود. توصیف WSDL تجزیه و تحلیل شده و یک مستند در قالب HTML که قابل خواندن برای انسان باشد، ایجاد می‌گردد. این صفحه ممکن است شامل چندین توصیف سرویس و اتصال به سرویس‌های مربوطه باشد. در مرحله‌ی بعد یا کاربر عبارت-متنی[۳] که مربوط به معانی پایه‌ای قابل فهم برای ماشین است را انتخاب می‌کند و آن‌ها را به مجموعه اصطلاحات خود نگاشت می‌کند و یا به طور مستقیم سرویس وب را فراخوانی می‌کند. نتیجه مرحله نگاشت، مجموعه‌ای از قوانین نگاشت بین هستان‌شناسی[۴] سرویس وب و هستان‌شناسی موجود است.

۴.۱.۳.۲ دیدگاه Orriens ، Yang و Papazoglou [۱]

Orriens و همکارانش در سال ۲۰۰۳ دیدگاهی مدل‌رانه برای پشتیبانی از ترکیب پویای سرویس‌ها ارائه کردند. دیدگاه پیشنهادی از UML به عنوان روشی برای مدل‌سازی ترکیب سرویس به همراه OCL[۵] برای بیان قوانین حرفه استفاده می‌کند. در این روش

---

[۱] Peer to Peer
[۲] Communities
[۳] Text-Phrase
[۴] Ontology
[۵] Object Constraint Language

از قوانین حرفه برای تعیین اینکه یک ترکیب سرویس چگونه باید ساختار دهی و برنامه‌ریزی شود، چگونه تأمین‌کنندگان باید انتخاب شوند و چگونه انقیاد سرویس باید صورت گیرد، استفاده می‌کند. روش پیشنهادی از یک مدل اطلاعاتی[۱] برای ترکیب سرویس استفاده می‌کند.

۵.۱.۳.۲ دیدگاه WebTransact

این دیدگاه [۳۵] در سال ۲۰۰۳ به منظور تأمین ساختاری برای ساخت ترکیب سرویس مورد اعتماد ارائه شد. WebTransact از یک چارچوب چند-لایه که شامل لایه ترکیب سرویس، لایه اجتماع سرویس، لایه مجتمع سازی سرویس و لایه توصیف است، تشکیل شده است. WebTransact از WSDL برای توصیف عملکرد سرویس استفاده کرده و یک زبان تراکنش سرویس‌وب با نام WSTL[۲] بر روی WSDL به منظور تسهیل عملکردهای ترکیب سرویس‌های وب، اضافه کرده است.

۶.۱.۳.۲ چارچوب METEOR-S[۳]

پروژه METEOR [۳۶] در آزمایشگاه LSDIS[۴] دانشگاه Georgia در سال ۲۰۰۴ به عنوان چارچوب پویایی برای ترکیب سرویس‌های وب مطرح شد.ویژگی این پروژه استفاده از مفاهیم معنایی برای تکمیل چرخه حیات سرویس وب معنایی است. چارچوب مطرح شده، مفاهیم معنایی را به استانداردهای کنونی نظیر WSDL اضافه کرد. در این دیدگاه پیدا کردن سرویس‌های مناسب ترکیب، با استفاده از موتور جستجو بر روی مخزن بهبودیافته UDDI تحقق یافته است. این دیدگاه یک ابزار ترکیب سرویس وب مبتنی بر محدودیت که به طراح فرآیند امکان مقید شدن سرویس‌های وب به یک فرآیند انتزاعی بر اساس محدودیت‌های حرفه و فرآیند را می‌دهد، فراهم می‌کند. این روش بر روی ایجاد ترکیب سرویس در زمان طراحی با توسعه الگوهایی که قابلیت انقیاد پویا در زمان اجرا را دارند، تمرکز دارد.

۷.۱.۳.۲ چارچوب WSMF[۵]

چارچوب مدل‌سازی سرویس‌های وبWSMF [۳۷]، که در سال ۲۰۰۵ ارائه شد، چارچوبی برای توصیف جنبه‌های مختلف مربوط به ترکیب سرویس‌وب است. این چارچوب، شامل یک محیط اجرای سرویس‌وب WSMX[۶]، که مرجعی برای پیاده‌سازی

---

[۱] Information Model
[۲] Web Services Transaction Language
[۳] Managing End-To-End OpeRations for Semantic web services
[۴] Large Scale Distributed Information Systems
[۵] Web Service Modeling Framework
[۶] Web Services Execution Environment

هستان‌شناسی مدل‌سازی سرویس‌وب WSMO[۱]، بوده و از زبان مدل‌سازی سرویس‌وب WSML[۲]، استفاده می‌کند، تشکیل شده است. مدل مفهومی این چارچوب، شامل چهار عنصر اصلی است که برای نمایش سرویس‌وب معنایی و موارد مرتبط با آن برای مثال هستان‌شناسی ضروری است. این مدل اصطلاحات رایج استفاده‌شده توسط عناصر WSMO، سرویس‌های درخواستی و سرویس‌های تأمین‌شده و توافقات تأمین‌کننده و درخواست‌کننده، اهدافی که برای توصیف کردن درخواست‌های کاربر بر حسب نیازمندی‌های عملکردی و غیر عملکردی مورد استفاده قرار می‌گیرد و واسطهایی که به مسائل تعامل بین عناصر مختلف WSMO رسیدگی می‌کنند، را تأمین می‌کند. علاوه بر این عناصر اصلی، WSMO مجموعه‌ای از ویژگی‌های غیر عملکردی که به طور کلی تعریف‌شده و ممکن است توسط هر یک از عناصر مدل‌سازی استفاده شود، معرفی می‌کند.

۸.۱.۳.۲ دیدگاه SeGSeC[۳]

در این دیدگاه[۳۸] که در سال ۲۰۰۶ معرفی شد، یک معماری مبتنی بر معنا[۴] برای ترکیب پویا ارائه شده است. کاربر سرویس را با زبان طبیعی درخواست می‌کند و سپس درخواست به یک قالب قابل فهم برای ماشین به طور مثال به صورت گراف معنایی تبدیل شده و بر اساس این گراف، SeGSeC ترکیب سرویس را انجام می‌دهد. در این دیدگاه به منظور دست‌یابی به ترکیب پویای مبتنی بر معنا، مدل‌سازی مؤلفه‌های ترکیب و نیز مکانیزم ترکیب لازم است از معناشناسی پشتیبانی کنند.

۹.۱.۳.۲ چارچوب MoSCoE[۵]

MoSCoE [۳۹] پروژه‌ای است که توسط دانشگاه Iowa در سال ۲۰۰۶ مطرح شد. اهداف این پروژه ایجاد یک چارچوب برای مدل‌سازی ترکیب سرویس و اجرای آن است. فرآیند ترکیب MoSCoE به سه مرحله تقسیم می‌شود: ۱. انتزاع ۲. ترکیب ۳. پالایش. انتزاع تأمین‌شده در این چارچوب، به کاربران اجازه می‌دهد که با استفاده از یک توصیف سطح بالا، یک سرویس را درخواست دهند. تأمین‌کنندگان سرویس، سرویس‌های خود را با استفاده از زبان‌های رایج توصیف استاندارد سرویس‌ها نظیر OWL-S و WSDL اعلان می‌کنند. با دادن یک درخواست سرویس، موتور ترکیب چارچوب، در صورت امکان، یک ترکیب مناسب از سرویس‌های موجود ایجاد می‌کند، در غیر این صورت، فاز پالایش را آغاز می‌کند و کاربر را در طی فرایند پالایش، درخواست سرویس هدایت می‌کند تا یک ترکیب سرویس را ایجاد نماید. فرایند پالایش تکرارشونده است و زمانی پایان می‌یابد که

---

[۱] Web Services Modelling Ontology
[۲] Web Services Modelling Language
[۳] Semantic Graph-Based Service Composition
[۴] Semantic-based
[۵] Modeling Web Service Composition and Execution

ترکیب سرویس مناسب یافت شود و یا کاربر تصمیم به پایان دادن فرآینـد ترکیـب بگیـرد. در صـورت حاصـل شـدن یـک ترکیـب سرویس، به یک جریان بهم پیوسته BPEL تبدیل می‌شود تا قابلیت اجرا پیدا کند. ترکیب سـرویس قـوانینی را بـرای ویژگـی‌هـای غیر عملکردی تعریف می‌کند. در زمان اجرا، این قوانین مورد نظارت واقع می‌شـوند و در هنگـام رخ دادن برخـی رویـدادها، اقـدامات مناسب اتخاذ می‌گردد.

۱۰.۱.۳.۲ دیدگاه SODIUM[۱]

SODIUM [۴۰] پروژه‌ای بین‌المللی است که در سال ۲۰۰۷ مطرح شد. این پروژه مدل سرویس جامعی[۲] که شامل مفـاهیم رایـج سرویس‌های ناهمگن از نقطه‌نظرهای متفاوت است را توسعه داد. متدلوژی SODIUM یک دیدگاه مدل‌رانه و تکاملی بـرای ترکیـب سرویس در چهار فاز زیر ارائه داد :

- فاز اول : کاربر با تعریف جزئیات وظیفه‌های پیچیده در سطح انتزاعی بالا کار خود را آغاز می‌کند.
- فاز دوم : کاربر از این توصیف انتزاعی برای تولید پرسش مورد استفاده برای کشف سرویس استفاده می‌کند.
- فاز سوم : سرویس‌های کشف‌شده به منظور ساکن کردن هر وظیفه‌های فرآیند انتزاعی استفاده مـی‌شـوند، بـدین منظـور به توصیف فرآیند واقعی[۳] تغییر شکل مـی‌یابـد و هـر دو توصـیف انتزاعـی و واقعـی هـر دو در ترکیـب سـرویس ذخیـره می‌شوند.
- فاز چهارم : توصیف واقعی فرآیند، به توصیف قابل‌اجرا و مستند قابل انتشار در مـورد سـرویس مرکبـی کـه سـاخته‌شـده است، تبدیل می‌شود.

۱۱.۱.۳.۲ دیدگاه Chan and Michael[۳۱]

در دیدگاهی که Chan و همکارش در سال ۲۰۰۸ برای ترکیب پویـای سـرویس‌هـای وب ارائـه کردنـد، هـر سـرویس بـا WSDL توصیف و تعامل مابین سرویس با سایر سرویس‌ها با استفاده از WSCI[۴] بیان می‌شـود. (WSCI زبـان توصـیفی مبتنـی بـر XML است که جریان پیام‌های مبادله شده بین سرویس‌های شرکت‌کننده در ترکیب سرویس را توصیف می‌کند.)

---

[۱] Service-Oriented Development In a Unified fraMework
[۲] Generic
[۳] Concrete
[۴] Web Service Choreography Interface

الگوریتم ارائه شده در این دیدگاه بر اساس اطلاعات این دو توصیف نوشته شده است و خروجی به صورت فایل BPEL نمایش داده می‌شود. این دیدگاه درخت ترکیب سرویس را به صورت پایین به بالا با استفاده از ورودی-خروجی‌ها و نیازمندی‌های مورد نظر کاربر می‌سازد. در این دیدگاه ایده‌ای برای ترکیب انواع مختلف سرویس‌های وب پیشنهاد نشده است و روش ترکیب تنها برای سرویس‌های مبتنی بر SOAP ارائه شده است.

۱۲.۱.۳.۲ چارچوب DynamiCoS[۱]

چارچوب DynamicCoS[۴۱] در سال ۲۰۰۹، به منظور خودکار سازی و ترکیب پویای سرویس‌ها ارائه شد. هدف اولیه آن آسان کردن پیچیدگی‌های ترکیب سرویس برای کاربران نهایی بود. در این دیدگاه پس از توصیف خصوصیات سرویس توسط کاربر بر اساس نیازمندی‌ها، کشف، تطبیق و ترکیب خودکار به صورت خودکار انجام می‌گیرد. نتیجه نهایی برای انتخاب مناسب‌ترین ترکیب، به کاربر نمایش داده می‌شود. DynamicCoS ایجاد و انتشار سرویس‌ها توسط توسعه دهندگان را در زمان طراحی و ترکیب خودکار سرویس توسط کاربر نهایی را در زمان اجرا قادر می‌سازد. در این دیدگاه الگوریتم‌های ترکیب بر اساس گراف معنایی هستند. این دیدگاه هنوز به عنوان یک روش آکادمیک محسوب می‌گردد.

۱۳.۱.۳.۲ چارچوب REST2SOAP [۴۲]

در این چارچوب که به عنوان یکی از اولین چارچوب‌ها برای ترکیب همزمان سرویس‌های RESTful و مبتنی بر SOAP در سال ۲۰۰۹ ارائه شده است، از توصیفات WADL[۲] کمک گرفته شده است و به صورت نیمه خودکار سرویس‌های RESTful به سرویس مبتنی بر SOAP تبدیل شده و در نهایت ترکیب سرویس توسط BPEL انجام می‌شود. تبدیل سرویس‌های RESTful به سرویس‌های مبتنی بر SOAP و سپس انجام ترکیب آن‌ها، منجر به از دست رفتن خواص ذاتی سرویس‌های RESTful می‌گردد.

۱۴.۱.۳.۲ دیدگاه Zhenhua, Chenting و Zhang [۳۰]

Zhang و همکارانش در سال ۲۰۰۹ دیدگاهی مدل‌رانه برای ترکیب پویای سرویس‌های وب ارائه کردند. ترکیب پویا در این دیدگاه با تعریف دو مدل UML برای مدل‌سازی سرویس‌وب و فرآیند حرفه به همراه توصیفت معنایی وظیفه‌مندی سرویس‌وب و

---

[۱] Dynamic Composition Of Services
[۲] Web Application Description Language

نیازمندی‌های فرآیند حرفه انجام شده است. در ادامه پیاده‌سازی سرویس‌وب و فرآیند حرفه با دنبال کردن الگوهای نگاشت[۱] و قوانین تبدیل از پیش تعیین شده، به صورت خودکار از مدل‌های معرفی شده، تولید می‌شود. در این روش با اعمال تطبیق معنایی بر مبنای هستان‌شناسی، سرویس وب واجد شرایط به صورت پویا انتخاب و به فرآیند حرفه می‌پیوندد. این روش از سه مرحله مدل‌سازی، انقیاد و ساخت تشکیل شده است. خروجی این روش فرمت قابل اجرای فرآیند حرفه مورد نظر با زبان BPEL است. در این روش انواع سرویس‌های وب در نظر گرفته نشده است و همچنین قابلیت ترکیب سرویس‌ها در صورتی که سرویس‌های موجود توانایی پاسخ‌گویی به وظیفه‌مندی مورد نیاز را نداشته باشند، ندارد، در واقع این روش تنها انتخاب و انقیاد سرویس‌ها را به صورت پویا انجام می‌دهد.

۱۵.۱.۳.۲ دیدگاه ترکیب خودکار سرویس‌های RESTful [۱۷]

در سال ۲۰۰۹، Zhao و همکارش دیدگاهی را برای ترکیب خودکار سرویس‌های وب RESTful ارائه دادند. در این روش ابتدا با کمک هستان‌شناسی مدلی مفهومی برای توصیف سه نوع موجود سرویس‌های RESTful ارائه داده شد. هدف از ارائه این مدل، خودکار سازی روند ترکیب سرویس‌های RESTful بیان شده است. در این روش با استفاده از situation calculus سرویس‌های RESTful بر اساس مدل مفهومی معرفی شده به طور خودکار ترکیب می‌شوند. در این دیدگاه امکان ترکیب سرویس‌های RESTful و مبتنی بر SOAP به عنوان کار آینده معرفی شده است.

۱۶.۱.۳.۲ دیدگاه Kaouthar و Zahi [۴۳]

Kaouthar و همکارش در سال ۲۰۱۰ دیدگاه پویایی برای ترکیب سرویس وب، ارائه کردند. هدف از این دیدگاه معرفی معماری انعطاف‌پذیر برای ترکیب پویای سرویس وب بر اساس نیازمندی کاربر و سرویس‌های در دسترس است. این معماری توانایی پیکربندی مجدد[۲] در زمان اجرا، هنگامی رخ دادن خطا را دارد.

این دیدگاه از یک مؤلفه Query Analyser برای تحلیل درخواست کار سپار، یکDiscovery Component به منظور کشف سرویس‌های منتشر شده در UDDI، یک Constraints Manager Component به منظور نظارت بر در دسترس بودن منابع در زمان اجرا، یک Selection Manager برای تولید برنامه اجرا از بین سرویس‌های کشف شده از UDDI و Orchestrator برای رسیدگی به اجرای BPEL ایجاد شده به منظور تولید سرویس مرکب نهایی تشکیل شده است.

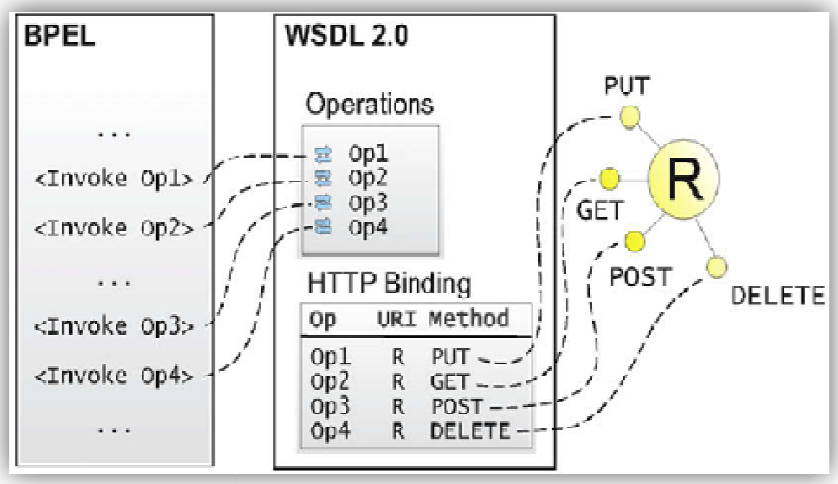
---

[۱] Mapping Pattern
[۲] Reconfigure

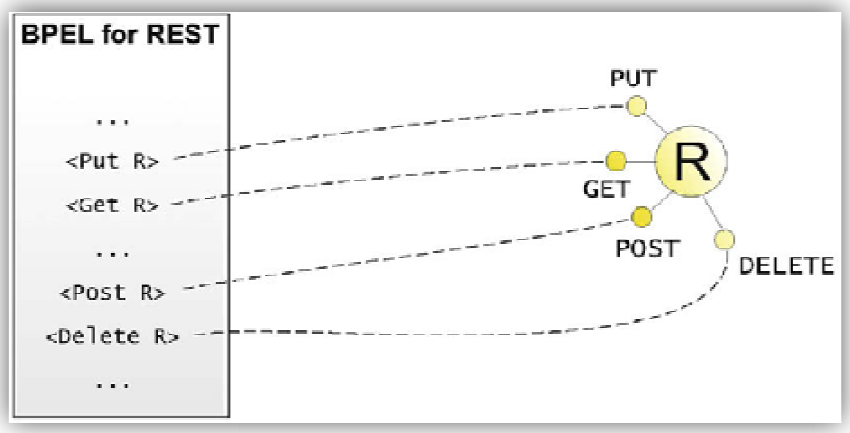
۱۷.۱.۳.۲ دیدگاه BPEL for REST [۴۴] و [۲۲]

همان‌طور کـه در بخـش ۲.۲.۲.۲ اشـاره شـد، آخـرین نسـخه BPEL بـا WSDL نسـخه ۱.۱سـازگار اسـت و قابلیـت فراخـوانی سرویس‌های وب RESTful را ندارد. راه‌های مختلفی برای ترکیب ناهمگن سرویس‌هـای وب در چنـد سـال اخیـر پیشـنهاد شـده است، برخی از راه‌ها با تبدیل این دو نوع سرویس به یکدیگر و همگن کردن سرویس‌های موجود، ترکیب را انجـام مـی‌دهنـد، ماننـد روش معرفی شده در ۱۳.۱.۳.۲، که این کار باعث از بین رفتن ویژگی‌های ذاتی هر یک از انواع سرویس‌هـا مـی‌گـردد. دو راه ممکـن دیگر بدین منظور مطرح شده است، یکی از راه‌ها به طوری که در شکل ۲-۹ نشان داده شده است، اسـتفاده از WSDL نسـخه ۲.۰ و پشتیبانی BPEL از این نسخه WSDL است که BPEL نسخه ۲.۰ نام گذاری شده است؛ اما ایـن روش هنـوز تنهـا بـه صـورت آکادمیک مطرح است. همچنین WSDL نسخه ۲.۰ نیز برای توصیف سرویس‌های RESTful رواج نیافته و در صورت اسـتفاده بـا توجه به ماهیت پویای سرویس‌های وب RESTful با هر تغییری در سرویس، نیاز است که بروز رسانی گـردد. از دیـد تئـوری نیـز، این روش اصول تعاملی منبع محور سرویس‌های RESTful را درون انتـزاع سـرویس محـور فـراهم شـده توسـط WSDL نادیـده می‌گیرد. به علاوه فراخوانی یک عملیات WSDL توسط تبادل پیام‌ها به صورت همگام یا نا همگـام همـواره بـا مفـاهیم درخواسـت ''POST'', ''PUT'', ''GET'' و ''DELETE'' بر روی URL یک منبع، تطابق کامل ندارد، بنابراین پشتیبانی ذاتی بـرای کنتـرل صریح تعاملات بر اساس اصول REST برای توسعه دهندگان ترکیب سرویس‌ها مناسب‌تر است [۲۲].

بنابراین راه دیگر برای فراخوانی مستقیم سرویس‌های RESTful از طریق توسعه دادن فایل BPEL به عنوان مناسب‌ترین راه‌حـل مطرح است که در شکل ۲-۱۰ نشان داده شده است. این راه‌حل به عنوان راه‌حل انتخـابی در فـاز پایـانی چـارچوب پیشـنهادی در فصل ۳، مورد استفاده قرار خواهد گرفت.



شکل ۲-۹: نحوه فراخوانی سرویس‌های RESTful از طریق WSDL 2.0 [۴۴]



شکل ۲-۱۰: فراخوانی مستقیم سرویس‌های RESTful از طریق توسعه BPEL for REST [۴۴]

به طور کلی اگر سرویس‌های RESTful را به صورت منبع در نظر بگیریم، چرخه حیات یک منبع با یک درخواست POST یا PUT برای مقدار دهی اولیه دادن[۱] به حالت آن منبع آغاز می‌گردد. زمانی که یک منبع ایجاد شد، یک کار سپار می‌تواند توسط درخواست GET حالت جاری آن را بازیابی کند و یا با درخواست PUT حالت آن را بروز رسانی کند، یا با درخواست DELET نیز آن منبع را حذف کند.

[۱] Initialize

دو هدف اصلی از توسعه BPEL for REST در ادامه معرفی می‌شوند:

۱. توانایی فراخوانی مستقیم یک سرویس وب RESTful درون یک فرآیند BPEL. برای دست‌یابی به فراخوانی مستقیم با استفاده از پروتکل HTTP از درون یک فرآیند BPEL، چهار فعالیت[۱] <put> <post> ,<get> و<delete> به فعالیت‌های BPEL افزوده شدند. همان‌طور که در شکل ۲-۱۱ نمایش داده شده است،

- چهار فعالیت معرفی شده، از مشخصه URL برای مشخص کردن، آدرس منبع هدف استفاده می‌کنند.
- تنها محدودیت ساختاری URL این است که باید یک منبع قابل دسترس از طریق پروتکل‌های HTTP یا HTTPS را نشان دهد.
- مشابه فعالیت <invoke>، داده‌های درخواست و پاسخ درون متغیرهایی که با مشخصه‌های request و response مشخص شده‌اند، ارجاع داده می‌شوند.
- فعالیت‌های <get> و <delete> ،مشخصه request را ندارند و تنها بر روی URL منبع، عملیات را انجام می‌دهند.
- برای فعالیت‌های <put> و <delete> مشخصه response اختیاری است، ممکن است برخی سرویس‌ها یک پاسخ خالی به ازای این دو متد ارسال کنند.
- سرآمد[۲] ارسال شده با درخواست HTTP، می‌تواند با استفاده از عنصر<header> در هر یک از چهار نوع فعالیت کنترل شود.
- مشابه فعالیت <invoke> در BPEL استاندارد، <get> ،<post> ، <put> و<delete> توانایی رسیدگی کردن به عدم موفقیت در فراخوانی را دارند.

---

[۱] Activity in BPEL Tags

[۲] Header

```
<get uri="" response="" response_headers=""?>
  <header name="">*value</header>
  <catch code="">*...</catch>
  <catchAll>?...</catchAll>
</get>

<post uri="" request="" response="" response_headers=""?>
...
</post>

<put uri="" request="" response=""? response_headers=""?>
...
</put>

<delete uri="" response=""? response_headers=""?>
...
</delete>
```

شکل ۲-۱۱: توسعه BPEL for REST برای فراخوانی مستقیم سرویس RESTful [۴۴]

۲. نمایش دادن بخشی از فرآیند اجرایی BPEL به عنوان یک منبع با استفاده از ساختاری اعلانی. بدین معنی که بخشی از فرآیند قابلیت انتشار به عنوان یک سرویس وب RESTful را داشته باشد. بدین منظور برای مجموعه‌ای از فعالیت‌ها، ساختارهای مربوط به اصول واسط یکنواخت REST معرفی شده‌اند. همان‌طور که در شکل ۲-۱۲ مشاهده می‌شود،

- عنصر <resource> اضافه شده است. زمانی که فرآیند به این عنصر می‌رسد، URL متناظر با این منبع منتشر شده و کارسپارها می‌توانند برای استفاده آن درخواست دهند.
- URL منابع تودرتو[۱] با به هم پیوستن ویژگی URLشان با جداکننده (/) بدست می‌آیند.
- مشابه <scope> در BPEL، یک <resource> ممکن است شامل مجموعه‌ای از اعلان <variable> برای ساخت حالت منبع یک URL داده شده باشد. این متغیرهای حالت تنها در زمان اعلان منابع قابل مشاهده هستند.

[۱] Nested

- مشابه <pick> در BPEL، <resource> شامل مجموعه‌ای از رسیدگی‌کننده[۱]ها ، برای زمانی است که فرآیند یک درخواست HTTP را دریافت می‌کند. رسیدگی‌کننده درخواست موجود در <resource> شامل ,<onGet> <onPost>, <onPut>, <onDelete> است.
- در صورتی که رسیدگی‌کننده‌ی درخواست برای متد درخواستی اعلان نشده باشد، درخواست به آن منبع توسط BPEL engine با کد HTTP ۴۰۵ به معنی (مجاز نبودن متد) پاسخ داده می‌شود.
- برای اعلان هر منبع حداقل وجود یک رسیدگی‌کننده درخواست ضروری است.
- برای دسترسی به داده‌های ارسال شده توسط کارسپارها با یک درخواست، یک متغیر از پیش تعریف شده با نام $request در محدوده رسیدگی‌کننده درخواست در دسترس است.
- نتایج با استفاده از فعالیت <response> افزوده شده، برگردانده می‌شوند.
- مشخصه Code برای کنترل کد وضعیت موجود در پاسخ HTTP که به کار سپار برای نشان دادن موفق یا ناموفق بودن رسیدگی به درخواست، استفاده می‌شود.

```
<resource uri="">
 <variable>*
 <onGet>?                          ... </onGet>
 <onPut>?                          ... </onPut>
 <onDelete>?                       ... </onDelete>
 <onPost isolated="false"?>?  ... </onPost>
</resource>

<respond code=""?>
 <header name="">*value</header>
  payload
</respond>
```

شکل ۲-۱۲: توسعه BPEL for REST برای اعلان کردن یک سرویس RESTful در بخشی از فرآیند BPEL [۴۴]

[۱] Handler

علاوه بر موارد اضافه شده به BPEL برای توسعه‌ای که امکان پشتیبانی از مورد ۱ و ۲ توصیف شده در بالا را داشــته باشــد،REST BPEL for تغییرات اندک زیر را در BPEL استاندارد در نظر گرفته است.

- فعالیت <exit/> در BPEL استاندارد، تنها بر روی بالاترین سطح <resource> تأثیر گذار است، که حتی پس از آنکه اجرای نرمال فرآیند کامل شد در دسترس کارسپارها خواهد بود.
- به دلیل عدم وجود یک توصیف واسط ایستا برای سرویس‌های وب RESTful و کمبود محدودیت‌های نوع‌دهی[۱] قوی بر روی داده‌هایی که باید منتقل شوند، توسعه BPEL for REST یک زبان نوع دهی پویا[۲] است. بنابراین نوع‌دهی ایستا در اعلان <variable> در این زبان اختیاری شده است. به ویژه، مشخصه messageType در آن استفاده نمی‌شود [۲۲].

[۱] Typing

[۲] Dynamically Typed Language

۴.۲ مقایسه روش‌ها

در این تحقیق سعی شده تا حد امکان تمامی روش‌های موجود برای ترکیب سرویس‌ها، مورد بررسی و ارزیابی قرار گیرند. از آنجا که این بخش از تحقیق، در سمیناری با موضوع "بررسی روش‌های ترکیب سرویس‌های وب"[۳] انجام شده است، لذا تنها خلاصه‌ای از روش‌ها در این تحقیق بیان شد و در جدول ۲-۱ نتیجه ارزیابی روش‌هایی که از نظر دسته‌بندی به صورت چارچوبی برای ترکیب سرویس ارائه شده‌اند، بر اساس برخی شاخص‌های با اهمیت‌تر نشان داده شده است. مقادیر معتبر در ستون‌های جدول ۲-۱ به صورت زیر است:

۱. راهبرد: در این ستون راهبرد ترکیب مورد استفاده در چارچوب مورد نظر بیان می‌شود. راهبردهای ممکن می‌تواند هر یک از موارد زیر باشد:
   - معنایی: استفاده از روش‌های معنایی و توصیفات معنایی در روال توصیف و ترکیب سرویس مرکب.
   - مدل‌رانه: استفاده از ایده‌های مدل‌رانه در روال ترکیب سرویس.
   - پویا یا ایستا: پشتیبانی از قابلیت سازگاری پویا با تغییرات در فرآیند سرویس مرکب.
۲. پشتیبانی از ویژگی‌های غیر عملکردی (QoS): در صورتی که روش حاوی دستورالعمل و یا الگوریتم‌هایی جهت پشتیبانی از نیازمندی‌های غیر عملکردی سرویس‌ها باشد با علامت (*) مشخص شده است.
۳. سطح خودکار سازی: می‌تواند شامل روش‌های دستی، روش‌های نیمه خودکار و یا روش‌های تمام خودکار باشد.
۴. زمان ترکیب: می‌تواند ترکیب در زمان طراحی انجام شود و یا در زمان اجرا.
۵. قابلیت استفاده در سازمان: در بر داشتن ویژگی‌هایی برای کاربردی کردن چارچوب و قابلیت استفاده در سطح سازمان.
۶. پشتیبانی از ترکیب سرویس‌های ناهمگن: در صورتی که روش حاوی راهکارهایی جهت پشتیبانی از ترکیب انواع سرویس‌های وب باشد با علامت (*) مشخص شده است.

توجه: علامت (-) در جدول ۲-۱ به این معناست که در مورد موضوع مورد نظر طی مراحل تحقیق این پایان نامه، اطلاعات کافی بدست نیامده است و همچنین عدم داشتن علامت در ویژگی‌هایی که با علامت مشخص شده‌اند نظیر قابلیت استفاده در سازمان، به معنی عدم پشتیبانی از این ویژگی برای آن روش است.

جدول ۲-۱: مقایسه چارچوب‌ها و دیدگاه‌های ترکیب سرویس‌ها

ردیف	چارچوب	راهبرد ترکیب	پشتیبانی از QoS	سطح خود کارسازی	زمان ترکیب	قابلیت استفاده در سازمان	پشتیبانی از ترکیب سرویس ناهمگن
۱.	eFlow [۴۵]	پویا		نیمه خودکار	طراحی	❋	
۲.	METEOR-S [۳۶]	معنایی		نیمه خودکار – دستی	طراحی و اجرا		
۳.	WebTransact [۳۵]	پویا		نیمه خودکار–دستی	طراحی		
۴.	DynamiCoS [۴۱]	معنایی	❋	خودکار	اجرا		
۵.	SELF-SERV [۳۳]	ایستا		دستی	–		
۶.	OntoMat-Service [۳۴]	معنایی		نیمه خودکار–دستی	–		
۷.	SODIUM [۴۰]	مدل‌رانه – پویا	❋	دستی	طراحی		
۸.	MoSCoE [۳۹]	معنایی–ایستا	❋	نیمه خودکار	طراحی		
۹.	WSMF [۳۷]	معنایی	❋	خودکار	اجرا		
۱۰.	دیدگاه Orriens [۱]	مدل‌رانه	❋	نیمه خودکار	طراحی	❋	
۱۱.	REST2SOAP [۴۲]	–		–	اجرا		❋

در این بخش ۱۷ روش، دیدگاه و چارچوب برای ترکیب سرویس معرفی شده که از بین آن‌ها یازده چارچوب ترکیب سرویس که اطلاعات لازم جهت مقایسه را داشتند در جدول ۲-۱، مورد مقایسه و ارزیابی قرار گرفت. باید توجه داشت که روش‌های معرفی شده در این بخش، روش‌هایی هستند که طی گام تحقیق این پایان نامه مورد بررسی قرار گرفته‌اند و ممکن است روش‌های دیگری نیز برای ترکیب سرویس‌ها ارائه شده باشد که به آن‌ها اشاره نشده است. به عنوان مثال روش‌های دیگری نیز برای ترکیب سرویس ارائه شده‌اند نظیر[۴۳]، [۴۶]، [۴۷] و .... که به علت عدم امکان دست‌یابی به اطلاعات کافی در این جدول به آن‌ها اشاره نشده است. در ادامه بر اساس شاخص‌های معرفی شده به تحلیل و ارزیابی روش‌های فوق می‌پردازیم.

۱.۴.۲ تحلیل روش‌های بررسی‌شده

اگر فرآیند ترکیب سرویس وب را به صورت مدلی پنج مرحله‌ای شامل توصیف سرویس، تبدیل سرویس، تولید فرآیند ترکیب، ارزیابی و اجرا در نظر بگیریم، هر مرحله به زبان‌ها، سکوها و روش‌های مختلفی نیاز دارد؛ روش‌های مرحله تولید فرآیند ترکیب، شامل روش‌های جریان کار و روش‌های هوش مصنوعی است، روش‌های جریان کار در شرایطی که مدل فرآیند با درخواست تعریف شده باشد اما نیازمند برنامه‌ای خودکار برای یافتن سرویس‌های اتمیک برای برآوردن نیازمندی‌هاست. روش‌های هوش مصنوعی در زمانی که درخواست دهنده مدل فرآیندی ندارد اما یک مجموعه از محدودیت‌ها و اولویت‌ها دارد. در این صورت مدل فرآیندی می‌تواند به طور خودکار توسط برنامه تولید شود. اگرچه روش‌های مختلف سطح متفاوتی از خود کارسازی در ترکیب سرویس را تأمین می‌کنند، نمی‌توان گفت که خود کارسازی بیشتر بهتراست، زیرا محیط سرویس وب محیطی پیچیده است و تولید هر چیزی به صورت تمام خودکار دست‌یافتنی نیست و همچنین استفاده از بازخورد[۱]های کاربر برای ارائه سرویسی که به نیازمندی مورد نظر کاربر نزدیک‌تر باشد، بسیار حائز اهمیت است.

در مقایسه با روش‌های سنتی ترکیب سرویس، امروزه روش‌های ترکیب‌سرویس‌وب شانس موفقیت بیشتری به دلیل مطرح شدن استانداردها و تلاش برای استانداردسازی دارند. با WSDL عملکرد سرویس‌های وب به طریقی سازگار توصیف می‌شود، البته QoS و توصیفات معنایی برای توسعه استاندارد WSDL کنونی پیشنهاد شده‌اند. UDDI و سایر مخازن بر اساس مدل داده‌ها پیاده‌سازی شدند اما هنوز به صورت گسترده استفاده نمی‌شوند، به نیازمندی‌های کشف پویای سرویس، هنوز به طور کامل پاسخ داده نشده است. از آنجا که نرم‌افزارهای جدید امروزه توسط سرویس‌های وب ساخته می‌شوند، لذا ترکیب سرویس به موضوعی تعیین‌کننده تبدیل شده است که تأثیر بسزایی بر حوزه‌های مختلف تحقیقات خواهد داشت.

امروزه بیشتر تلاش‌ها در توسعه چارچوب برای ترکیب سرویس صورت می‌گیرد در حالیکه حالت کنونی تحقیقاتی کمبودی را در زمینه توصیف دغدغه‌های QoS با گسترش WSDL احساس می‌کند. نتایج تحقیقات نشان می‌دهد که محدودیت‌های موجود در فناوری‌های کنونی ترکیب سرویس به این دلیل است که این روش‌ها کاملاً مبتنی بر استانداردهایی نظیر SOAP، WSDL و UDDI هستند. از بررسی روش‌های معرفی شده در بخش قبل به این نتیجه می‌رسیم که در نظر گرفتن راه‌حل‌هایی برای کنترل

---

[۱] Feedbacks

پیچیدگی روش و چارچوب ترکیب سرویس در اکثر روش‌ها مورد توجه قرار نگرفته لذا چارچوب ارائه شده به این دلیل قابلیت استفاده در سازمان و تبدیل به روشی صنعتی را ندارد.

در نظر گرفتن انواع سرویس‌های وب، تنها در یک چارچوب مد نظر قرار داده شده است. تبدیل شدن به روشی تمام خودکار برای ترکیب سرویس‌های وب، مستلزم استفاده از روش‌های معنایی و هستان شناسی بوده و از طرف دیگر استفاده از روش‌های معنایی در چارچوب ترکیب سرویس در یک سازمان نیازمند عمومی کردن هستان شناسی سازمان بوده که ممکن است سازمان در این زمینه توافق نداشته باشد. پشتیبانی از نیازمندی‌های غیر عملکردی در فاز کشف سرویس‌ها و در نهایت انتخاب سرویس مرکب نهایی از بین سرویس‌هایی کاندیدا تأثیر گذار خواهد بود که در برخی چارچوب‌های ترکیب سرویس، در نظر گرفته نشده است.

در صورتی که یک چارچوب هم‌زمان با توسعه سرویس مرکب تمهیدات لازم برای اجرای سرویس مرکب را نیز فراهم آورد می‌تواند از فراخوانی زمان اجرا برای سرویس‌های موجود در سرویس مرکب استفاده کند. روش‌هایی که از ترکیب زمان طراحی استفاده می‌کنند نیز می‌توانند این انقیاد را به آخرین مرحله توسعه سرویس مرکب منتقل کنند تا پویای روش توسعه ترکیب افزایشی‌اید.

۵.۲ جمع‌بندی مطالب فصل

در این فصل ابتدا به تعریف مفاهیم مورد استفاده در این تحقیق پرداختیم و به طور کلی به مفاهیم مرتبط با موضوعات، معماری سرویس‌دهنده، ترکیب سرویس، معماری سبک REST، معماری مدل‌رانه (به عنوان روش مورد استفاده در چارچوب پیشنهادی) اشاره کردیم. سپس در ادامه فصل به بررسی برخی روش‌های موجود برای ترکیب سرویس‌های وب پرداختیم.

در ادامه‌ی فصل با معرفی شاخص‌هایی برای تحلیل و مقایسه دیدگاه‌های ترکیب سرویس، دیدگاه‌های معرفی شده بر اساس این شاخص‌ها مورد مقایسه و ارزیابی قرار گرفت. این شاخص‌ها عبارتند از : راهبرد ترکیب، پشتیبانی از ویژگی‌های کیفی، سطح خود کارسازی، زمان ترکیب، قابلیت استفاده در سازمان، پشتیبانی از ترکیب ناهمگن.

نکته‌ای که تنها در یکی از روش‌ها و دیدگاه‌های معرفی شده ترکیب‌سرویس‌وب به آن اشاره‌ای شد، ترکیب انواع سرویس‌های وب اعم از RESTful و مبتنی بر SOAP است. اکثر روش‌های معرفی شده بر روی ترکیب سرویس‌های وب مبتنی بر SOAP تمرکز داشته و به دلیل جدید بودن حوزه سرویس‌های RESTful، روش‌های ترکیب سرویس‌های وب RESTful از توجه زیادی برخوردار نبوده‌اند. همچنین قابلیت استفاده در سطح سازمان و امکان صنعتی شدن روش تنها در چارچوب e-flow امکان‌پذیر است که این چارچوب تنها برای سرویس‌های مبتنی بر SOAP بوده و بخش زیادی از فازهای آن لازم است به صورت دستی انجام شود.

۳ فصل سوم – مبانی روش پیشنهادی

۱.۳ مقدمه

در فصل‌های گذشته با مفاهیم سرویس و مراحل کلی ترکیب سرویس آشنا شدیم، همچنین روش‌های کنونی ترکیب سرویس معرفی و برخی شاخص‌ها برای مقایسه این روش‌ها معرفی شدند.

مشخص شد که تحقیقات بسیاری بر روی ترکیب سرویس‌های وب متمرکز شده‌اند. روش‌های معرفی شده تاکنون را می‌توان به دو دسته‌ی کلی تقسیم نمود: یکی تلاش‌های صنعتی که اساساً به راه‌حل‌های مرتبط با WSDL و BPEL منجر می‌شوند و دیگری روش‌های معنایی و آکادمیک که با OWL-S و هوش مصنوعی ترکیب سرویس را انجام می‌دهند. علاوه بر این، این نکته حائز اهمیت است که استفاده از معماری مدل‌رانه کارایی فرآیند ترکیب سرویس‌ها در راه‌حل‌های صنعتی را بهبود بخشیده است[۳۰].

با توجه به چالش‌ها و کمبودهای روش‌های موجود هدف اصلی از این تحقیق استخراج و بر طبق آن روش پیشنهادی ارائه گردید، برخی از این چالش‌ها عبارتند از:

- تعداد سرویس‌های وب روز به روز در حال افزایش است، از این رو روند ترکیب سرویس‌های وب به یک فرآیند پیچیده تبدیل شده است، اما اغلب روش‌های پیشنهادی راه‌حل مشخصی برای غلبه بر این پیچیدگی ارائه نکرده‌اند.
- عدم توجه کافی روش‌های ارائه شده به تغییرات مکرر در محیط سرویس‌های وب و قابلیت سازگاری و تطبیق با تغییر در ویژگی سرویس‌ها، به بیانی دیگر پشتیبانی از ترکیب پویا.
- تلاش‌های محدودی بر روی ارائه روش ترکیبی که تمهیدات لازم جهت پشتیبانی از ترکیب انواع سرویس‌های وب را دارا باشد، انجام شده است.
- عدم توجه کافی به ویژگی‌های کیفی سرویس‌ها در حین روال ترکیب و انتخاب سرویس‌ها.
- لازمه استفاده از روش‌های معنایی و هستان‌شناسی برای راه‌حل‌های سازمانی، انتشار عمومی هستان‌شناسی مورد استفاده برای آسان کردن مجتمع سازی میان تأمین‌کنندگان و مصرف‌کنندگان مختلف است [۴۸]. در صورتی که در راه‌حل‌های سازمانی متاداده‌های سازمان گاهاً جزء دارایی‌های محرمانه سازمان به شمار می‌آیند و ممکن است سازمان تمایلی به انتشار آن‌ها نداشته باشد. لذا استفاده از روش‌های معنایی برای سازمان‌ها با محدودیت‌هایی همراه است.

به منظور پاسخ‌گویی به این چالش‌ها و با هدف ارائه روشی که قابلیت صنعتی شدن و استفاده برای سازمان‌های کشور را دارا باشد، هدف اصلی زیر را برای روش ارائه شده در این تحقیق، برگزیدیم:

"روش ترکیبی جامع که به صورت مدل‌رانـه قـادر بـه توسـعه نیمـه خودکـار ترکیـب انـواع سـرویس‌هـای وب بـا در نظـر گـرفتن نیازمندی‌های غیر عملکردی باشد."

از آنجا که روش به صورت مدل‌رانه قابلیت ترکیب انواع سرویس‌های وب را دارد به این روش ترکیب مدل‌رانـه سـرویس‌هـای وب نـا همگن (Model-driven Composition of Heterogeneous Service) یا به اختصار MDCHeS گوییم. منظـور از نـاهمگنی در سرویس‌ها در سطح استانداردهای مورد استفاده و پروتکل‌هایی است که آن سرویس بر اسـاس آن تعریـف و اسـتفاده مـی‌شـود. همچنین به چارچوب معرفی شده برای این روش در این تحقیق، چارچوب MDCHeS گفته می‌شود.

با توجه به توصیف بالا از هدف این تحقیق در ادامه به قابلیت‌های روش MDCHeS اشاره می‌کنیم:

- قابلیت توصیف سرویس‌های وب RESTful و مبتنی بر SOAP بـه صـورت همزمـان بـا اسـتفاده از مـدل متـاداده ای[۱] پیشنهادی.
- قابلیت تغییر در ویژگی‌های عناصر ترکیب، نظیر تأمین‌کنندگان و سرویس‌های آن‌ها در حین ترکیب، بـه دلیـل اسـتفاده از مفهومی به نام فعالیت[۲] در مدل متاداده ای روش پیشنهادی، به بیان دیگـر دار بـودن قابلیـت پویـای در رونـد ترکیـب سرویس.
- قابلیت ارائه انواع سرویس‌های وب در خروجی بنا به درخواست کـاربر یـا بـا توجـه بـه سـرویس‌هـای شـرکت‌کننـده در سرویس مرکب نهایی با استفاده از توسعه‌ای برای زبان BPEL.
- تأمین مکانیزمی جهت ترکیب بازگشتی به منظور افزایش قابلیـت اسـتفاده مجـدد از سـرویس‌هـا. بـه طـوری کـه یـک سرویس مرکب، خود می‌تواند در روش پیشنهادی برای ترکیب‌های بعدی مورد استفاده قرار گیرد.
- استفاده از ایده معماری مدل‌رانه و معرفی مدل متاداده ای بـرای کنتـرل پیچیـدگی روال ترکیـب و توانـایی پشـتیبانی از تغییرات محیط سرویس‌های وب. به بیان دیگر فازهای ارائه شده برای توسعه سرویس مرکب در این روش بر اسـاس ایـده اصلی معماری مدل‌رانه که در بخش ۳.۲.۲ بیان شد، به صورت مراحل تبدیل مدل‌هایی با سطح انتزاع بالا بـه مـدل‌هـایی به صورت توصیف دقیق سرویس‌های درگیر است.

---

[۱] Metadata

[۲] Activity

از آنجا که استفاده از مدل‌های مورد استفاده در روش پیشنهادی، علاوه بر فاز ساخت سرویس مرکب، کلیه فازهای ترکیب سرویس را نیز تحت تأثیر قرار می‌داد و لازم بود تمهیداتی برای سایر فازها نیز در نظر گرفته شود، لذا در مرحله‌ی اول این تحقیق، پس از انجام مطالعات گسترده در این زمینه، به معرفی چارچوبی جامع با در نظر گرفتن ویژگی‌های بیان‌شده در روش MDCHeS می‌پردازیم. معماری مؤلفه‌های درگیر در چارچوب، وظیفه‌مندی هر مؤلفه، نحوه ارتباط بین آن‌ها، فازهای ترکیب و مدل‌های مورد نیاز در هر فاز، در این تحقیق ارائه می‌شود. اما از آنجا که تمرکز اصلی در این تحقیق، ارائه روش ترکیب و الگوریتم‌های مربوط به فاز ترکیب است به همین دلیل نهایی کردن برخی الگوریتم‌های مورد نیاز در سایر فازهای چارچوب MDCHeS به عنوان کارهای آتی در این تحقیق اشاره شده است.

همان‌طور که اشاره شد، روش ترکیب MDCHeS از اصول معماری مدل‌رانه پیروی می‌کند، لذا با نگاهی کلی، مراحل ترکیب در این چارچوب را می‌توان به صورت تبدیل مدل[۱]های معرفی شده در بخش ۳.۳ در نظر گرفت و الگوریتم‌های ارائه شده در فصل ۴ در واقع الگوریتم‌های مرتبط با تبدیل این مدل‌ها به یکدیگر هستند.

در این فصل ضمن معرفی مدل متاداده ای روش MDCHeS که به عنوان شمای مخزن اصلی چارچوب معرفی شده مورد استفاده قرار می‌گیرد، به معرفی مدل‌های مورد استفاده در چارچوب می‌پردازیم. در ادامه، چارچوب MDCHeS و مؤلفه‌های موجود در آن توصیف خواهند. برای تشریح بهتر مدل‌ها و روش ارائه شده، از مثال کاربردی "برنامه‌ریزی سفر" بهره می‌گیریم، این مثال از [۴۹] برگرفته شده و برای بیان قابلیت‌های روش پیشنهادی تکمیل شده است. همچنین از این مثال برای معرفی فازهای توسعه در فصل ۴ نیز استفاده خواهد شد. سناریوی نیازمندی کاربر از سرویس درخواستی در این مثال به شرح زیر است.

" کاربر قصد برنامه‌ریزی کردن سفری به منظور شرکت در یک رویداد[۲] را دارد، این رویداد می‌تواند شرکت در یک کنفرانس، جلسه کاری و یا یک سفر توریستی باشد. بدین منظور او لازم است در این رویداد ثبت نام کرده و برای مکان و نحوه رفت و آمد برنامه ریزی کند. او همچنین نیاز به دانستن پیش بینی آب و هوای شهر مقصد در زمان حضور در آن رویداد، را دارد."

این نیازمندی در چارچوب پیشنهادی به صورت مدل ورودی دریافت شده و به عنوان یک سرویس مرکب درخواست شده، توسعه داده می‌شود، در پایان فرآیند توسعه، به صورت مدل قابل اجرا در اختیار کاربر قرار خواهد گرفت.

---

[۱] Model Transformation

[۲] Event

۲.۳ مدل متاداده ای برای توسعه ترکیب سرویس

همان‌طور که گفته شده، روش MDCHeS ترکیب انواع سرویس‌های وب را با استفاده از ایده مدل‌رانه انجام می‌دهد. بـرای مجتمـع سازی سرویس‌های نا همگن، لایه انتزاع یکپارچه‌ای مورد نیاز است [۴۶]، که ما این لایه را در قالب مـدل متـاداده ای ارائـه داده‌ایـم. ایده اولیه این مدل از[۵۰]، [۱]، [۵۱]و [۵۲] برگرفته شده و بر اساس اهداف این تحقیق تکمیل و بهبود داده شده است. ایـن مـدل انتزاعی قابلیت توصیف همزمان نیازمندی‌های سرویس‌های وب مبتنی بر SOAP و RESTful جهت تشکیل یک سرویس مرکـب را دارد و همچنین عناصر و ارتباطات تعریف شده موجود در آن سبب ایجاد قابلیت پویایی در عناصر ترکیب، بـرای روش پیشـنهادی می‌شوند. این مدل تمامی بلاک‌های سازنده[۱] ترکیب سرویس‌ها به همراه نحـوه ارتبـاط بـین آن‌هـا را در بـردارد. ارتبـاط بـین ایـن بلاک‌های سازنده مشخص می‌کند که ترکیب بر چه اساسی انجام می‌شود. در واقع یک سرویس مرکب از نمونه‌هـایی از بـلاک‌هـای سازنده معرفی شده در این مدل تشکیل می‌شود. به بلاک‌های سازنده مدل، کلاس‌های ترکیـب و بـه هـر نمونـه از ایـن کـلاس‌هـا، عنصر ترکیب گفته می‌شود. ارتباط بین کلاس‌های ترکیب در این مدل بر اساس قوانین حرفه شناخته شده، شکل گرفته اسـت. ایـن مدل بر اساس UML برای پشتیبانی از توسعه مستقل از فناوری توصیفات ترکیب سرویس بیان شده است.

به طور کلی خودکار سازی روش‌های ترکیب سرویس مستلزم افزودن متاداده‌هایی به سرویس‌ها می‌باشد. این متاداده‌هـا یـا در قالـب توصیفات معنایی با استفاده از SAWSDL[۲] یا OWL[۳] فراهم می‌شود و یا می‌توان بدون کمک گـرفتن از توصـیفات معنـایی بـا در نظر گرفتن مفاهیمی نظیر مفهوم فعالیت برای سرویس‌های مـورد اسـتفاده در ترکیـب و تعریـف کـردن روابـط موجـود در عناصـر ترکیب با یکدیگر، به نیمه خودکار سازی روش ترکیب دست یافت. مزیت استفاده از متاداده‌ها به جای توصـیفات معنـایی و هسـتان شناسی در راه‌حل‌های سازمانی عدم نیاز به انتشار هستان‌شناسی سازمان در مواردی است که سازمان‌هـا در ایـن رابطـه محـدودیت دارند. در واقع استفاده از مفاهیم متاداده ای به صنعتی شدن روش ترکیب کمک می‌کند.

شکل ۳-۱ ساختار مدل متاداده ای روش MDCHeS، کلاس‌های ترکیب و نحوه ارتباط آن‌ها بـا یکـدیگر را نشـان مـی‌دهـد. ایـن شکل با استفاده از روش UML مدل شده است. تبدیل مدل‌های مورد اسـتفاده در ایـن روش، کـه در بخـش ۳.۳ معرفـی

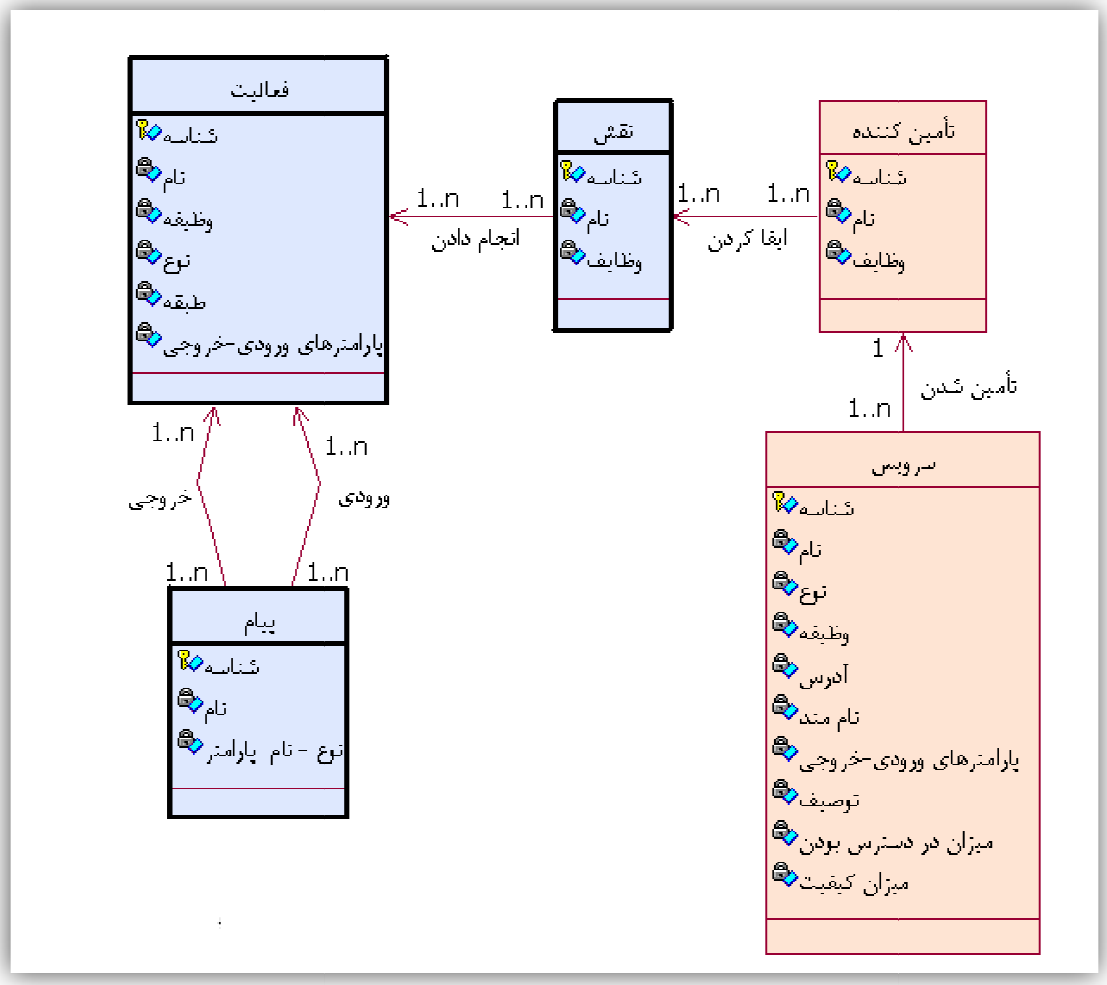
---

[۱] Building Blocks

[۲] Semantic Annotations for WSDL

[۳] Web Ontology Language

می‌شوند و همچنین شمای مخزن اصلی این چارچوب، بر اساس ساختار این مدل شکل گرفته است. از آنجا که مدل خروجی نهایی با زبان ترکیبی قابل اجرای به طور مثال BPEL، ارائه می‌شود، لذا این نکته مد نظر قرار داده شده است که عناصر این مدل با نیازمندی‌های زبان ترکیب خروجی تطابق داشته باشند تا تبدیل مدل نهایی به فرمت خروجی به آسانی امکان‌پذیر باشد. در ادامه به توصیف کلاس‌های ترکیب در این مدل متاداده ای و مشخصه[۱]های هر کدام پرداخته می‌شود.



شکل ۳-۱: ساختار مدل متاداده ای در چارچوب پیشنهادی

[۱] Attribute

۱.۲.۳ توصیف عناصر و ارتباطات مدل متاداده

در این بخش به توصیف کلاس‌های ترکیب موجود در مدل می‌پردازیم. در هر کلاس ابتدا هدف وجودی آن بیان می‌شود و در ادامه به توصیف مشخصه‌های آن کلاس می‌پردازیم.

- **فعالیت**[۱]: این کلاس یک مفهوم خوش-تعریف[۲] برای وظیفه‌مندی حرفه است. این کلاس، عنصر کلیدی متاداده‌ای مربوط به هر سرویس را شامل می‌شود. بر این اساس، سرویس‌ها با وظیفه‌مندی یکسان به یک فعالیت مشترک منجر شوند اما با ورودی خروجی‌های متفاوت. استفاده از این مفهوم باعث مجزا شدن مراحل اولیه ترکیب از سرویس‌های موج می‌شود و تنها بر اساس مفهوم وظیفه‌مندی[۳] تعیین شده‌ی کاربر، مدل ترکیب اولیه شکل می‌گیرد. (در این تحقیق به اموری که کاربر برای سرویس مرکب مورد نیاز خود درخواست می‌دهد، وظیفه‌مندی گوییم. به طور مثال برای سرویس مرکب "برنامه ریزی سفر" کاربر وظیفه‌مندی‌های رزرو بلیط هواپیما، رزرو هتل و رزرو ماشین را درخواست می‌دهد) هر وظیفه‌مندی درخواست شده‌ی کاربر، به فعالیتی متناظر موجود در چارچوب، منطبق می‌شود و در ادامه روش تنها با این مفهوم مدل ترکیب مرحله به مرحله دقیق‌تر می‌گردد. در این مدل فرض بر آن است که هر فعالیت می‌تواند توسط چندین نقش انجام شود.

مشخصه‌های کلاس فعالیت عبارتند از: شناسه، نام، وظیفه، نوع، طبقه، پارامترهای ورودی، پارامترهای خروجی.

- شناسه، نام: این دو ویژگی در تمامی کلاس‌های مدل پیشنهادی وجود دارند. شناسه ویژگی یکتایی است که به عنوان کلید اصلی در کلیه عناصر استفاده می‌شود. نام هم بیانگر نام تعیین شده برای آن عنصر است و در مراحل جستجو مورد استفاده قرار می‌گیرد. در ادامه از معرفی این دو مشخصه به دلیل شباهت، در سایر کلاس‌ها خودداری می‌کنیم.
- وظیفه[۴]: بیانگر وظیفه‌مندی اصلی این فعالیت است و به عنوان عامل اصلی در ارتباط با سایر عناصر ترکیب در مدل استفاده می‌شود. فرض بر آن است که هر فعالیت تنها یک وظیفه معین را بر عهده دارد.
- نوع[۱]: نوع هر فعالیت می‌تواند اتمیک یا مرکب باشد. از آنجا که سرویس‌های مرکب تولید شده در روش MDCHeS تحت عنوان فعالیت مرکب برای استفاده مجدد، در *مخزن اصلی* ذخیره می‌شوند، لذا به ازای هر

---

[۱] Activity
[۲] Well-defined
[۳] Functionality
[۴] Function

فعالیت، اتمیک یا مرکب بودن آن با این مشخصه معین می‌شود. منظور از فعالیت مرکب، فعالیتی است که از چندین فعالیت اتمیک برای انجام وظیفه‌مندی خاص ترکیب شده است. به طور مثال فعالیت مرکب با وظیفه "برنامه‌ریزی سفر" از فعالیت‌های اتمیک رزرو بلیط هواپیما، رزرو هتل و رزرو ماشین تشکیل شده است. مدل‌های توسعه داده شده در حین روال ترکیب یک فعالیت مرکب، در *مخزن مدل‌ها* از چارچوب ذخیره می‌شوند. از آنجا که فرض بر آن است که هر فعالیت تنها یک وظیفه معین بر عهده دارد، لذا برای فعالیت‌های مرکب تولید شده لازم است وظیفه‌ای مشخص نیز تعیین گردد.

- طبقه: این ویژگی در برگیرنده دسته‌بندی این فعالیت، از بین دسته‌بندی معرفی شده در شاخص استاندارد برای دسته‌بندی صنعت[۲] [۵۳] است. از این ویژگی برای مرحله کشف فعالیت و دقیق‌تر کردن نتایج جستجو با مقایسه با طبقه معرفی شده برای وظیفه‌مندی درخواستی کاربر، استفاده می‌شود. بنابراین لازم است کاربر به ازای هر وظیفه‌مندی درخواستی خود، طبقه استاندارد آن را نیز تعیین کند.
- پارامترهای ورودی-خروجی: هر فعالیت تنها یک وظیفه را انجام می‌دهد، اما ممکن است برای انجام آن وظیفه با مجموعه‌هایی متفاوت از پارامترهای ورودی- خروجی در ارتباط باشد. پارامترهای ورودی- خروجی فعالیت، پارامترهای عمومی سرویس‌هایی هستند که این وظیفه را بر عهده دارند.

- **پیام**[۳]: ورودی و خروجی هر فعالیت در قالب عناصر کلاس پیام، معرفی می‌شود. این کلاس در برگیرنده پارامترهای پیام‌هایی است که به عنوان پیام‌های ورودی و خروجی با کلاس فعالیت در ارتباطاند. هر عنصر فعالیت می‌تواند با یک یا بیش از یک عنصر پیام ورودی-خروجی در ارتباط باشد. کلاس پیام شامل پارامترهای عمومی است که برای انجام آن وظیفه‌مندی به عنوان ورودی-خروجی استفاده می‌شوند. این کلاس شامل مشخصه‌های شناسه، نوع و نام پارامترها می‌باشد.
  - نوع - نام پارامتر: این مشخصه بیانگر مجموعه‌ای از پارامترها و نوع هر کدام است. این مجموعه به عنوان پیام ورودی یا پیام خروجی با عنصر فعالیت در ارتباطاند. نحوه نامگذاری پارامترها باید به صورت استاندارد باشد.

---

[۱] Type
[۲] Industry Classification Benchmark (ICB)
[۳] Message

- **نقش**[1]: این کلاس یک کلاس انتزاعی است که مسئولیت اجرای یک فعالیت را بر عهده دارد. دلیل وجود عنصر نقش افزودن قابلیت تغییر پذیری و سازگاری در اضافه و کم شدن تأمین‌کنندگان در محیط واقعی است. به طور مثال برای فعالیت باز کردن حساب، نقش بانک وجود دارد که یکی از وظایف آن باز کردن حساب است و از طرف دیگر بانک ملت به عنوان یکی از تأمین‌کنندگان مرتبط با این نقش به شمار می‌آید. این کلاس دارای مشخصه‌های: شناسه، نام و وظایف است.
  - وظایف: این مشخصه شامل تمامی وظایفی است که این نقش می‌تواند ایفا کند. در واقع به ازای هر ارتباطی از این عنصر با عنصر فعالیت، یک وظیفه به وظایف آن افزوده می‌شود. هر عنصر نقش می‌تواند یک یا بیش از یک وظیفه را بر عهده داشته باشد بدین معنا که هر عنصر نقش می‌تواند با یک یا بیشتر از یک عنصر فعالیت در ارتباط باشد.

سه عنصر فعالیت، پیام و نقش که تا به حال معرفی شده‌اند عناصر انتزاعی هستند که برای مستقل کردن مدل انتزاعی سرویس مرکب از سرویس و عناصر مرتبط با آن استفاده می‌شوند. در شکل ۳-۱با خطوط پررنگ حاشیه مشخص شده‌اند. این سه عنصر توسط معمار حرفه به هر سرویس اضافه شده به *مخزن اصلی* مرتبط می‌شوند تا در مراحل جستجو و ایجاد مدل انتزاعی سرویس مرکب درخواستی، مورد استفاده قرار گیرند.

- **تأمین‌کننده:** هر تأمین‌کننده، در واقع مسئول فراهم کردن یک سرویس واقعی در زمان اجرا است. در صورتی که یک تأمین‌کننده قابلیت فراهم کردم تمامی وظایف یک نقش را داشته باشد، می‌تواند آن نقش را بر عهده بگیرد و با آن در ارتباط باشد. هر تأمین کننده می‌تواند با یک یا بیش از یک نقش در ارتباط باشد. به طور مثال بانک تجارت می‌تواند به عنوان تأمین کننده، نقش بانک به شمار آید. هر نقش نیز می‌تواند با یک یا بیشتر از یک تأمین کننده در ارتباط باشد. هر تأمین‌کننده شامل مشخصه‌های شناسه، نام، وظایف است.
  - وظایف: این مشخصه شامل تمامی وظایفی است که این تأمین‌کننده قادر به تأمین سرویس مورد نیاز برای آن‌ها در زمان اجرا است.
- **سرویس**: کلاس سرویس شامل تمامی مشخصه‌های رایج برای یک سرویس وب است که طبق نیازمندی‌های مطرح شده در فایل BPEL، برای فراخوانی یک سرویس در زمان اجرا نیاز است. این کلاس شامل مشخصه‌هایی است که امکان توصیف هر دو نوع سرویس مبتنی بر SOAP و RESTful را دارا باشد. روال افزودن عنصر سرویس به مخزن به این طریق است که به ازای

[1] Role

افزوده شدن هر سرویس جدید، فعالیت، پیام و نقش مربوط به آن معین شده و در صورتی که از قبل موجود نباشد، اضافه می‌شود، پس از آن لازم است تأمین‌کننده سرویس و سایر ویژگی‌های مربوط به عنصر سرویس افزوده شود و ارتباط بین نقش، تأمین‌کننده و سرویس برقرار گردد. این حالت برای تمامی ارتباط‌های نشان داده شده در شکل ۳-۱برقرار است. مشخصه‌های این کلاس عبارتند از:

- شناسه و نام سرویس.
- نوع: مشخصه نوع سرویس، بیانگر دو نوع مختلف از سرویس‌های وب است. سرویس افزوده شده یا از نوع سرویس وب مبتنی بر SOAP و یا او نوع سرویس وب RESTful است.
- وظیفه: بیانگر وظیفه اصلی است که توسط این سرویس انجام می‌شود.
- آدرس: این مشخصه url[۱] سرویس وب را بیان می‌کند. مشابه Endpoint در WSDL نسخه ۲.۰ است.
- نام متد[۲]: . این مشخصه در سرویس‌های وب مبتنی بر SOAP، نام عملیاتی[۳] است که به هنگام فراخوانی سرویس همراه با URL و پارامترهای ورودی- خروجی سرویس، فراخوانی می‌شود و در سرویس‌های وب RESTful بیانگر یکی از متدهای HTTP (GET, DELETE, POST و PUT) برای آن سرویس که در هنگام فراخوانی به همراه پارامترهای آن سرویس استفاده می‌شود. لازم به ذکر است که در روش پیشنهادی فرض بر آن است که هر سرویس چه سرویس‌های مبتنی بر SOAP و چه سرویس‌های RESTful تنها قادر به انجام یک عملیات است و سرویسی با دو عملیات، به عنوان دو سرویس مجزا در مدل پیشنهادی معرفی می‌شود.
- پارامترهای ورودی-خروجی: در سرویس‌های مبتنی بر SOAP، این کلاس بیانگر مجموعه پارامترهای ورودی-خروجی سرویس و در سرویس‌های RESTful به عنوان مجموعه پارامترهایی است که به همراه URL سرویس در هر یک از متدهای HTTP ارسال می‌شوند.
- توصیف: این پارامتر آدرس فایل توصیفی سرویس وب را در صورت وجود شامل می‌شود. برای سرویس‌های وب مبتنی بر SOAP، آدرس فایل WSDL است. برای سرویس‌های RESTful این پارامتر می‌تواند مقداری نداشته

---

[۱] Uniform Resource Locator
[۲] Method Name
[۳] Operation

باشد (اختیاری است) یا آدرس فایل WADL[1] باشد که برای توصیف برخی سرویس‌های RESTful استفاده می‌شود.

- نیازمندی‌های غیر عملکردی: سایر مشخصه‌های کلاس سرویس مربوط به نیازمندی‌های غیر عملکردی است که روش ترکیب پیشنهادی پشتیبانی می‌کند. از آنجا که روش پیشنهادی قابلیت افزوده شدن نیازمندی‌های غیر عملکردی بیشتری را دارد، پشتیبانی از ویژگی‌های بیشتر به عنوان کارهای آتی این تحقیق مطرح شده است، نیازمندی‌های کنونی عبارتند از:
  - میزان در دسترس بودن: بیانگر میزان در دسترس بودن سرویس وب مورد نظر در زمان اجرا است، این میزان با توجه به توصیف تأمین‌کننده سرویس مشخص می‌شود. مقادیر این مشخصه می‌تواند، بالا(۳)، متوسط(۲) و پایین(۱) باشد.
  - میزان کیفیت: بیانگر میزان کیفیت سرویس ارائه شده است. مقادیر این مشخصه نیز می‌تواند، بالا(۳)، متوسط(۲) و پایین(۱) باشد که توسط تأمین‌کننده سرویس تعیین شده است.

دو عنصر تأمین‌کننده و سرویس عناصری هستند که به ویژگی سرویس‌ها در محیط واقعی وابسته هستند و لازم است به صورت دوره‌ای ویژگی‌های آن‌ها بروز گردد. این دو عنصر با خطوط حاشیه‌ای کم رنگ در شکل ۳-۱ نشان داده شده‌اند.

همان‌طور که در بخش ۱.۲.۳ مطرح شد، نحوه ارتباط کلاس‌های معرفی شده در مدل متاداده با استفاده از قوانین حرفه مفروض در چارچوب MDCHeS شکل گرفته است. برای تشریح بهتر کلاس‌های مدل متاداده و نحوه ارتباط آن‌ها، جدول ۷-۱ در پیوست ۱ نمایی از عناصر موجود در مخزن اصلی برای پاسخگویی به نیاز درخواستی سناریوی "برنامه ریزی سفر" را نشان می‌دهد.

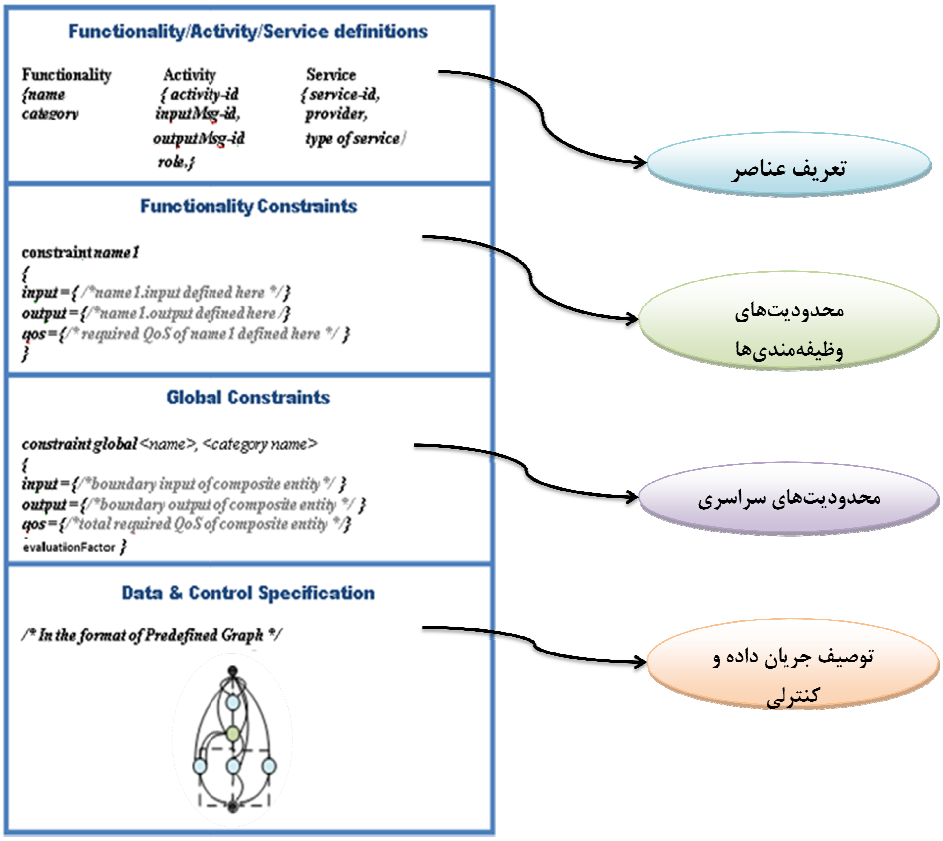
در روش MDCHeS فرض بر آن است که مخزن اصلی بر اساس کلاس‌های معرفی شده، ایجاد و عناصر ترکیب که به عنوان نمونه‌هایی از هر کلاس هستند، قبل از انجام ترکیب، توسط معمار حرفه به آن افزوده شده است. ارائه مکانیزمی جهت افزودن خودکار عناصر به مخزن با توجه به ویژگی‌های سرویس‌های منتشر شده در وب و مورد استفاده در چارچوب و یا بروز رسانی ویژگی‌های آن‌ها به صورت دوره‌ای، در محدوده تمرکز این تحقیق قرار نمی‌گیرد. با توجه به ارتباطات مطرح شده کلاس‌های ترکیب، نیاز به معمار حرفه برای افزودن عناصر به مخزن اصلی هم در مرحله آغازین و هم در مراحلی که پس از انجام ترکیب

[1] Web Application Description Language

نهایی در مراحل پایانی (فاز انتخاب) توسعه سرویس مرکب، نیاز به اضافه شدن عنصر به مخزن است، داریم. نحوه تکمیل مخزن از عنصر سرویس آغاز می‌شود، با اضافه شدن سرویس جدیدی به مخزن اصلی چارچوب، لازم است بر اساس ویژگی‌های سرویس جدید، مشخصه‌های عنصر سرویس تکمیل گردد در ادامه لازم است با توجه به ارتباط کلاس سرویس با کلاس تأمین‌کننده در شکل ۳-۱ ، در صورتی که تأمین‌کننده این سرویس جدید در بین عناصر تأمین‌کننده موجود در *مخزن اصلی* وجود نداشت، به این جدول افزوده شود و به دنبال آن عناصر موجود در کلاس نقش، فعالیت و پیام بروز گردند، سرویس‌های مرکبی که توسط چارچوب MDCHeS تولید می‌شوند با عنصر تأمین‌کننده MDCHeS افزوده می‌شوند و سایر عناصر دیگر مدل به طور متناظر بروز می‌گردند.

۳.۳ معرفی مدل‌های مورد استفاده

در روش MDCHeS برای غلبه بر پیچیدگی روند ترکیب سرویس‌های وب مدل‌هایی در هر فاز از ترکیب استفاده می‌شوند و در طول فرآیند ترکیب به یکدیگر تبدیل می‌شوند. الگوریتم‌هایی که در فصل ۴ معرفی می‌شوند تبدیل این مدل‌ها را بر اساس ساختار مدل متاداده ای معرفی شده در بخش ۲.۳ انجام می‌دهند. شکل ۳-۲ الگوی مورد استفاده در هر یک از سه مدل معرفی شده در ادامه این بخش را نمایش می‌دهد. در این الگو که از چهار بخش اصلی تشکیل شده است، نیازمندی‌های هر مدل بر اساس فاز توسعه‌ای که آن مدل مورد استفاده قرار می‌گیرد، تشریح می‌شود.



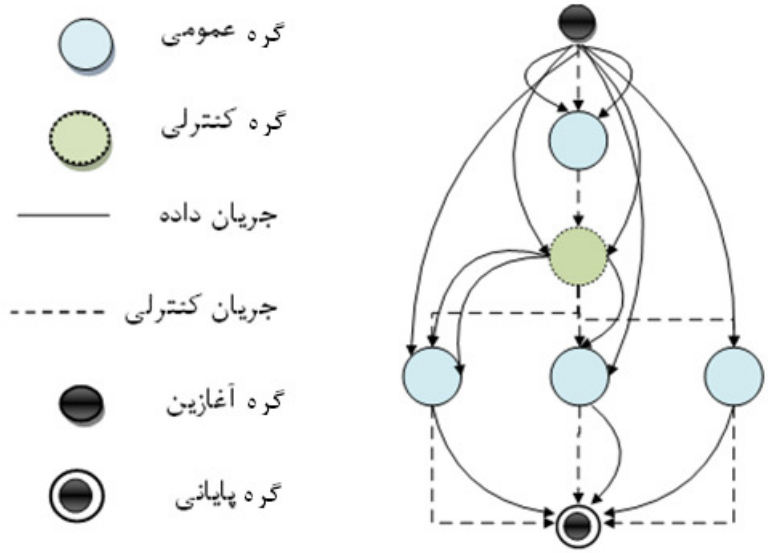
شکل ۳-۲ : الگوی مدل‌های مورد استفاده در روش پیشنهادی

همان‌طور که در توصیفات مختصر هر بخش در شکل ۳-۲ مشخص شد، برای هر یک از مدل‌هایی که در ادامه معرفی می‌شوند، هر بخش از این الگو بیان‌گر عناصر مورد استفاده در آن مدل است. بخش پایانی معرفی شده در الگوی شکل ۳-۲، بیانگر نمودار جریان داده[۱] و گراف جریان کنترلی[۲] است. (نمودار جریان داده نمایشی گرافیکی از جریان داده‌ها مابین مؤلفه‌های یک سیستم نرم‌افزاری است. در نمودار جریان داده، اطلاعی در مورد نحوه پردازش وجود ندارد[۵۴]) (گراف جریان کنترل، گرافی است که جریان کنترلی ما بین بلاک‌های سازنده یک سیستم نرم‌افزاری را مشخص می‌کند. این گراف از گره‌ها و لبه‌ها تشکیل شده است[۵۵]). برای نمایش همزمان جریان داده‌ای و کنترلی در روش پیشنهادی گرافی معرفی می‌شود که شامل دو نوع گره و دو نوع لبه[۳] است. در شکل ۳-۳ الگوی این گراف به همراه معرفی گره‌ها و لبه‌های آن نشان داده شده است.

---

[۱] Data Flow Diagram (DFD)
[۲] Control Flow Graph (CFG)
[۳] Edge

گره عمومی در هر مدل بیانگر موجودیت متفاوتی است به طور مثال در مـدل ورودی بیـانگر وظیفـه‌منـدی درخواسـتی کـاربر و در مدل انتزاعی ترکیب سرویس، بیانگر فعالیت است. جریان داده نیز برای هر مـدل بیـان‌گـر پارامترهـای ورودی-خروجـی مـرتبط بـا موجودیت گره عمومی است. گره کنترلی و جریان کنترلی بیانگر ترتیب و نحوه اجرای موجودیت‌های گره عمومی است.



شکل ۳-۳: الگوی گراف جریان کنترل-داده در روش پیشنهادی

لازم به ذکر است که لبه‌های خروجی گره آغازین بیانگر، ورودی‌های مرزی سـرویس درخواسـتی و ورودی‌هـای گـره پایـانی بیـانگر خروجی‌های مرزی سرویس درخواستی هستند. این ورودی و خروجـی‌هـا بـه ورودی- خروجـی گـره‌هـای عمـومی گـراف مـرتبط می‌شوند. در ادامه در معرفی هر مدل به تفصیل عناصر موجود در هر بخش از مدل، تشریح می‌شوند.

۱.۳.۳ مدل ورودی (IM[1])

مدل ورودی، نمونه‌ای برگرفته از الگوی معرفی شده در شکل ۳-۲ است که برای بیان ویژگی‌های مرتبط با وظیفه‌منـدی مـورد نظـر کاربر، محدودیت‌ها و نحوه ارتباط آن‌ها با یکدیگر برای پاسخ‌گویی به سرویس مرکب درخواستی کاربر مورد استفاده قـرار مـی‌گیـرد. این مدل بر اساس اطلاعات ورودی کاربر، تکمیل می‌گردد و به عنوان ورودی چارچوب MDCHeS در نظر گرفته می‌شود.

- همان‌طور که در شکل ۳-۴ برای مثال "برنامه ریـزی سـفر" نشـان داده شـده اسـت، در بخـش اول کـه مربـوط بـه توصـیف وظیفه‌مندی‌هاست، کلیه وظیفه‌مندی‌هایی که کاربر در سرویس مرکب درخواستی نیاز دارد، معرفی می‌شود. وظیفـه‌منـدی در

[1] Input Model

واقع انتزاعی از سرویس درخواستی کاربر است که کاربر برای توصیف سرویس درخواستی خود، به بیان ویژگی‌های مورد نیاز آن می‌پردازد. لازم است برای هر وظیفه‌مندی نام آن به همراه طبقه استاندارد معرفی شده در شاخص استاندارد دسته‌بندی صنعت[ICB[۵۳، بیان شود. این طبقه با طبقه‌ای که به عنوان مشخصه در کلاس فعالیت معرفی شد، یکسان است.

- بخش دوم که مربوط به توصیف محدودیت‌ها و ویژگی‌های وظیفه‌مندی‌هاست شامل لیستی از پارامترهای ورودی-خروجی هر وظیفه‌مندی به همراه میزان ویژگی کیفی است که ممکن است برای آن وظیفه‌مندی توسط کاربر تعیین شود، است.
- بخش سوم این مدل، محدودیت‌ها و ویژگی‌های سرویس مرکب درخواستی را بیان می‌کند. این بخش پارامترهای ورودی-خروجی مرزی این سرویس مرکب را تعیین می‌کند. همچنین ویژگی‌های کیفی سرویس مرکب درخواستی که توسط کاربر تعیین می‌شود، در این بخش بیان می‌شود.
- بخش آخر این مدل، گراف جریان کنترل-داده را معین می‌کند. گره عمومی در گراف داده-کنترل IM، بیانگر وظیفه‌مندی‌ها و لبه‌های جریان داده‌ای در این گراف نشان‌دهنده پارامترهای ورودی-خروجی هر وظیفه‌مندی است. لبه‌های خروجی گره آغازین بیانگر ورودی‌های مرزی و ورودی‌های گره پایانی بیانگر خروجی‌های مرزی وظیفه‌مندی مرکب درخواستی است.

از نقطه‌نظر معماری مدل‌رانه، IM به عنوان یک PIM در سطح انتزاع بالایی مطرح است. (در بخش ۱.۳.۲.۲ با توصیف این مدل آشنا شدیم). از آنجا که در مدل ورودی، از مفهوم وظیفه‌مندی استفاده می‌شود و همچنین جریان کنترل و داده مابین وظیفه‌مندی‌ها نیز تعیین می‌گردد، لذا به عنوان مفهومی مستقل از ساختار و عناصر مورد استفاده در چارچوب، ارائه می‌شود و کاربری که این مدل را تکمیل می‌کند تنها نیاز به دانش زمینه دارد.

شکل ۳-۴ نمونه‌ای از مدل ورودی برای مثال تشریح شده‌ی " برنامه ریزی سفر" در بخش ۱.۳ است. در بخش دوم این مدل، به منظور خوانایی بیشتر، تنها محدودیت‌ها و ویژگی‌های مربوط به یک وظیفه‌مندی به عنوان نمونه آورده شده است. هر یک از بخش‌های این مدل در فرآیند توسعه‌ی ترکیب سرویس مورد استفاده قرار می‌گیرند.

Functionality definitions

//defining Functionality names & related category;

```
functionality     (planBooking,          *.Airlines;
functionality     hotelShuttleBooking,   *.Hotels;
functionality     eventRegistrating,     *.Travel & Tourism;
functionality     carBooking,            *.Automobiles & Parts;
functionality     weatherForcasting,     *.Internet;)
```

Functionality Constraints

// defining Input, Output and QoS of each Functionality;

```
constraint planBooking

input = { depaturePlace;
          dateLeave;
          arrivePlace;
          dateBack; }

output = { planBookingId; }
```

// only one functionality for instance illustrates here.

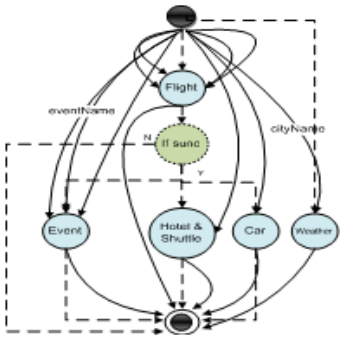
Global Constraints

// defining Boundary Input, Output and QoS of Requested composite functionality;

```
constraint global *.Travel & Tourism; {
input = {  startDate;
           finishDate;
           destination;
           source;
           eventName; }

output = {  planBooking_id;
            hotelBooking_id;
            shuttleBooking_id;
            carRental_id:
            carRental_id;
            weatherForcasted;
            carPickupAddress;}
qos = {
           quality = 2;
           availability = 2; }
```

Data & Control Specification



شکل ۳-۴: IM در سناریوی "برنامه ریزی سفر"

۲.۳.۳ مدل انتزاعی سرویس مرکب (ACSM[۱])

ACSM نمونه‌ای برگرفته از الگوی معرفی شده در شکل ۳-۲ بوده که برای بیان ویژگی‌های مرتبط با فعالیت‌های کشف شده برای هر وظیفه‌مندی شامل ورودی-خروجی و نقش آن فعالیت و نحوه ارتباط آن‌ها با یکدیگر مورد استفاده قرار می‌گیرد. این مدل در مراحل ترکیب در روش پیشنهادی از تبدیل IM بدست می‌آید.

- همان‌طور که در شکل ۳-۵ برای مثال "برنامه ریزی سفر" نشان داده شده است، بخش اول این مدل شامل شناسه فعالیت‌ها، نقش‌ها و پیام‌های ورودی-خروجی است که در فاز کشف روش MDCHeS برای هر وظیفه‌مندی و پارامترهای متناظر آن بدست آمده است و در مجموع سرویس مرکب درخواستی کاربر را پاسخگو هستند. از آنجا که هر فعالیت در روش پیشنهادی بر اساس شناسه متناظر در *مخزن اصلی* شناخته می‌شود، در این بخش شناسه آن مورد استفاده قرار می‌گیرد.
- بخش دوم الگوی نشان داده شده در شکل ۳-۲ تنها در مدل ورودی استفاده می‌شود و از آنجا که این محدودیت‌های بیان شده در این فاز تنها در کشف فعالیت مناسب با آن وظیفه‌مندی کاربرد دارد، در سایر مدل‌ها معادلی برای این بخش وجود ندارد.
- بخش دوم در ACSM، محدودیت‌ها و ویژگی‌های سرویس مرکب درخواستی را که از مجموعه فعالیت‌های کشف‌شده برای وظیفه‌مندی‌ها تشکیل‌شده است، بیان می‌کند. این بخش پارامترهای ورودی- خروجی مرزی این سرویس مرکب را بر اساس پیام‌های ورودی-خروجی فعالیت‌های متناظر تعیین می‌کند. همچنین ویژگی‌های کیفی سرویس مرکب درخواستی همانند مدل ورودی، به صورت کلی در این بخش مشخص می‌شود و برابر همان مقادیر درخواست شده‌ی کاربر از این ویژگی‌ها می‌باشد.
- بخش آخر این مدل، گراف جریان کنترل-داده را معین می‌کند. گره عمومی در گراف داده-کنترل ACSM بیانگر شناسه فعالیت، شناسه نقش متناظر، شناسه پیام ورودی و شناسه پیام خروجی آن است و لبه‌های جریان داده‌ای در این گراف نشان‌دهنده پارامترهای ورودی-خروجی هر فعالیت است که در قالب پیام‌های ورودی-خروجی در *مخزن اصلی* چارچوب قرار دارند. لبه‌های خروجی گره آغازین بیانگر ورودی‌های مرزی و ورودی‌های گره پایانی بیانگر خروجی‌های مرزی فعالیت مرکب درخواستی است. از آنجا که ممکن است به ازای هر وظیفه‌مندی درخواستی کاربر، فعالیت اتمیک متناظر یافت نشود، ممکن

---

[۱] Abstract Composite Service Model

است به جای یک وظیفه‌مندی مجموعه‌ای از فعالیت‌ها که بر اساس الگوریتم ترکیب، ایجاد شده‌اند جایگزین شوند. در واقع در این حالت گراف جریان داده-کنترل ACSM با IM متفاوت خواهد بود.

از نقطه‌نظر معماری مدل‌رانه، ACSM به عنوان یک PIM مطرح است. (در بخش ۲.۳.۲.۲ با این مدل آشنا شدیم و گفتیم که PIM برای بیان توصیف ویژگی‌ها مستقل از سکو مورد استفاده قرار می‌گیرد.)

در این تحقیق منظور از سکو که در مدل PIM و PSM اشاره می‌شود در واقع فناروی مورد استفاده برای توصیف فرآیند ترکیب است که این فناروی به طور مثال می‌تواند BPEL، OWL-S، Web Components و یا Algebraic Process Composition [۲۳] باشد. از آنجا که در مدل ACSM از عناصر انتزاعی ترکیب استفاده می‌شود، لذا امکان انطباق این مدل با هر نوع فناروی برای توصیف فرآیند ترکیب در انتهای مراحل توسعه سرویس مرکب امکان‌پذیر است لذا این مدل، مدلی مستقل از فناروی توصیف فرآیند ترکیب است.

در شکل ۳-۵ نمونه یکی از مدل‌های ACSM بدست آمده متناظر با IM برای مثال تشریح شده " برنامه ریزی سفر" ، نشان داده شده است. برای خوانایی بهتر ACSM، نام فعالیت‌ها نیز در بخش اول مدل آورده شده است در بخش ۲.۴ که فرآیند توسعه ترکیب سرویس معرفی می‌شود، نحوه بدست آمدن هر یک از بخش‌های این مدل‌ها بیان می‌گردد.

مدل انتزاعی سرویس مرکب، مدلی است که وابسته به ویژگی سرویس‌های موجود در مخزن اصلی نبوده و سطح انتزاع مناسبی برای توصیف استاندارد سرویس مرکب درخواستی کاربر در چارچوب پیشنهادی دارد.

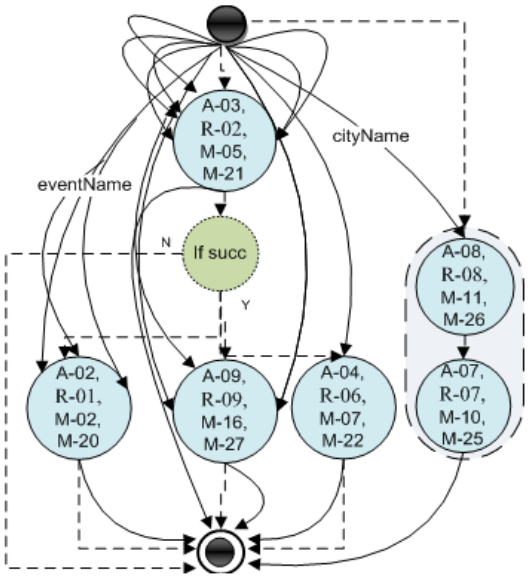
Activity definitions

```
// defining activity-id; InputMsg-id, OutputMsg-id;
activity  PlaneReservation            {A-03,  R-02,  M-05,  M-21;
activity  HotelShuttleReservation     A-09,   R-01,  M-16,  M-27;
activity  EventRegistration           A-02,   R-09,  M-02,  M-20;
activity  CarReservation              A-04,   R-06,  M-07,  M-22;
activity  CityGeographicalGetting     A-08,   R-08,  M-11,  M-26;
activity  CityGeographicalGetting     A-07,   R-07,  M-10,  M-25; }
```

Global Constraints

```
// defining Boundary Input, Output and QoS of Requested composite activity;
 constraint global *.Travel & Tourism; {
input = {
        date    DepartDate;
        date    ReturnDate;
        string  To;
        string  From;
        string  eventName;
        boolean Journey type }
output = {
        num     flightNum;
        num     seatNum;
        string  flightBooking_ID;
        string  eventParticipant _ID
        string  hotelBooking_ID;
        string  shuttleBooking_ID;
        string  carBooking _ID;
        string  forcasted WeatherOfCity;
qos = {
        quality = 2;
        availability = 2; } }
```

Data & Control Specification



شکل ۳-۵: نمونه‌ای از ACSM در سناریوی "برنامه ریزی سفر"

۳.۳.۳ مدل واقعی سرویس مرکب (CCSM[۱])

CCSM نیز مانند IM و ACSM نمونه‌ای برگرفته از الگوی معرفی شده در شکل ۳-۲ است که برای بیان ویژگی‌های مرتبط با سرویس‌های متناظر با فعالیت و پیام‌های ورودی-خروجی و نحوه ارتباط آن‌ها با یکدیگر مورد استفاده قرار می‌گیرد. این مدل در مراحل ترکیب در روش پیشنهادی از تبدیل ACSM بدست می‌آید.

- همان‌طور که در شکل ۳-۶ برای مثال "برنامه ریزی سفر" نشان داده شده است، بخش اول این مدل شامل شناسه سرویس، شناسه تأمین‌کننده و در نهایت نوع سرویس متناظر با زوج فعالیت و ورودی-خروجی در IM است.
- بخش دوم در CCSM، ویژگی‌های کیفی سرویس مرکب درخواستی حاصل از ترکیب سرویس‌های معرفی شده در بخش اول را بر نیز اساس شاخص‌های از پیش تعریف شده برای محاسبه ویژگی کیفی سراسری یک سرویس مرکب، معین می‌کند. این بخش برای انتخاب CCSM نهایی در مرحله انتخاب استفاده می‌شود. فاکتور ارزیابی[۲] موجود در این بخش نیز بر اساس ویژگی‌های کیفی تعیین شده در این قسمت و سایر شاخص‌هایی که در فاز انتخاب در فرآیند توسعه بخش ۱.۴.۲.۴ تعریف شده است، تکمیل می‌گردد.
- بخش آخر این مدل، گراف جریان کنترل-داده را معین می‌کند. این گراف مشابه گراف داده-کنترل ASCM متناظر است با این تفاوت که گره‌های عمومی، بیانگر شناسه سرویس، شناسه نقش و شناسه تأمین‌کننده متناظر با فعالیت و پیام‌های ورودی-خروجی گراف ACSM است.

از نقطه‌نظر معماری مدل‌رانه، CCSM به عنوان یک PIM مطرح است. مدل‌های معرفی شده جنبه‌های اصلی مدل‌سازی سرویس وب و فرآیند حرفه (سرویس مرکب) را در بر دارند. این مدل‌ها توسعه‌دهنده سرویس را قادر می‌سازد بدون در نظر گرفتن فناوری، سرویس وب مرکب را توصیف کند.

در شکل ۳-۶ نمونه یکی از مدل‌های CCSM بدست آمده متناظر با ACSM شکل ۳-۵ برای مثال تشریح شده " برنامه ریزی سفر" نشان داده شده است.

---

[۱] Concrete Composite Service Model
[۲] Evaluation Factor

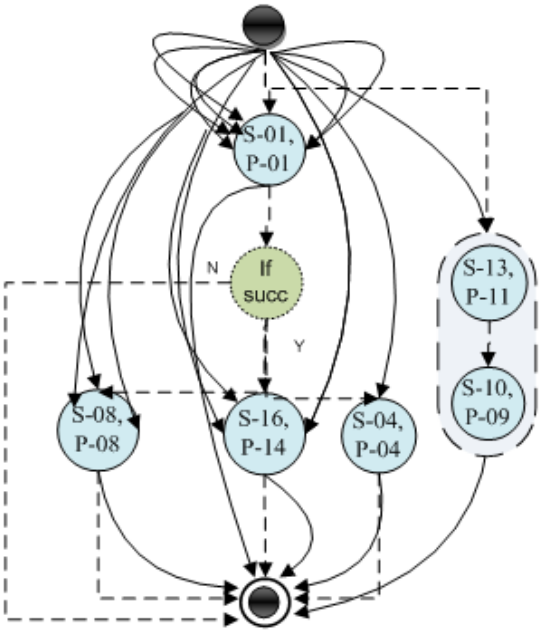
Service definitions

// defining Service-id, Role-id, Provider-id and Service-type;
service { S-01, P-01, SOAP-based;
service S-08, P-08, RESTful;
service S-16, P-14, SOAP-based;
service S-04, P-04, RESTful;
service S-13, P-11, RESTful;}
service S-10, P-09, SOAP-based;}

Global Constraints

// defining total QoS of Requested composite service;
constraint global {
qos = {
quality = 2.6;
availability = 2.5; }
evaluationFactor=0.8 *// it should be completed in Selection phase;*}

Data & Control Specification

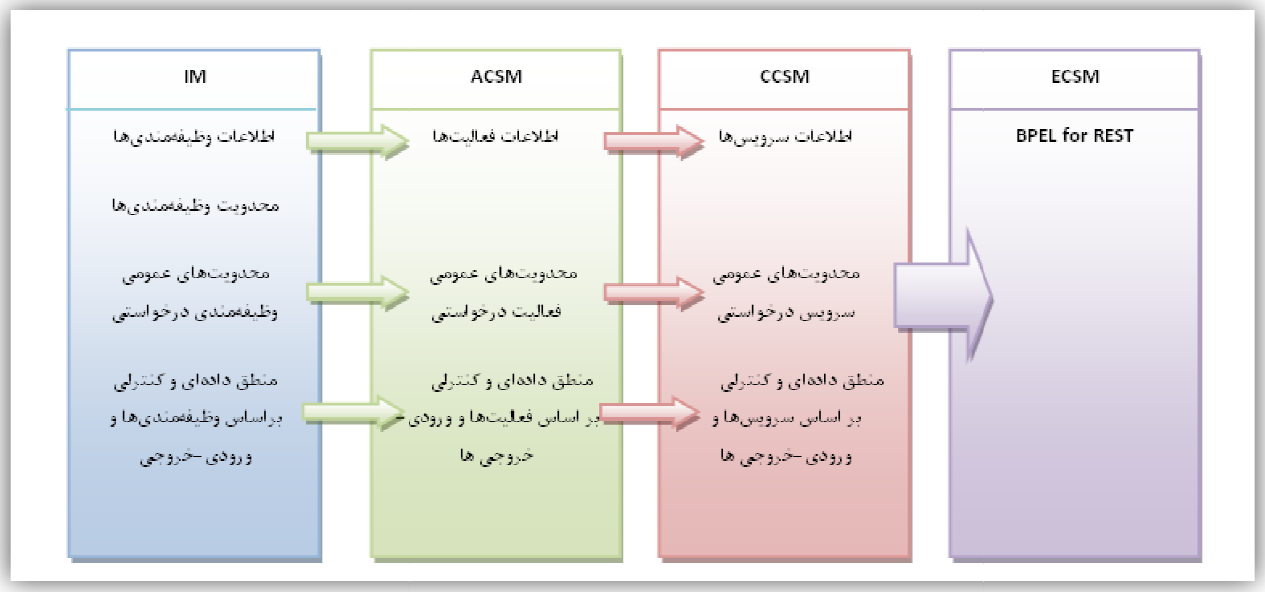


شکل ۳-۶: نمونه‌ای از CCSM در سناریوی "برنامه ریزی سفر"

۴.۳.۳ مدل اجرایی سرویس مرکب (ECSM[۱])

خروجی نهایی چارچوب MDCHeS، ECSM است، که از تبدیل CCSM نهایی در مراحل پایانی ترکیب، بدست می‌آید. این مدل، سرویس مرکب را بر اساس زبان قابل اجرای ترکیب سرویس، بر مبنای فناوری خاصی بیان می‌کند. در چارچوب پیشنهادی به دلایل ویژگی‌های زبان BPEL for REST [۴۴] و [۲۲]. که توسعه‌ای از BPEL به منظور پشتیبانی از فراخوانی سرویس‌های RESTful در خلال فرآیند ترکیب و ارائه سرویس RESTful به عنوان خروجی اجرای فرآیند، است که به طور کامل در بخش ۱۷.۱.۳.۲ معرفی شد، استفاده می‌گردد. از آنجا که این مدل سرویس مرکب را با زبانی وابسته به فناوری معرفی می‌کند، از نقطه‌نظر معماری مدل‌رانه، به عنوان یک مدل PSM مطرح است. (در بخش ۳.۳.۲.۲ با این مدل آشنا شدیم و گفتیم که PSM برای بیان توصیف ویژگی‌ها بر اساس سکویی خاص مورد استفاده قرار می‌گیرد.)

از آنجا که این مدل بر اساس فناوری BPEL for REST توصیف می‌شود لذا مدلی وابسته به فناوری تلقی می‌گردد. قبل از آنکه به معرفی چارچوب پیشنهادی بپردازیم، لازم است دید جامعی نسبت به مدل‌های معرفی شده در این بخش و ارتباط آن‌ها با یکدیگر داشته باشیم. شکل ۳-۷ بیانگر ارتباط بخش‌های مختلف در هر مدل و نحوه تبدیل مدل‌ها به یکدیگر از نگاهی مجزا از فازهای توسعه سرویس مرکب است.



شکل ۳-۷: ارتباط مدل‌های چارچوب پیشنهادی

[۱] Executable Composite Service Model

۴.۳ معرفی چارچوب پیشنهادی

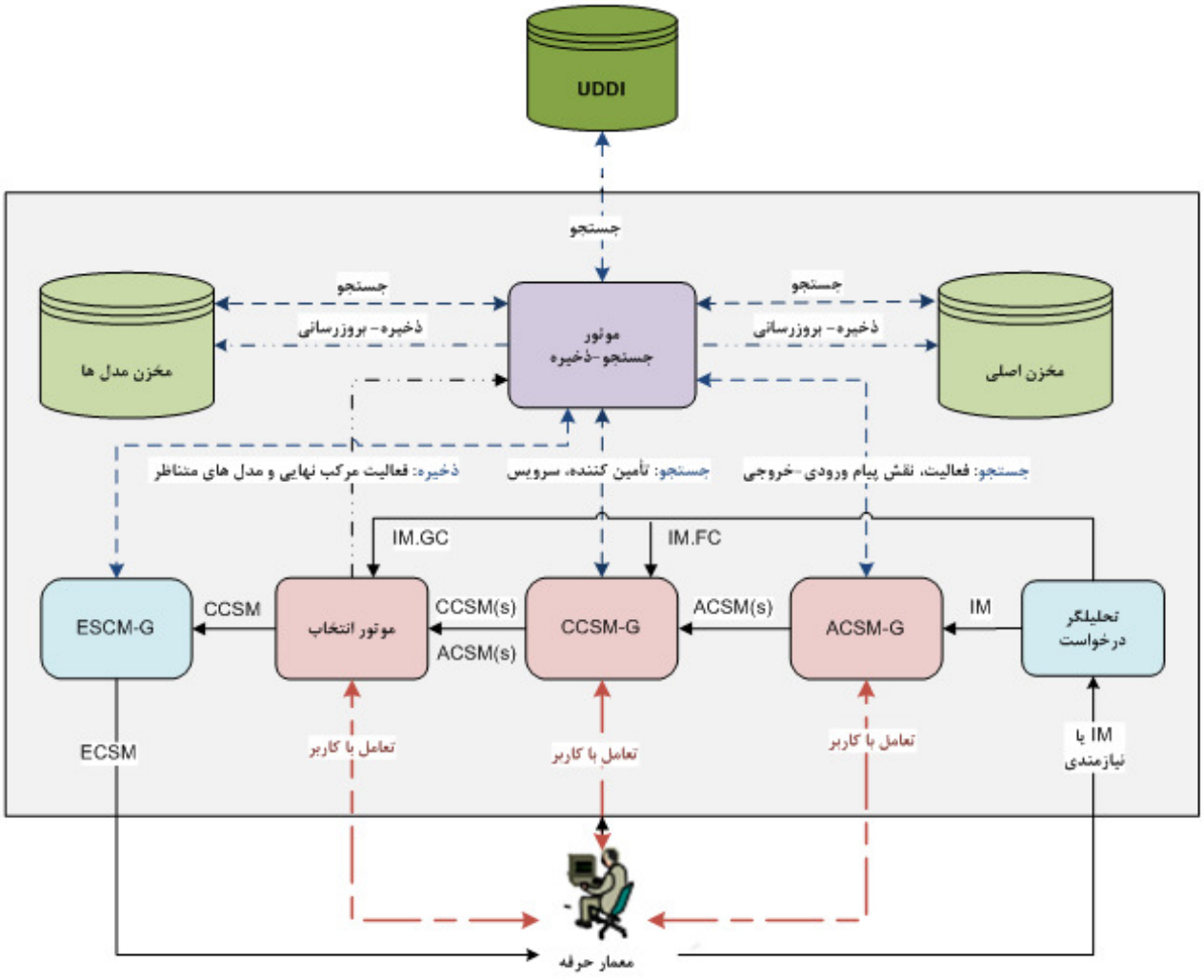
در بخش ۲.۳ مدل متاداده ای و عناصر ترکیب در آن تشریح شد و در بخش ۳.۳ مدل‌های مورد استفاده در فرآیند توسعه چارچوب پیشنهادی با جزئیات معرفی گردید. در این بخش به معرفی معماری چارچوب و مؤلفه‌های موجود در این معماری می‌پردازیم.

همان‌طور که در بخش ۱.۱ بیان شد، ترکیب سرویس به طور کلی به دو سناریوی "توسعه ترکیب سرویس" و "مدیریت و اجرای ترکیب سرویس" تقسیم می‌شود[۵۶]. سناریوی اول مربوط به ایجاد فرآیند حرفه با ترکیب سرویس‌های در دسترس است و از زمان درخواست سرویس جدید آغاز و با برگرداندن سرویس مرکب قابل اجرا به پایان می‌رسد. سناریوی دوم، از زمانی که کاربر قصد اجرای یک سرویس مرکب را دارد آغاز می‌شود و با اجرا و نظارت آن به پایان می‌رسد. هدف از ارائه چارچوب MDCHeS، پوشش سناریوی اول یعنی توسعه ترکیب سرویس است. پرداختن به سناریوی مدیریت و اجرای سرویس مرکب در محدوده تمرکز این تحقیق قرار نمی‌گیرد.

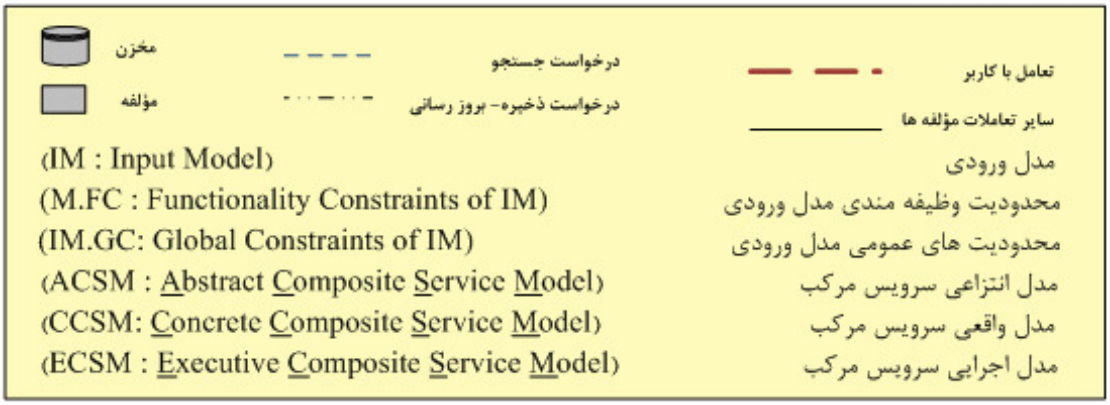
به طور کلی یک چارچوب شامل مجموعه‌ای از ابزارها، فازها، مؤلفه‌ها و ارتباطات بین آن‌هاست. در این بخش چارچوبی مدل‌رانه جهت ترکیب انواع سرویس‌های وب پیشنهاد می‌شود. در این تحقیق مؤلفه‌ها، ارتباط بین آن‌ها و فازهای مربوط به این چارچوب معرفی می‌گردد، البته تکمیل تمامی مؤلفه‌ها و فازهای این چارچوب در محدوده‌ی تمرکز این تحقیق قرار ندارد، اما از آنجا که مدل‌های معرفی شده و روش ترکیب مدل‌رانه پیشنهادی، کلیه فازهای این چارچوب را تحت تأثیر قرار می‌دهد، در این تحقیق ابتدا معماری کل چارچوب و فازهای توسعه ترکیب سرویس در این چارچوب، پس از مطالعات وسیع در این زمینه، پیشنهاد می‌شود، سپس بر روی فاز ساخت سرویس مرکب و الگوریتم‌های مربوط به این فاز تمرکز می‌کنیم. لازم به ذکر است که در این تحقیق با جزئیات کمتری به سایر فازهای چارچوب نیز پرداخته شده و مکانیزم‌های مربوط به هر فاز با مطالعات صورت گرفته استخراج شده و در هر فاز تشریح می‌گردد. تمرکز اصلی در این تحقیق همان‌طور که در فصل ۱ نیز بیان شد، ارائه روشی جهت ترکیب انواع سرویس‌های وب به صورت مدل‌رانه است که این روش وظیفه‌مندی بخشی از چارچوب و فازهای آن را در بر می‌گیرد. پیاده‌سازی ابزار مرتبط با چارچوب پیشنهادی از کارهای آتی این تحقیق به شمار می‌آید. یکی از اهدافی که در ضمن ارائه چارچوب مدنظر قرار داده شده است، امکان پذیری استفاده از آن در سازمان‌هایی است که در زمینه معماری سرویس گرا فعالیت می‌کنند.

۱.۴.۳ معماری چارچوب

شکل ۳-۸ معماری چارچوب پیشنهادی را نشان می‌دهد. در این بخش ابتدا روال کلی چارچوب MDCHeS از مرحله دریافت نیازمندی تا مرحله ارائه مدل اجرایی به کاربر، برای آشنایی مقدماتی با روال چارچوب، به طور خلاصه تشریح می‌شود و در ادامه هر یک از مؤلفه‌های چارچوب معرفی خواهند شد. سپس در بخش۲.۴ از فصل ۴ فرآیند توسعه ترکیب سرویس در چارچوب پیشنهادی با کمک مثال "برنامه ریزی سفر" به تفصیل بیان خواهد شد. فرآیند توسعه در چارچوب MDCHeS از زمانی آغاز می‌شود که کاربر سرویس درخواستی خود را بر اساس وظیفه‌مندی‌های مورد نظر خود، ورودی-خروجی‌های پیشنهادی هر وظیفه‌مندی و جریان داده‌ای و کنترلی مابین این وظیفه‌مندی‌ها با قالب تعیین شده چارچوب بیان می‌کند، سپس درخواست دریافتی به فرم IM مورد استفاده در چارچوب تبدیل می‌گردد. ابتدا لازم است وظیفه‌مندی‌های درخواستی کاربر، با فعالیت، نقش و پیام‌های ورودی-خروجی متناسب در چارچوب جایگزین شوند. در صورتی که فعالیت مرتبط با وظیفه‌مندی مورد نظر یافت نشود، روش ارائه شده با توجه به پارامترهای خروجی وظیفه‌مندی مورد نظر، به صورت پایین به بالا با استفاده از فعالیت‌های موجود در *مخزن اصلی* برای ساخت فعالیتی که پاسخگوی این وظیفه‌مندی باشد، تلاش می‌کند. از آنجا که ممکن است به ازای هر وظیفه‌مندی و مجموعه پارامترهای ورودی-خروجی آن، بیش از یک زوج فعالیت-پیام‌های ورودی-خروجی بدست آید، لذا ممکن است بیش از یک ACSM متناظر با IM حاصل شود. در این مرحله برای تأیید مدل بدست آمده نیاز به تعامل با کاربر وجود دارد. در ادامه به منظور تبدیل ACSM به CCSM نیاز به یافتن عناصر تأمین‌کننده و سرویس متناظر با فعالیت، نقش و پیام ورودی-خروجی است. از آنجا که در این مرحله نیز ممکن است بیش از یک CCSM متناظر بدست آید، مانند مرحله قبل نیاز به تعامل با کاربر وجود دارد. همچنین برخی ترجیح‌ها و محدودیت‌های کاربر ممکن است در این انتخاب تأثیر گذار باشند. در مرحله بعدی لازم است بر اساس نیازمندی‌های غیر عملکردی و ترجیحات مربوط به سرویس درخواستی کاربر، CCSM نهایی جهت تبدیل به ECSM انتخاب گردد. در این مرحله نیز در صورت تمایل کاربر امکان تعامل و تأیید وجود دارد. در ادامه ضمن تولید ECSM تمامی مدل‌های مورد استفاده در روال ترکیب در مخزن مدل‌ها برای استفاده مجدد در سرویس‌های درخواستی آتی ذخیره می‌گردد و همچنین فعالیت متناظر با سرویس مرکب خروجی و عناصر وابسته به آن به مخزن اصلی، به عنوان یک فعالیت مرکب که تأمین‌کننده آن چارچوب MDCHeS است، افزوده می‌شود.



علائم اختصاری



شکل ۳-۸: معماری چارچوب پیشنهادی

۲.۴.۳ عناصر معماری چارچوب

در شکل ۳-۸ مؤلفه‌های چارچوب، نحوه ارتباط آن‌ها و ورودی-خروجی هر مؤلفه نشان داده شده است. در این بخش وظیفه‌مندی مؤلفه‌های چارچوب به طور مختصر بیان می‌گردد و در بخش ۲.۴ از فصل ۴ ضمن تشریح فرآیند توسعه ترکیب سرویس در این چارچوب، مؤلفه‌های درگیر در هر فاز نیز معرفی می‌شوند.

- **مؤلفه تحلیل‌گر درخواست**: این مؤلفه نیازمندی‌های کاربر را به عنوان ورودی دریافت می‌کند و پس از تحلیل و تأیید، در خروجی آن‌ها را به صورت IM به مؤلفه *ACSM-G* می‌دهد. در صورتی که کاربر با الگوی ساده‌ی مربوط به IM آشنا باشد می‌تواند ورودی را به صورت IM وارد کند در این صورت این مؤلفه تنها بررسی صحت IM کاربر را بر عهده خواهد داشت.

همان‌طور که در فاز توصیف نیازمندی‌ها بیان شد، چارچوب MDCHeS نیازمندی‌های را به صورت توصیف کامل که شامل وظیفه‌مندی‌های درخواستی و ویژگی‌های آن‌ها به همراه جریان کنترل و داده بین آن‌ها دریافت می‌کند. در این صورت نیاز است که کاربر هم دانش فنی و هم دانش زمینه[۱] داشته باشد. یکی از وظیفه‌مندی‌هایی که می‌توان به منظور کاربر پسند[۲]تر کردن چارچوب به این مؤلفه اضافه کرد، قابلیت دریافت نیازمندی کاربر با زبان سطح بالا، به طور مثال زبان طبیعی است، سپس با تبدیل آن‌ها به زبانی قابل فهم برای ماشین که مورد استفاده چارچوب است، مدل ورودی را ایجاد کرد. در این صورت نیازی به داشتن دانش فنی برای کاربر نبوده و تنها کافی است کاربر تسلط به دانش زمینه به منظور تشریح نیازمندی داشته باشد. تمرکز بر این وظیفه‌مندی از محدوده تمرکز این تحقیق خارج است اما پس از مطالعات و بررسی در این زمینه راه‌حل پیشنهادی مطرح شده در [۴۷] به عنوان راه‌حلی این وظیفه‌مندی پیشنهاد می‌گردد.

- **مخزن اصلی**: همه عناصر ترکیب معرفی شده در بخش ۱.۲.۳ در *مخزن اصلی* چارچوب بر اساس ساختار مدل متاداده ای که در شکل ۳-۱ نمایش داده شد، ذخیره شده‌اند. همان‌طور که قبلاً نیز گفته شد، لازم است *مخزن اصلی* بر اساس سرویس‌های اعلان شده در UDDI با مکانیزمی (خودکار یا دستی توسط معمار) بر اساس ساختار مدل متاداده ای قبل از شروع فرآیند توسعه سرویس مرکب، تکمیل شود، همچنین از آنجا که ویژگی‌های سرویس‌های اعلان شده در وب قابل تغییر است، نیاز به مکانیزمی برای بروز رسانی دوره‌ای ویژگی‌های سرویس‌های ذخیره شده در مخزن چارچوب است. سرویس‌های مرکبی که در این چارچوب توسعه داده می‌شوند نیز در پایان فرآیند توسعه و پس از اجرای آن به عنوان فعالیتی مرکب، به

---

[۱] Domain Knowledge

[۲] User Friendly

مخزن اصلی چارچوب جهت استفاده مجدد، افزوده خواهند شد. دسترسی به *مخزن اصلی* تنها از طریق مؤلفه *موتور جستجو-ذخیره* امکان پذیر خواهد بود.

- **مخزن مدل‌ها**: این مخزن در برگیرنده مدل‌های مرتبط با سرویس مرکب درخواستی است که شامل همه مدل‌های (IM، ACSM, CCSM, ECSM) شرکت‌کننده در فرایند ایجاد یک سرویس مرکب هستند. از آنجا که سرویس‌های مرکب توسعه داده شده توسط چارچوب، در پایان به عنوان فعالیت مرکب به *مخزن اصلی* چارچوب افزوده می‌شوند، در این صورت لازم است مدل‌های مرتبط با توسعه آن نیز ذخیره گردند تا در صورتی که نیاز به استفاده مجدد از این فعالیت شد، اطلاعات مربوط به محدودیت‌ها و ویژگی‌های کیفی درخواستی کاربر با شرایطی که سرویس طبق آن توسعه داده شده، مقایسه گردد.
- **مؤلفه موتور جستجو ـ ذخیره**: این مؤلفه وظیفه ارتباط با *مخزن اصلی* و *مخزن مدل‌ها* در چارچوب را بر عهده دارد. کلیه تراکنش‌های مورد نیاز سایر مؤلفه‌های چارچوب با هر یک از مخازن چارچوب به وسیله درخواست به این مؤلفه انجام می‌شود. این مؤلفه یکی از مؤلفه‌های اصلی چارچوب MDCHeS محسوب می‌شود. وظیفه‌مندی‌های این مؤلفه عبارتند از:
  - درخواست مؤلفه *ACSM-G* به منظور جستجوی فعالیت، نقش و پیام‌های ورودی-خروجی متناسب با وظیفه‌مندی درخواستی کاربر، توسط این مؤلفه با استفاده از الگوریتم شکل ۴-۴ از فصل ۴، انجام می‌شود. همچنین در صورتی که فعالیت مرکبی برای یک وظیفه‌مندی یافت شود، از طریق ارتباط این مؤلفه با *مخزن مدل‌ها* کلیه مدل‌های مورد نیاز برای فرآیند توسعه استخراج می‌شود.
  - مؤلفه *CCSM-G* درخواست جستجوی تأمین‌کننده و سرویس متناظر با هر فعالیت، نقش و پیام ورودی-خروجی را به این مؤلفه داده و نتیجه را دریافت می‌کند.
  - مؤلفه *موتور انتخاب* نیز ذخیره مدل‌های مرتبط با سرویس مرکب توسعه داده شده را با ارتباط با این مؤلفه در *مخزن مدل‌ها* ذخیره می‌کند.
  - همچنین این مؤلفه به منظور تکمیل و بروز رسانی ویژگی‌های عناصر موجود در *مخزن اصلی* و همچنین افزودن سرویس‌های جدید به *مخزن اصلی* چارچوب، با UDDI در ارتباط است.
- **مؤلفه ACSM-G**[1]: این مؤلفه با ارتباط با مؤلفه *موتور جستجو ذخیره* نیازمندی‌های مربوط به ایجاد مدل‌های ACSM مرتبط با IM که به عنوان ورودی دریافت کرده است را استخراج می‌کند. خروجی این مؤلفه یک یا بیش از یک ACSM

[1] ACSM Generator

است. این مؤلفه برای تأیید فعالیت‌های بدست آمده به ازای هر فعالیت و انتخاب مدل‌های ACSM با کاربر تعامل می‌کند. همچنین در صورتی که به ازای یک وظیفه‌مندی فعالیت مناسبی در جستجو کشف نشود، این مؤلفه وظیفه ساخت فعالیت مرکبی که پاسخ‌گویی این وظیفه‌مندی باشد را بر عهده دارد. تعاملات لازم با مخزن اصلی در این رابطه نیز از طریق مؤلفه *موتور جستجو.ذخیره* انجام می‌شود.

- **مؤلفه CCSM-G**[1]: این مؤلفه وظیفه تبدیل مدل‌های ACSM ورودی به مدل‌های CCSM متناظر را با ارتباط با مؤلفه *موتور جستجو.ذخیره* بر عهده دارد. در این ارتباط عناصر تأمین‌کننده و سرویس متناظر با هر فعالیت بدست می‌آید. همچنین این مؤلفه به عنوان ورودی، محدودیت‌های درخواستی کاربر مرتبط با هر وظیفه‌مندی اولیه را نیز دریافت می‌کند تا درخواست جستجو عناصر نام برده را با توجه به ترجیحات کاربر انجام دهد. این ترجیحات می‌تواند محدود کردن با تأمین کننده خاص و یا سطح خاصی برای ویژگی کیفی سرویس و یا سایر محدودیت‌های پشتیبانی شده در چارچوب باشند. این مؤلفه نیز به منظور تأیید مدل‌های CCSM تولید شده با کاربر تعامل دارد.
- **مؤلفه موتور انتخاب**: این مؤلفه بر اساس ورودی‌های دریافتی و الگوریتم انتخاب تعریف شده در چارچوب با محاسبه شاخص ارزیابی برای هر CCSM بر اساس بخش محدودیت عمومی از مدل ورودی(GC از IM)، میزان ویژگی‌های کیفی در بخش محدودیت عمومی از هر مدل انتزاعی سرویس مرکب (GC از ACSM) و همچنین نوع سرویس‌ها در بخش تعریف سرویس‌ها از هر مدل واقعی سرویس مرکب (SD از CCSM)، شاخص ارزیابی را برای هر CCSM محاسبه کرده و با تکمیل این پارامتر در بخش محدودیت عمومی از هر مدل واقعی سرویس مرکب (GC از CCSM)، مدل‌های CCSM کاندیدا را بر اساس این شاخص مرتب می‌کند. سپس از بین مدل‌های CCSM کاندیدا با تعامل با کاربر، مناسب‌ترین گزینه برای پاسخ‌گویی به سرویس درخواستی کاربر را انتخاب و به عنوان خروجی به مؤلفه *ECSM-G* تحویل می‌دهد. این مؤلفه در پایان مرحله انتخاب کلیه مدل‌های مرتبط با مدل انتخابی شامل IM، ACSM، CCSM برای پاسخگویی به این سرویس درخواستی را به منظور استفاده مجدد از طریق مؤلفه *موتور جستجو.ذخیره* در *مخزن مدل‌ها* ذخیره می‌کند.

---

[1] CCSM Generator

- **مؤلفه ECSM-G[1]**: این مؤلفه CCSM ورودی را به زبان BPEL for REST که زبانی قابل اجرا برای سرویس مرکب درخواستی است تبدیل کرده و به منظور اجرا و مدیریت سرویس مرکب به کاربر تحویل می‌دهد. خروجی این مؤلفه به عنوان خروجی نهایی فرآیند توسعه ترکیب سرویس مرکب در چارچوب MDCHeS است.

۵.۳ جمع بندی مطالب فصل

در این فصل با مبانی روش پیشنهادی که شامل مدل‌های مورد استفاده و معماری چارچوب است، آشنا شدیم. در ابتدا مدل متاداده ای که عناصر ترکیب نحوه ارتباط آن‌ها با یکدیگر را تعیین می‌کرد تشریح شد و مشخص شد که شمای مخزن اصلی چارچوب بر اساس این مدل است، سپس چهار مدلی که در فرآیند توسعه ترکیب سرویس که در بخش ۲.۴ تشریح خواهد شد، استفاده می‌شوند و بخش‌های اصلی هر کدام به تفصیل با بررسی مثال "برنامه ریزی سفر" بیان شد. این مدل‌ها عبارتند از مدل ورودی (IM)، مدل انتزاعی سرویس مرکب (ACSM)، مدل واقعی سرویس مرکب (CCSM) و مدل اجرایی سرویس مرکب (ECSM).

در ادامه فصل پس از آشنایی با مدل‌ها و عناصر ترکیب، معماری چارچوب MDCHeS به همراه مؤلفه‌ها و وظیفه‌مندی هر یک از آن‌ها معرفی گردید. در فصل آینده با فرآیند توسعه ترکیب سرویس در چارچوب MDCHeS و الگوریتم‌های مورد استفاده در هر فاز از این فرآیند آشنا می‌شویم.

[1] ECSM Generator

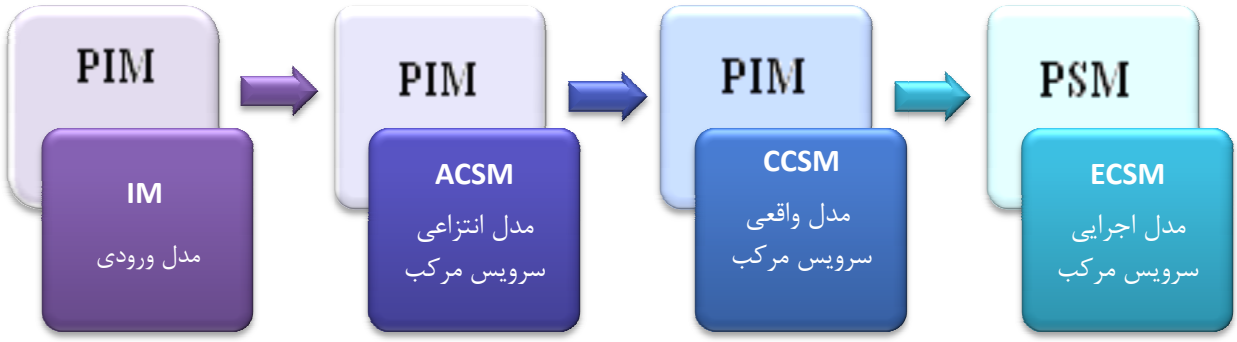
۴ فصل چهارم – فرآیند ترکیب سرویس و الگوریتم‌های مورد استفاده

۱.۴ مقدمه

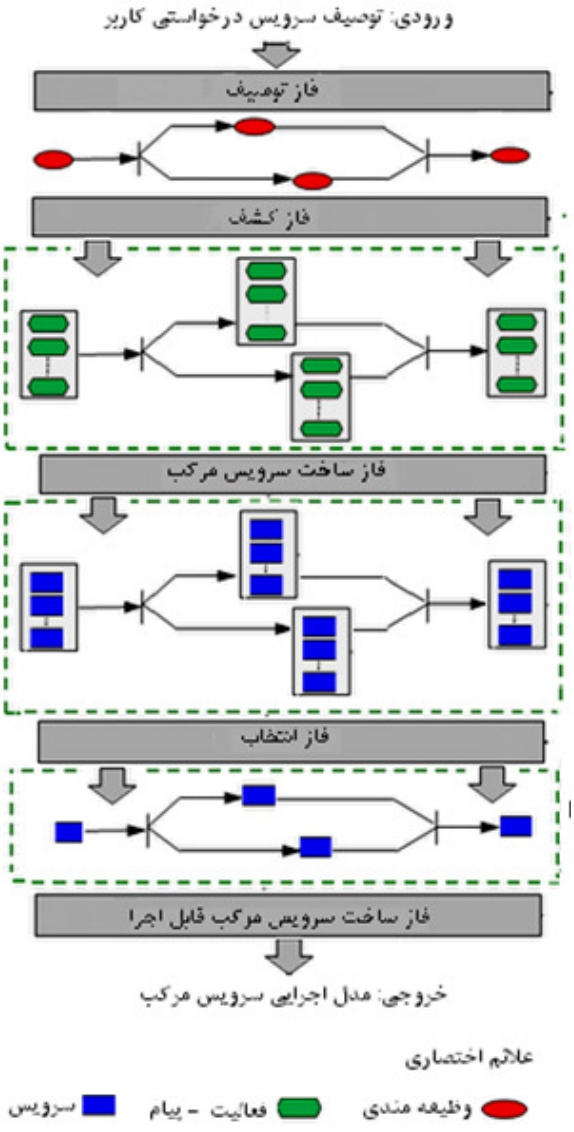
در فصل۳ با مفاهیم مورد استفاده در روش پیشنهادی شامل مدل متاداده ای، مدل‌های معرفی شده و معماری چارچوبی که در بستر آن توسعه ترکیب سرویس‌های وب انجام می‌شود، آشنا شدیم. همچنین وظیفه‌مندی مؤلفه‌های چارچوب توصیف شد. همان‌طور که در فصل ۳ با استفاده از سناریو " برنامه ریزی سفر" سعی بر تشریح بهتر مدل‌های پیشنهادی گردید، در این فصل نیز فرآیند توسعه ترکیب سرویس بر اساس دیدگاهی فازبندی شده به همراه معرفی هر فاز و الگوریتم‌های مورد استفاده با استفاده از این سناریو، تشریح می‌گردد.

۲.۴ فرآیند توسعه ترکیب سرویس

در این بخش، روند توسعه ترکیب سرویس، با استفاده از مدل‌های معرفی شده در بخش ۳.۳، طی فرآیندی فاز بندی شده تشریح می‌شود. لازم به ذکر است که به طور کلی فرآیندی استاندارد و فازبندی شده برای ترکیب سرویس وجود نداشته و دیدگاه‌های معرفی شده هر یک فازهای خود را ارائه داده‌اند اما در این تحقیق سعی شده است که بر اساس مراحل کلی توسعه یک سرویس مرکب فازهای جامعی ارائه شود که با مراحل استاندارد ترکیب سرویس سازگار هستند. در این فرآیند شروع ترکیب از یک تعریف انتزاعی آغاز و با تکمیل کردن آن به صورت تدریجی برای تولید یک سرویس مرکب قابل اجرای متناسب با درخواست کاربر، ادامه می‌یابد. در شکل ۴-۱ فرآیند توسعه در روش پیشنهادی از نقطه نظر تبدیل مدل‌های مورد استفاده در چارچوب (که در فصل ۳ معرفی شدند) و همچنین مدل‌های مطرح در معماری مدل‌رانه نشان داده شده است. همان‌طور که در فصل ۱ بیان شد، در معماری مدل‌رانه فرآیند توسعه به صورت تبدیل خودکار مدل‌های CIM، PIM و PSM است و این تبدیل ممکن است بین سطوح مختلف انتزاع در یک مدل نیز وجود داشته باشد. همان‌طور که در شکل ۴-۲ نیز نشان داده شده است، فرآیند توسعه ترکیب در چارچوب پیشنهادی نیز با استفاده از ایده معماری مدل‌رانه و با تبدیل از مدل PIM در سطح انتزاع بالا به دو سطح از PIM و سپس به PSM است. در ادامه‌ی این بخش فازهای چارچوب، ورودی-خروجی آن‌ها به همراه تکنیک‌ها و الگوریتم‌های مورد استفاده و مؤلفه‌های درگیر در هر فاز توصیف می‌شوند. در شکل ۴-۲ فازهای توسعه ترکیب سرویس نشان داده شده است.



شکل ۴-۱: فرآیند تبدیل مدل‌ها در روش پیشنهادی



شکل ۴-۲: فازهای توسعه ترکیب سرویس در چارچوب MDCHeS

۱.۲.۴ پیش فرض‌های توسعه ترکیب در چارچوب پیشنهادی

در این بخش با پیش فرض‌هایی که قبل از شروع توسعه ترکیب سرویس با چارچوب پیشنهادی، لازم است مد نظر قرار گیرد، آشنا می‌شویم.

۱. لازم است قبل از شروع فرآیند توسعه ترکیب سرویس، عناصر ترکیب معرفی شده در بخش ۱.۲.۳، در *مخزن اصلی* چارچوب ذخیره شده باشند. در واقع قبل از فاز اول چارچوب، مرحله اعلان سرویس‌های اتمیک است. در این مرحله لازم است طی یک مکانیزم که خود شامل دو مرحله زیر است، سرویس‌های اتمیک اعلان شده در وب، به مخزن چارچوب افزوده شوند.

   a. تفسیر توصیف سرویس‌های وب

   b. استخراج نیازمندی‌های مورد نیاز به منظور اضافه کردن عناصر ترکیب به مخزن

برای پاسخگویی به این نیازمندی به دو مؤلفه یکی برای تفسیر انواع مختلف توصیفات سرویس‌های وب و دیگری جهت استخراج اطلاعات لازم و افزودن به مخزن بر اساس ساختار معرفی شده، نیاز است. این بخش از چارچوب، فعلاً به صورت دستی توسط کاربر با سطح دانش کافی (به طور نمونه معمار حرفه) انجام می‌گیرد و تکمیل وظیفه‌مندی و مکانیزم‌های مورد نیاز جهت خودکار سازی این مرحله به عنوان کارهای آینده چارچوب MDCHeS مطرح شده است.

۲. مخزن اصلی بر اساس تغییرات در ویژگی‌های سرویس‌های محیط واقعی و یا اضافه و کم شدن سرویس‌ها، توسط معمار یا مکانیزمی خودکار بروز می‌شود.

۳. لازم است کاربری که نیازمندی‌های خود را وارد می‌کند، از قبل با گرامر و الگوی مدل ورودی آشنا بوده و بر اساس آن نیازمندی خود را وارد می‌کند.

۴. لازم است مکانیزمی خودکار برای بروز رسانی فعالیت‌های مرکبی که متناظر با سرویس‌های مرکب ذخیره‌شده در مخزن هستند، وجود داشته باشد که به صورت دوره‌ای با توجه به تغییر در ویژگی سرویس‌های مورد استفاده و عناصر مخزن این عناصر را بروز گرداند. با این فرض، فعالیت‌های مرکب موجود در مخزن همواره بروز و قابل استفاده هستند.

۲.۲.۴ فاز اول: توصیف سرویس درخواستی (تبدیل نیازمندی کاربر به IM)

هدف از این فاز دریافت نیازمندی‌ها و محدودیت‌های مورد نظر کاربر از سرویس درخواستی است.

بر اساس دسته بندی مطرح شده در [۵۷]، دو سطح مختلف برای نحوه دریافت توصیف سرویس درخواستی از کاربر وجود دارد:

- توصیف جزئی[۱]: در این دسته بسته به سطح دانش کاربر از زمینه[۲]، تنها نمای کلی از وظیفه‌مندی‌های موجود در سرویس درخواستی، به سیستم داده می‌شود.
- توصیف کامل[۳]: در این دسته، تمامی ریز وظیفه‌مندی‌های سرویس درخواستی توسط کاربر مشخص می‌شود.

در هر دو دسته بالا نیاز به دانش زمینه و فنی کاربر است. در صورتی که کاربر تنها دانش زمینه داشته باشد و توانایی نمایش وظیفه‌مندی‌های سرویس درخواستی را با فرمت جریان داده و کنترل نداشته باشد، می‌تواند توصیف سرویس درخواستی خود را به زبان طبیعی وارد کند. اگرچه این بحث در حیطه تمرکز این تحقیق نیست، اما با استفاده از ایده [۴۷] می‌توان به این نیاز به خوبی پاسخ گفت، که این کار به عنوان کار آینده برای تکمیل چارچوب، پیشنهاد خواهد شد.

از طرف دیگر بر اساس دسته‌بندی [۵۸]، روش‌های ترکیب سرویس‌های وب بر اساس تکنیک مورد استفاده برای ترکیب، به دو دسته کلی تقسیم می‌شوند: روش‌های مبتنی بر جریان-کار[۴] و روش‌های مبتنی بر هوش مصنوعی. روش اول خود به دو دسته ایستا و پویا تقسیم می‌شود. در دسته‌بندی ایستای روش اول، لازم است درخواست‌کننده سرویس خود فرآیند حرفه درخواستی را که شامل مجموعه‌ای از وظیفه‌مندی‌های و وابستگی داده‌ای بین آن‌هاست را قبل از شروع ترکیب ایجاد نماید. در واقع در این روش‌های این دسته، تنها کشف و انقیاد سرویس‌ها به صورت خودکار انجام می‌گیرد. در حالی که در دسته بندی پویا، ایجاد مدل فرآیند و کشف سرویس‌های مورد نظر هر دو به طور خودکار توسط سیستم انجام می‌شود.

با توجه به توضیحات بیان شده، روش MDCHeS از نظر دریافت وظیفه‌مندی‌ها و جریان داده و کنترل بین آن‌ها در دسته روش های ایستای مبتنی بر جریان کار قرار می‌گیرد، اما از آنجا که در سناریوهایی که وظیفه‌مندی درخواستی به طور مستقیم با سرویس‌های موجود در *مخزن اصلی* پاسخ داده نمی‌شود، قادر به ساخت فعالیت مرکب، برای آن وظیفه‌مندی است، ما بین روش های ایستا و پویا قرار می‌گیرد.

هدف از مطالب بیان شده، تعیین جایگاه چارچوب MDCHeS بر اساس دسته‌بندی‌های موجود مرتبط با فاز توصیف نیازمندی‌ها بود. پس به طور کلی ورودی‌های روش پیشنهادی را می‌تواند به چهار دسته زیر تقسیم کرد:

---

[۱] Partial Specification
[۲] Context
[۳] Full Specification
[۴] Workflow-based

- وظیفه‌مندی‌های تشکیل دهنده سرویس مرکب درخواستی به همراه پارامترهای ورودی-خروجی و دسته‌بندی استاندارد هر وظیفه‌مندی، معادل بخش توصیف وظیفه‌مندی از مدل ورودی.
- محدودیت‌ها و ترجیحات و ویژگی‌های کیفی برای هر وظیفه‌مندی، معادل بخش محدودیت‌های وظیفه‌مندی از مدل ورودی.
- محدودیت‌ها، ترجیحات، ویژگی‌های کیفی و ورودی-خروجی برای سرویس مرکب درخواستی، معادل بخش محدودیت‌های سراسری از مدل ورودی.
- کنترل داده و جریان بین وظیفه‌مندی‌های معرفی شده بر اساس قوانین گراف معرفی شده در شکل ۳-۳، معادل بخش توصیف جریان داده-کنترل از مدل ورودی.

همان‌طور که بیان شد، هر یک از چهار بخش معرفی شده به عنوان قسمت‌های مختلف IM که در بخش ۱.۳.۳ معرفی شدند، قرار خواهند گرفت. خروجی این فاز مدل ورودی (IM) است که به عنوان ورودی وارد فاز دوم می‌شود. در این فاز تنها مؤلفه *تحلیل‌گر درخواست* نشان داده شده در شکل ۳-۸، به منظور تحلیل، تأیید و نهایی کردن درخواست کاربر و تبدیل آن به IM دخالت دارد. لازم به ذکر است که IM مربوط به سناریوی "برنامه ریزی سفر" در شکل ۳-۴ نشان داده شد.

ایده اولیه نحوه دریافت توصیف نیازمندی‌های کاربر برای سرویس درخواستی در چارچوب MDCHeS و همچنین برخی ویژگی‌های الگوی مدل‌های مورد استفاده در چارچوب، از زبان VCL[۱] [۵۹] گرفته شده است. هدف اصلی زبان VCL فراهم آوردن یک زبان ساده‌ی DSL[۲] برای محیط ترکیب سرویس است. این زبان قادر است چیزی که لازم است یک ترکیب انجام دهد به همراه نیازمندی‌های کیفی سرویس از دیدگاه سرویس مرکب درخواستی و یا به ازای هر وظیفه‌مندی شرکت کننده در سرویس درخواستی، را توصیف کند. به علاوه این زبان توانایی بیان محدودیت‌ها بر روی ورودی-خروجی هر وظیفه‌مندی یا ورودی-خروجی کلی را نیز دارد.

۳.۲.۴ فاز دوم: کشف (تبدیل IM به ACSM)

در این فاز IM که به عنوان ورودی وارد شده است با جایگزینی فعالیت، نقش و پیام ورودی- خروجی به جای وظیفه‌مندی‌ها و پارامترهای ورودی-خروجی معرفی شده‌ی کاربر، توسط الگوریتم شکل ۴-۳ به ACSM تبدیل می‌شود.

---

[۱] Vienna Composition Language

[۲] Domain-Specific Language

ACSM یک مدل انتزاعی است که از سرویس‌های واقعی مستقل بوده و دارای مفهوم انتزاعی استانداردی بر مبنای قوانین چارچوب MDCHeS از سرویس مرکب درخواستی است. این تبدیل بر اساس جستجو به ازای هر وظیفه‌مندی با فراخوانی الگوریتم شکل ۴-۳ (خط ۴)، و با درخواست مؤلفه ACSM-G و انجام جستجوی توسط مؤلفه موتور جستجو-ذخیره بر روی مخزن اصلی آغاز می‌شود. در شرایطی که الگوریتم مورد نظر فعالیت متناظر با وظیفه‌مندی درخواستی را کشف نکرد (خط ۷ از الگوریتم شکل ۴-۳)، برای ترکیب فعالیت‌های موجود و پاسخ‌گویی به وظیفه‌مندی درخواستی فراخوانی می‌شود. در پایان الگوریتم در صورتی که با تعامل با کاربر به ازای هر وظیفه‌مندی حداقل یک فعالیت و پیام‌های ورودی-خروجی کشف شده باشد، با تبدیل بخش‌های مختلف IM به بخش‌های متناظر در ACSM، این فاز با موفقیت نتیجه نهایی را که یک یا بیش از یک ACSM است به عنوان خروجی ارائه می‌دهد. لازم به ذکر است عبارات مخفف مورد استفاده در شبه کد الگوریتم‌هایی که در ادامه آورده شده است در جدول ۸-۱از پیوست ۱ توصیف شده است.

GenerateACSM ()

Input: IM**;**
Output: S_{ACCSM} as a Set of ACSMs related to given IM;

1: $S_{Approved} = \{\}$
2: $S_{NotApproved} = \{F_i | F_i \in IM\}$
3: **For** *each Fi* **in** $S_{NotApproved}$
4: S_{Fi} = ***SearchActivity*** (F_i)
//The main loop for making sure there is at least one <$Activity_i$> for each functionality
5: **While** ($S_{NotApproved}$ is not Empty)
//Trying to fulfill functionalities which there is no single activity for them
6: **For** *each* S_{Fi} *which is empty* **do**
//Compose some activities as a new ACSM instance to fulfill the requested functionality
7: S_{Fi} = ***ComposeActivities*** (F_i)
8: **If** S_{Fi} *is NULL* **then** //If composition fails
9: interactionResult = ***InteractWithUser*** ();
10: **Switch** (interactionResult)
11: **Case** *"Fi decomposed":*
//User may decompose the Fi into some more fine-grain Functionalities
// to increase the success probability in discovery process
12: $F_{decomposed}$ = *all new sub Functionalities for Fi with Related IO(i) and CF*
//update $S_{NotApproved}$

13: **Remove** F_i *from* $S_{NotApproved}$

14: **Add** *each* $F_j \in F_{decomposed\ to}\ S_{NotApproved}$

//search new functionalities

15: **For** *each* $F_j \in F_{decomposed}$

16: S_{Fj} = ***SearchActivity*** (F_j)

17: **Case** *"Abort": //User may give up the composition process because he cannot //decompose the functionality anymore*

18: ***ShowMessage*** *("Abort");*

19: **For** *each Fi* **in** $S_{NotApproved}$ *// taking the user's approval*

20: $Similarity_i$ = ***ComputeSimilarity*** *(Fi,* S_{Fi}*);*

// Removing some improper <activity$_i$, IO$_j$> from S_{Fi} based on the amounts // of similarity. It can be done both automatically and by user interactions.

21: ***Refine*** (S_{Fi})*;*

22: **If** *(there is at least one* <$Activity_i$, IO_j> in S_{Fi} *)* **then**

23: **Add** *Fi to* $S_{Approved}$

24: **Remove** F_i *from* $S_{NotApproved}$

25: **End While**

26: **For** each F_i **in** $S_{Approved}$ **do**

27: **For** each A_i **in** S_{Fi} **do**

//Fetch related Role with the same functionality for each A_i from the main Repository.

28: ***FetchRole*** *(Ai)*;

//Making all combinations of different <activity$_i$, IO$_j$> ∈ S_{Fi} for each F_i ∈ IM.FD // by initializing ACSM.DCG instances from IM.DCG and then replacing the F_i nodes by //activity$_i$ and related IO and R_i ; // ACSM.AD= for each F_i in IM.FD replace Ai in ACSM.AD and Add R_i; //ACSM.GC= for each GC.IO in IM.GC replace boundary IO of IM.DFG;

29: S_{ACSM} = ***MakeAllCombinations*** *();*

30: **Return** S_{ACSM}*;*

شکل ۴-۳: شبه کد الگوریتم تبدیل IM به ACSM

جستجوی فعالیت بر اساس الگوریتم شکل ۴-۴ در این چارچوب انجام می‌شود. در این الگوریتم بر اساس تطبیق طبقه، نام و پارامترهای ورودی-خروجی جستجو انجام می‌شود. این مکانیزم باعث جلوگیری از نتایج غیر مؤثر[۱] و افزایش دقت نتایج حاصل از تطبیق می‌شود.

از آنجایی که جستجوی معنایی در حیطه تمرکز این تحقیق نیست، از کتابخانه‌ی WordNet [۶۰] و [۶۱] جهت تطبیق کلمات مشابه فعالیت و وظیفه‌مندی و همچنین پارامترهای ورودی-خروجی استفاده می‌شود.

الگوریتم جستجو به این طریق است که ابتدا برای هر وظیفه‌مندی معرفی شده در بخش تعریف مدل ورودی (IM.D)، جستجو بر اساس تطبیق طبقه معرفی شده‌ی آن و طبقه فعالیت‌های موجود در مخزن اصلی انجام شده (خط ۲ از الگوریتم شکل ۴-۴)، سپس از بین عناصر مشترک، تطبیق نام با استفاده از WordNet صورت می‌گیرد (خط ۳ از الگوریتم شکل ۴-۴)، سپس از بین فعالیت‌های انتخاب شده تطبیق پارامترهای ورودی-خروجی وظیفه‌مندی با پارامترهای پیام‌های ورودی-خروجی انجام می‌شود (خط ۷ از الگوریتم شکل ۴-۴). ابتدا تطبیق برای پارامترهای ورودی انجام می‌شود. لازم به ذکر است که نوع پارامتر در این مرحله در روند تطبیق تأثیرگذار است. حاصل این تطبیق به سه حالت زیر تقسیم می‌شود:

- تطبیق دقیق (EM[۲]): در این حالت تمامی پارامترهای مقایسه شده، با یکدیگر منطبق هستند.
- تطبیق جزئی(PM[۳]) : در این حالت یک یا بیش از یک پارامتر، از پارامترهای مقایسه شده، متفاوت هستند. تطبیق حداقل یک پارامتر لازم است.
- عدم تطبیق(NM[۴]) : در این حالت هیچ یک از پارامترهای مقایسه شده، منطبق نشده‌اند.

بعد از این مرحله به ازای هر وظیفه‌مندی مجموعه‌ای شامل < فعالیت؛ نقش؛ پیام ورودی-خروجی؛ میزان تشابه> بدست می‌آید. میزان تشابه[۵] در مرحله تطبیق شاخصی است که با توجه به سه حالت تشریح شده در بالا در تطبیق، <فعالیت؛ نقش؛ پیام ورودی-خروجی> های متناسب با میزان تشابه بیشتر برای جایگزینی انتخاب می‌شوند.

SearchActivity()

Input: Fi, $IM.DF_i$

[۱] Ineffective
[۲] Exact Matching
[۳] Partial Matching
[۴] No Matching
[۵] Similarity

Output: $S_{Activities}$ as a Set of proper Activities related to given Functionality;

1: $S_{Activities} = \{\}$
2: $S_{Activities}$ = ***FindActivity*** $(F_i.Category)$;
3: $S_{functionalityNames}$ = ***FindSimilarWords*** $(F_i.name)$; // Using WordNet for find all similar words
// for this Functionality name;
4: **For** each A_i in $S_{Activities}$ **do**
5: **If** $Ai.name \in \{S_{functionalityNames}\}$ // There is at least one $name_i$ in $S_{functionalityNames}$ which
//is matched with $A_i.name$)
6: **then**
//It remains in $S_{Activities}$
7: $Similarity\ [A_i]$ = ***ComputeIOSimilarity*** $(A_i.IO , F_i.IO)$;
8: **else**
9: **Remove** A_i **From** $S_{Activities}$
10: **For** each A_i in $S_{Activities}$ **do**
11: If $Similarity\ [A_i] < SimilarityThreshold$
12: **Remove** A_i **From** $S_{Activities}$ *OR* ***InteractWithUser*** ()
13: **Return** $S_{Activities}$;

شکل ۴-۴: شبه کد الگوریتم جستجوی فعالیت

تفاوت روش پیشنهادی با اغلب روش‌های کنونی ترکیب سرویس‌ها در این است که در اغلب روش‌ها تنها به عینیت بخشی نیازمندی کاربر با استفاده از سرویس‌های در دسترس می‌پردازند اما در روش پیشنهادی ضمن عینیت بخشی به آن دسته از نیازمندی‌هایی که سرویس متناظر آن‌ها در *مخزن اصلی* موجود است، به آن دسته از نیازمندی‌ها که سرویس متناظر برای آن‌ها یافت نمی‌شود با تلاش برای ترکیب فعالیت‌های موجود برای پاسخگویی به نیاز مطرح شده، پاسخ داده خواهد شد. به طوری که اگر فعالیت منفردی متناظر با وظیفه‌مندی درخواستی یافت نشد، با استفاده از الگوریتم ترکیب فعالیت در شکل ۴-۵ و با توجه به فعالیت‌های موجود در مخزن، فعالیتی مرکب را برای پاسخگویی به وظیفه‌مندی مورد نظر ایجاد می‌کند. این الگوریتم بر اساس راهبرد ترکیب عقب‌گرد[۱] ترکیب را به صورت گراف ترکیب ایجاد می‌کند. در الگوریتم تولید گراف ترکیب در شکل ۴-۶ که توسط الگوریتم ترکیب فعالیت در شکل ۴-۵ فراخوانی می‌شود، ابتدا گرافی با گره آغازین ایجاد می‌شود و خروجی‌های وظیفه‌مندی مورد نظر به عنوان لبه‌های ورودی گره آغازین در نظر گرفته می‌شوند سپس الگوریتم با یک راهبرد ترکیب عقب‌گرد برای تأمین کردن

[۱] Backward

آن برگ‌ها پیش می‌رود به طوری که در پارامترهای خروجی فعالیت‌ها جستجو می‌کند تا فعالیت‌هایی را به گراف اضافه کند که آن برگ‌ها را پاسخ دهند. در صورت تطبیق آن فعالیت به گراف اضافه می‌شود و سپس الگوریتم به دنبال تأمین ورودی‌های آن فعالیت پیش می‌رود. در هر تکرار از الگوریتم بخشی از برگ‌های گراف با خروجی‌های فعالیت‌های کشف شده پاسخ داده می‌شود و بررسی می‌شود که آیا برگ‌های موجود با پارامترهای ورودی وظیفه‌مندی مورد نظر یکسان است، در صورت تساوی، الگوریتم به پایان می‌رسد و گراف حاصل از انتها به ابتدا پیمایش سطحی می‌شود. در صورتی که در هر مرحله بیش از یک فعالیت برای پاسخ‌گویی یافت شود (خط ۱۸ و ۱۹ از الگوریتم شکل ۴-۶)، گرافی دیگر به عنوان گراف کاندیدا با وضعیتی مشابه گراف قبلی تولید می‌شود و الگوریتم برای هر دو گراف فراخوانی می‌گردد. این الگوریتم به صورت بازگشتی نوشته شده است.

ComposeActivity ()

Input: S_{input}, S_{output};
Output: $S_{globalPossibleGraphs}$;

1: $S_{globalPossibleGraphs} = \{\}$
2: $N_0 =$ ***InitilizeNode*** *();*
3: $N_0.Input = S_{output}$;
4: $G_0 =$ ***InitializeGraph*** *();*
5: $G_0.Root = N_0$;
4: $S_{globalPossibleGraphs} =$ ***GenerateCompositionGraph*** (G_0, S_{input});
5: **Return** $S_{globalPossibleGraphs}$;

شکل ۴-۵: شبه کد الگوریتم ترکیب فعالیت

GenerateCompositionGraph()

Input: initialGraph, S_{input};
Output: $S_{localPossibleGraphs}$;

1: $S_{leaf} =$ ***GetLeafs*** *(initialGraph);*
2: $S_{localPossibleGraphs} = \{\}$
3: **For** *each* $leaf_i \in S_{leaf}$ **do**
4: **For** *each* $p_i \in leaf_i.Input$ **do**
5: **If** $(p_i \in S_{input})$

```
6:                      // Input parameter is satisfiable and can be removed from the list
7:                      Remove p_i From S_input
8:                      MarkAsBoundryInput (p_i);
9:                      If(S_input == {})
10:                             return initialGraph;
11:             else
12:                     S_Activity = FindActivityByOutput (p_i);
13:                     If(S_Activity.size == 0)
14:                             Return {};          //Fail
15:                     firstActivity = S_Activity.begin ();
14:                     initialGraph.AddChild (firstActivity, leaf_i, p_i);
16:                     Remove firstActivity from S_Activity
17:                     If(S_Activity.size != 0)                //If there is more activity
18:                             For each activity_i ∈ S_Activity do
19:                                     newGraph = InitiateFromExistingGraph (initialGraph);
20:                                     newGraph.AddChild (activity_i, leaf_i, p_i);
21:                                     S_resultedGraph= GenerateCompositionGraph (newGraph, S_inputl);
22:                                     S_localPossibleGraphs = S_localPossibleGraphs ∪ S_resultedGraph;
23:     S_resultedGraph= GenerateCompositionGraph (initialGraph, S_input);
24:     S_localPossibleGraphs = S_localPossibleGraphs ∪ S_resultedGraph;
25:     Return S_localPossibleGraphs;
```

شکل ۴-۶: شبه کد الگوریتم تولید گراف ترکیب

در مرحله نهایی این فاز، در صورت موفقیت در هر یک از الگوریتم‌های معرفی شده، برای هر وظیفه‌مندی یک مجموعه غیر تهی از <فعالیت؛ نقش؛ پیام ورودی-خروجی؛ میزان تشابه> وجود دارد، که لازم است با تعامل با کاربر تعدادی از گزینه‌های مناسب برای جایگزینی با وظیفه‌مندی تأیید شوند.

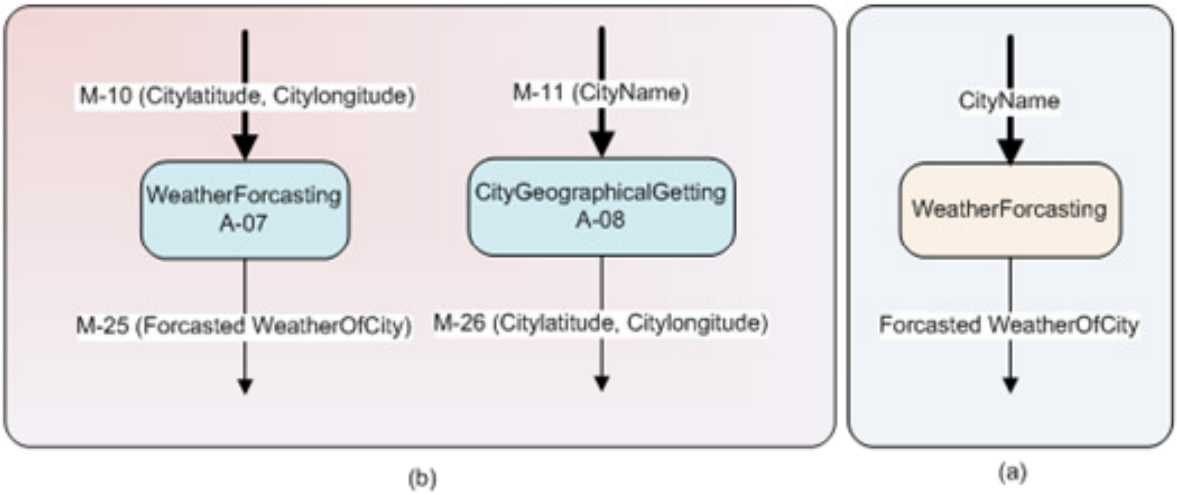
در ادامه لازم است با در نظر گرفتن مجموعه‌ی تأیید شده برای هر وظیفه‌مندی، ACSM های ممکن از این مجموعه بدست آیند. خروجی این فاز مدل‌های ACSM ممکن برای پاسخگویی به سرویس مرکب درخواستی است، از آنجا که ممکن است به ازای هر وظیفه‌مندی بیش از یک چندتایی<فعالیت، نقش و پیام ورودی-خروجی> یافت شود، لذا ممکن است خروجی بیش از یک ACSM باشد. تعداد مدل‌های ACSM که لازم است وارد مرحله بعد شوند را می‌توان با تعامل با کاربر بدست آورد. لازم به ذکر است در صورتی که تعامل با کاربر برای تأیید خروجی امکان‌پذیر نباشد، روش پیشنهادی بر اساس شاخص‌های از پیش تعریف

شده، تعدادی از خروجی‌ها را برای فاز بعدی انتخاب می‌کند. به طور مثال در این فاز مدل‌های ACSM با بیشترین میزان کلی تشابه انتخاب می‌شوند. میزان تشابه یک ACSM، بر اساس میانگین میزان تشابه هر فعالیت جایگزین شده در آن، محاسبه می‌شود.

برای تشریح بهتر این فاز از گراف مدل ورودی برای سناریوی "برنامه ریزی سفر" استفاده کرده و گراف مدل‌های ACSM ممکن برای پاسخگویی به نیاز مطرح شده را بر اساس عناصر مخزن اصلی نمایش می‌دهیم.

شکل ۴-۹ گراف IM سناریوی "برنامه ریزی سفر" را نشان می‌دهد و در شکل ۴-۱۰ دو نمونه از گراف‌های ACSM متناظر با گراف IM برای این مثال نشان داده شده است. الگوی گره‌ها و لبه‌های این گراف‌ها قبلاً در شکل ۳-۳ معرفی شدند. برای وضوح بهتر، کنترل جریان در گراف IM و ACSM نشان داده نشده است. لازم به ذکر است که شمایی از عناصر مخزن اصلی برای سناریوی "برنامه ریزی سفر" در جدول ۷-۱ از پیوست ۱ آورده شده است و شناسه‌های نشان داده شده در شکل ۴-۱۰، بر این اساس هستند. اختلاف گراف‌ها در این شکل به رنگ قرمز (و مایل - خط دار) مشخص شده است. با در نظر گرفتن نیازمندی کاربر برای وظیفه‌مندی پیش‌بینی آب و هوا که با کلمه weather در گراف IM شکل ۴-۹، نشان داده شده است، نحوه عملکرد الگوریتم ترکیب (ComposeActivities) برای وظیفه‌مندی weather تشریح می‌گردد.

در سناریوی برنامه ریزی سفر، کاربر سرویسی را برای پیش بینی آب و هوای شهر مقصد درخواست داده است که به عنوان ورودی نام شهر را گرفته و در خروجی پیش‌بینی آب و هوای آن شهر را نشان می‌دهد. در صورتی که طبق عناصر کلاس پیام در پیوست ۱، تنها فعالیتی برای پیش‌بینی آب و هوا وجود دارد که موقعیت جغرافیایی شهر را به عنوان ورودی می‌گیرد. در بخش (a) از شکل ۴-۷ وظیفه‌مندی درخواستی کاربر برای پیش‌بینی آب و هوا به همراه پارامترهای ورودی (خطوط پر رنگ) و پارامترهای خروجی درخواستی نشان داده شده است و در بخش (b) دو فعالیتی است که در مخزن اصلی چارچوب موجود است به همراه پیام‌های ورودی-خروجی هر کدام نشان داده شده است. هیچ‌یک از این دو فعالیت پاسخ‌گویی وظیفه‌مندی درخواستی نیست. در حقیقت این فعالیت‌ها متناظر با سرویس‌های موجود در مخزن اصلی هستند و در این صورت سرویس مورد نیاز برای پاسخگویی به وظیفه‌مندی درخواستی کاربر در شکل ۴-۷ (a) وجود ندارد.



شکل ۴-۷: نمایی از وظیفه‌مندی پیش بینی آب و هوا و فعالیت‌های متناظر موجود در مخزن اصلی

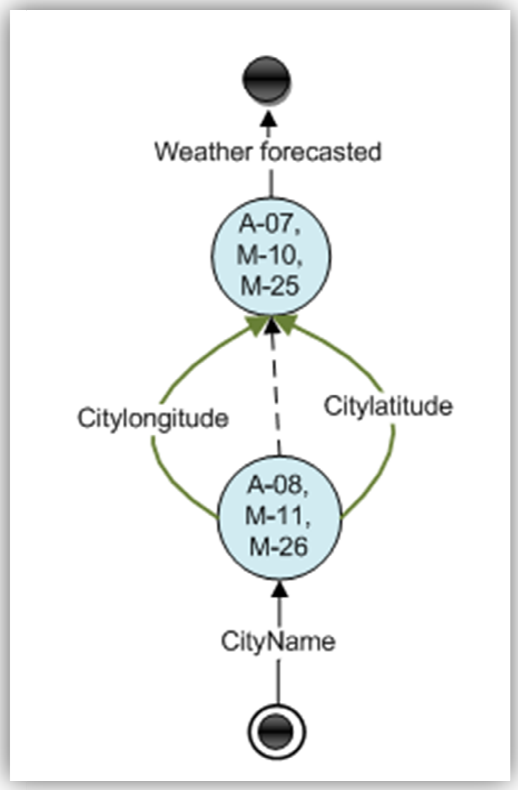
در این مرحله مجموعه‌ی فعالیت‌های متناسب کشف‌شده برای وظیفه‌مندی weather در شکل ۴-۹ تهی است لذا الگوریتم ComposeActivities برای وظیفه‌مندی weather فراخوانی می‌شود (خط ۷ از الگوریتم شکل ۴-۳).

طبق الگوریتم ترکیب فعالیت در شکل ۴-۵، گراف شکل ۴-۸ مرحله به مرحله ایجاد می‌گردد.

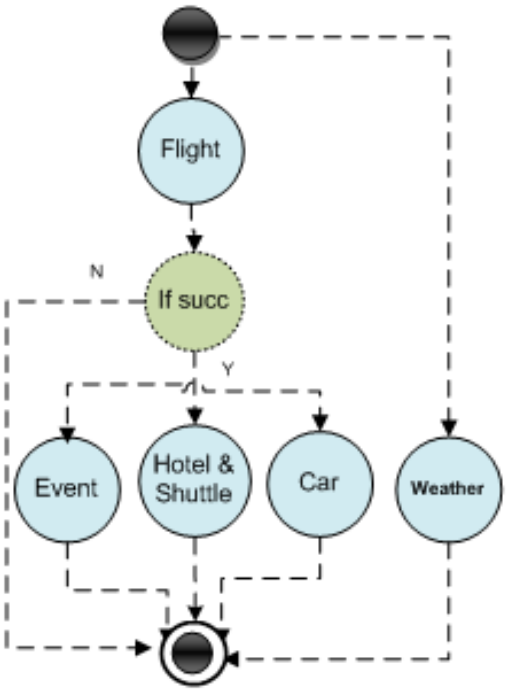
Funcrionality Name = weather

S_{output} = {forcastedWeatherOfCity}

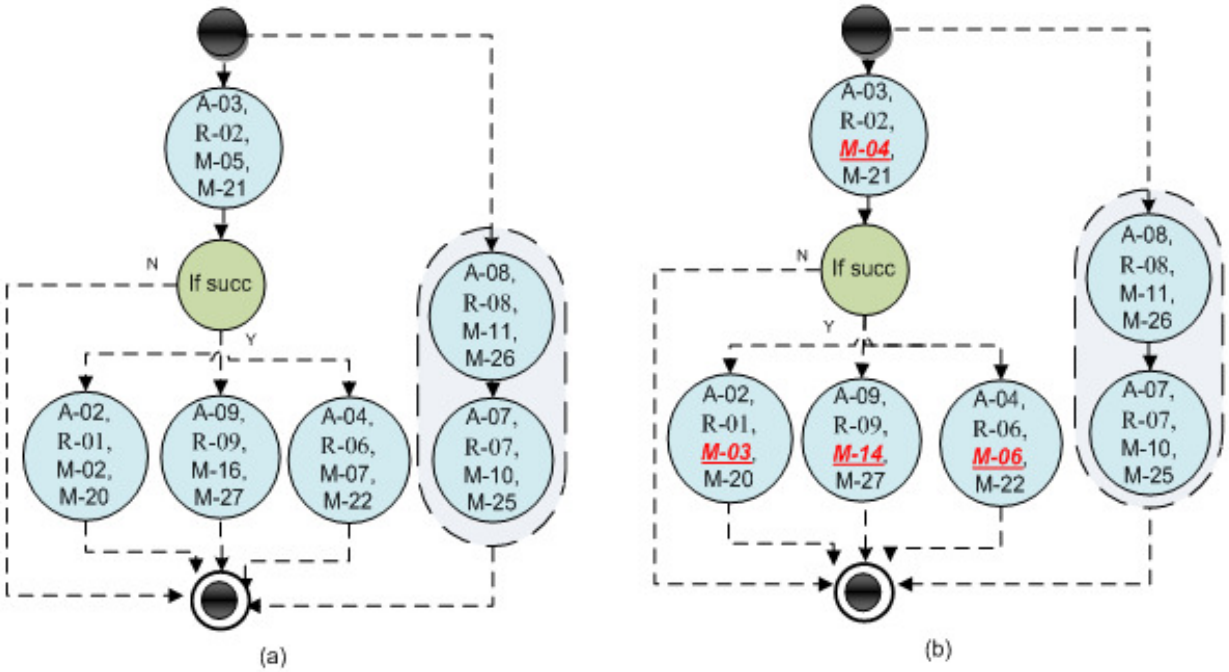
S_{Input} = {cityName}



شکل ۴-۸: گراف ترکیب وظیفه‌مندی "پیش بینی آب و هوا"



شکل ۴-۹: گراف IM برای سناریوی "برنامه ریزی سفر"



شکل ۴-۱۰: دو نمونه از گراف‌های ACSM برای سناریوی "برنامه ریزی سفر"

۱.۳.۲.۴ زمان مصرفی الگوریتم‌های معرفی شده در فاز کشف

در این بخش زمان مصرفی الگوریتم‌هایی که در این بخش معرفی شدند و برای تبدیل مدل ورودی به مدل انتزاعی سرویس مرکب مورد استفاده قرار می‌گرفت، محاسبه می‌گردد. باید توجه کرد که هدف از الگوریتم‌های ارائه شده، بهبود در زمان پاسخگویی الگوریتم‌های موجود در این زمینه نیست و تمامی الگوریتم‌ها جدید هستند که برای روش پیشنهادی و با توجه به ساختار مخزن اصلی و فرآیند توسعه ترکیب در این تحقیق ارائه شده است. ذکر کردن زمان مصرفی الگوریتم‌های ارائه شده برای بیان قابلیت استفاده از آن‌ها در شرایط واقعی بوده است.

زمان مصرفی الگوریتم شکل ۴-۳ با در نظر گرفتن فرضیات زیر، در ادامه محاسبه می‌شود:

- *تعداد وظیفه‌مندی‌های درخواستی* $= n$
- *میانگین تعداد زوج فعالیت-ورودی-خروجی‌های یافت شده‌ی متناظر با هر وظیفه‌مندی* $= m$
- *ماکزیمم تعداد پارامترهای ورودی یا خروجی برای وظیفه‌مندی یا فعالیت* $= x$
- *حداکثر سطح تعیین شده برای درخت ترکیب فعالیت* $= l$
- *میانگین تعداد فعالیت‌های کشف شده از مخزن اصلی برای پاسخ‌گویی به یکی برگ‌های درخت ترکیب با خروجی آن فعالیت‌ها* $= z$

- **زمان مصرفی الگوریتم جستجوی فعالیت:**

$O^{1}(SearchActivity) = n * m * x$

در این الگوریتم x مستقل از پیچیدگی سرویس درخواستی است و تقریباً محدود است. n در سناریوهای پیچیده تعداد بیشتری خواهد داشت و m نیز با توجه به اینکه متناظر است با تعداد زوج فعالیت و ورودی-خروجی‌های متناظر با وظیفه‌مندی درخواستی که در مخزن اصلی چارچوب موجود بوده است، و با توجه به مفهوم انتزاعی پیام و فعالیت، به ازای یک وظیفه‌مندی محدود است.

---

[1] Order

- **زمان مصرفی الگوریتم ترکیب فعالیت = زمان مصرفی الگوریتم تولید گراف ترکیب**

$O(ComposeActivity) = O(GenerateCompositionGraph) = (z \hat{} x) \hat{} l$

الگوریتم ترکیب با فراخوانی الگوریتم تولید گراف ترکیب، گراف ترکیب را بر اساس عناصر فعالیت، پیام موجود در مخزن اصلی چارچوب انجام می‌دهد. الگوریتم تولید گراف ترکیب به صورت بازگشتی تلاش می‌کند در هر مرحله پارامترهای ورودی وظیفه‌مندی درخواستی را با استفاده از فعالیت‌ها و پیام‌های ورودی-خروجی متناسب پاسخ دهد و الگوریتم تا جایی پیش می‌رود که یا به ورودی درخواستی وظیفه‌مندی مورد نظر برسد و یا درخت ترکیب از سطح l (تعیین شده توسط کاربر و مقداری محدود) بیشتر گردد. در حالت اول الگوریتم موفقیت آمیز است و در حالت دوم الگوریتم ناموفق است.

z میانگین تعداد فعالیت‌هایی است که خروجی آن‌ها، یکی از برگ‌های درخت ترکیب را پاسخ‌گو است، z به تعداد عناصر فعالیت در مخزن اصلی بستگی دارد و به پیچیدگی سرویس درخواستی وابسته نیست. x نیز که در بخش قبل معرفی شد و محدود است و از پیچیدگی سرویس درخواستی نیز مستقل است. لذا با بزرگ شدن تعداد وظیفه‌مندی‌ها در سرویس درخواستی، هیچ یک از مقادیر مؤثر در زمان مصرفی الگوریتم رشد نمی‌کنند.

- **زمان مصرفی الگوریتم محاسبه تشابه**

$O(ComputeSimilarity) = x!$

همان‌طور که گفته شد x عددی محدود و مستقل از پیچیدگی سرویس درخواستی است لذا فاکتور یل آن نیز حتی در مسائل پیچیده بزرگ نخواهد بود و تقریباً زیر ۵ است. اما به هر حال الگوریتم محاسبه تشابه بین یک وظیفه‌مندی و پارامترهای ورودی-خروجی آن با فعالیت و پیام ورودی-خروجی متناظر در روش پیشنهادی به صورت غیر بهینه ارائه شده است و بهبود در زمان مصرفی این الگوریتم در محدوده تمرکز این تحقیق نبوده است.

- **زمان مصرفی الگوریتم ساختن ترکیب‌های ممکن**

$O(MakeAllCombinations) = n * m$

اگرچه زمان مصرفی این الگوریتم با افزایش پیچیدگی سرویس درخواستی افزایش می‌یابد، اما به هر حال تعداد وظیفه‌مندی‌ها و فعالیت‌های متناظر آن‌ها محدود است، لذا ایجادACSM با ترکیب تمامی زوج فعالیت و پیام ورودی-خروجی متناظر با وظیفه‌مندی‌های درخواستی نیز، زیاد نیست.

- **زمان مصرفی الگوریتم تبدیل IM به ACSM**

O (GenerateACSM) = n * ((z ^ x) ^ l)

بیشترین زمان مصرفی در این الگوریتم متعلق به الگوریتم ترکیب فعالیت است که همان‌طور کـه قـبلاً بیـان شـد، زمـان مصرفی برابر (z ^ x) ^ l داشته و در صورتی که در بدترین حالت به تعداد تمامی وظیفه‌مندی‌ها این الگوریتم فراخـوانی گردد، لذا زمان مصرفی الگوریتم تبدیل IM به ACSM برابر مقدار بالا خواهد شد.

۴.۲.۴ فاز سوم: ساخت سرویس مرکب (تبدیل ACSM به CCSM)

در این فاز لازم است ACSM با جایگزین شدن تأمین‌کننده، سرویس و ویژگی‌های لازم آن‌ها به جای فعالیت، به CCSM تبدیل شود. این فاز نیز بر اساس ارتبـاط عنصـر نقـش بـا تـأمین‌کننـده و تـأمین‌کننـده بـه سـرویس شـکل مـی‌گیـرد. مؤلفه‌هایCCSM-G، مخزن اصلی و موتور جستجو-ذخیره نشان داده شده در شکل ۳-۸ در این فاز شـرکت مـی‌کننـد. شکل ۴-۱۱ شبه کد الگوریتم تولید CCSM را نشان می‌دهد.

GenerateCCSM ()

Input: S_{ACSM} as a Set of ACSMs, and IM for constraints;
Result: S_{ccsm} as a Set of CCSMs related to given S_{ACSM};

1: $S_{CCSM} = \{\}$; //A Set of concrete models which is initially empty
2: **For** *each* $acsm_i.DCG$ *that* $acsm_i \in S_{ACSM}$ **do**
3: **Create** $ccsm_j$ *as a CCSM instance and initialize it from the* $acsm_i$;
4: **For** *each nodes and related edges* $\in acsm_i.DCG$ **do**
//Finding proper providers and services for this given activity
5: $Services_i$ = ***FindServices*** ($activity_i$, IO_j, $IM.FC_j$);
// Discover proper services based on the predefined relationship between message,
// activity, role, provider and service in the Main Repository
// Updating the $ccsm_j$ by these discovered services.
// The corresponding providers are added implicitly.
6: **Add** $Services_i$ *and* $Provider_i$ *to the* $ccsm_j.SD$ *and* $ccsm_j.DCG$;
7: **Compute** *totsl QoS for each* $ccsm_i$ *based on the QoS of each contained Services and complete* $ccsm_i.GC$;
8: **Add** *type of Services to* $ccsm_i.DS$**;**

8:	**Add** $ccsm_j$ to $S_{CCSM;}$
9:	**Return** $S_{CCSM;}$

شکل ۴-۱۱: شبه کد الگوریتم تبدیل ACSM به CCSM

ورودی‌های این فاز مدل‌های ACSM تولید شده در فاز قبل به همراه IM است. IM به این دلیل به عنوان ورودی مطرح می‌شود که محدودیت‌های مورد نظر کاربر برای هر یک از وظیفه‌مندی‌ها در بخش IM.FC (Functionality Constraints) نظیر تعریف محدودیت بر روی تعیین تأمین‌کننده‌ای مشخص برای یک وظیفه‌مندی و یا حداقل‌هایی برای ویژگی‌های کیفی هر یک از سرویس‌ها و یا غیره در یافتن سرویس متناسب با نیاز درخواستی کاربر تأثیرگذار باشد.

همچنین در این فاز لازم است بخش مربوط به محدودیت‌های سراسری سرویس مرکب درCCSM که شامل ویژگی کیفی سرویس مرکب است محاسبه گردد (خط ۷ از الگوریتم تولید CCSM در شکل ۴-۱۱). این محاسبه بر اساس ویژگی کیفی هر یک از سرویس‌های تشکیل دهنده سرویس مرکب بدست خواهد آمد و در پارامتر ویژگی کیفی سرویس[۱] در بخش CCSM.GC (Global Constraint) از هر CCSM قرار می‌گیرد. این قسمت از CCSM، در فاز انتخاب، یکی از شاخص‌های انتخاب CCSM نهایی است. از آنجا که ممکن است یک فعالیت با بیش از یک سرویس تأمین شود، خروجی این فاز نیز ممکن است بیش از یک CCSM به ازای هر ACSM ورودی باشد. کلیه مدل‌های CCSM به همراه مدل‌های ACSM متناظر وارد فاز بعدی می‌شوند.

به طور کلی یکی از جنبه‌های پویا بودن و یا ایستا بودن روش ترکیب بسته به انقیاد سرویس‌ها در زمان طراحی یا اجرا دارد. از آنجا که روش پیشنهادی در این تحقیق بر روی توسعه ترکیب سرویس تمرکز داشته و وارد بخش مربوط به مدیریت و اجرای ترکیب سرویس نمی‌گردد، لذا انقیاد به سرویس‌ها در زمان طراحی سرویس مرکب صورت می‌گیرد. بر اساس [۵۷] می‌توان دو حالت برای محیط متصور شد، جهان بسته[۲] و جهان باز[۳]. در جهان بسته فرض بر آن است که محیط از زمان طراحی سرویس مرکب تا زمان اجرای آن بدون تغییر باقی خواهد ماند. اما در جهان باز بر عکس حالت اول است و با شرایطی متغیر روبرو هستیم. در حقیقت محیط سرویس‌های وب، محیطی پویا و متغیر است و امکان تغییر ویژگی‌های سرویس‌های اعلان شده توسط تأمین‌کنندگان و همچنین ارائه سرویس‌های جدید وجود دارد، در چنین محیط پویایی به یک روش ترکیب پویای سرویس نیاز است، اما لازمه پشتیبانی از یک روش پویای ترکیب، پوشش اجرا و مدیریت سرویس مرکب است که به عنوان کار آینده در این

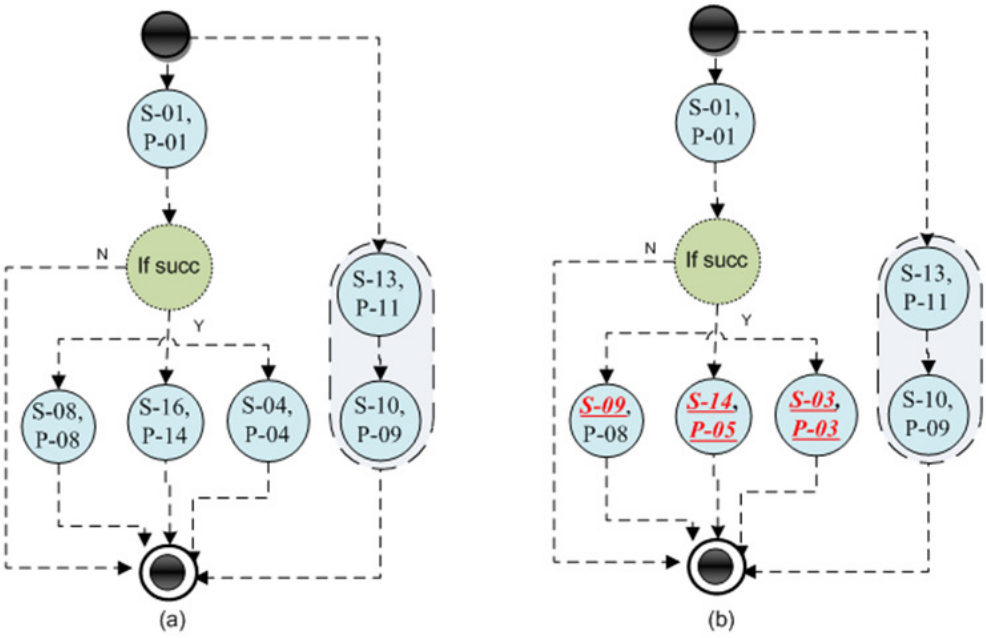
---

[۱] QoS: Quality of Service

[۲] Closed-World

[۳] Open-World

تحقیق نام‌برده شده است. لذا در این تحقیق از روش ترکیب ایستا که ترکیب سرویس را در زمان طراحی انجام می‌دهد بهره می‌گیریم اما روش پیشنهادی با در نظر گرفتن عناصر انتزاعی نظیر فعالیت، پیام و نقش خاصیت پویایی مدل سرویس مرکب را تا مرحله نهایی حفظ کرده و تغییر در ویژگی سرویس‌ها و تأمین‌کنندگان آن‌ها در حین ترکیب بر روی مدل انتزاعی تأثیرگذار نخواهد بود. همچنین با در نظر گرفتن فرض جهان باز، لازم است مکانیزمی جهت بروز رسانی مخزن بر اساس تغییر در ویژگی سرویس‌ها به صورت دوره‌ای در نظر گرفته می‌شود تا با آخرین وضعیت آن‌ها، ترکیب‌های جدید صورت گیرد. همان‌طور که در شکل ۳-۸ نشان داده شده است، این کار را مؤلفه *موتور جستجو-ذخیره* با ارتباط با UUDI بر عهده خواهد داشت. در شکل ۴-۱۲ گراف‌های CCSM متناظر با گراف‌های ACSM نشان داده شده در شکل ۴-۹ نمایش داده شده‌اند. گراف CCSM (a) متناظر با گراف ACSM (a) در شکل ۴-۹ است. این ارتباط برای گراف‌های (b) نیز برقرار است.



شکل ۴-۱۲: دو نمونه از گراف‌های CCSM برای سناریوی" برنامه ریزی سفر"

۱.۴.۲.۴ زمان مصرفی الگوریتم تبدیل ACSM به CCSM در فاز ساخت

زمان مصرفی الگوریتم معرفی شده در شکل ۴-۱۱ با در نظر گرفتن فرضیات زیر عبارت است از:

- *تعداد ACSM های حاصل از الگوریتم تبدیل IM به ACSM* $= a$
- *میانگین تعداد فعالیت‌های موجود در یک ACSM* $= g$

O (GenerateCCSM) = a * g

۵.۲.۴ فاز چهارم: انتخاب (انتخاب CCSM نهایی)

ویژگی‌های کیفی سرویس درخواستی و محدودیت‌های کاربر در مورد سرویس مرکب درخواستی که خروجی فاز اول است (بخش IM.GC از مدل ورودی اولیه) به همراه مدل‌های CCSM تولید شده در فاز قبلی به عنوان ورودی‌های فاز انتخاب در نظر گرفته می‌شوند. همان‌طور که قبلاً اشاره شد، برای هر سرویس در *مخزن اصلی* ویژگی‌های کیفی که توسط تأمین‌کننده هر سرویس تعیین می‌شود، وجود دارد. بر اساس ویژگی کیفی سراسری هر CCSM که در بخش CCSM.GC (Global Constraints) در فاز قبل محاسبه شده است (خط ۷ از الگوریتم تولید CCSM شکل ۴-۱۱) و همچنین نوع سرویس‌های تشکیل دهنده CCSM (RESTful یا مبتنی بر SOAP) که در بخش CCSM.SD (Service Definition) معرفی می‌شود، شاخص ارزیابی CCSM ها محاسبه می‌شود(خط ۱ از الگوریتم انتخاب مناسب‌ترین CCSM شکل ۴-۱۳)، این پارامتر به بخش محدودیت‌های عمومی از هر CCSM (CCSM.GC) که به عنوان ورودی دریافت شده است، اضافه می‌شود.

سپس با مقایسه میزان ویژگی کیفی درخواستی کاربر در IM.GC مقایسه و با تعامل نهایی با کاربر، مناسب‌ترین CCSM برای ورود به فاز بعدی انتخاب می‌شود. در این مرحله انتخاب بدون دخالت کاربر بر اساس دارا بودن بالاترین شاخص ارزیابی نیز امکان پذیر خواهد بود.

ChooseTheBestCCSM()

Input: S_{ccsm} as a Set of CCSMs, IM.FC;
Output: The best CCSM for given S_{ACSM} based on the user's constraints in IM.FC;

1: **For** *each ccsm in* S_{CCSM} **do**
2: *EvaluationFactor*= ***ComputeEvaluationFactor*** *(*S_{CCSM}*, CCSM.SD);*
3: **Update** CCSM.GC based on the computed EvaluationFactor.
4: **Sort** S_{CCSM} based on the EvaluationFactor;
//To the aim of taking user confirmation, interactions with the user can be made at this time
5: **Select** the max**.**
6. **Return** CCSM**;**

شکل ۴-۱۳: شبه کد الگوریتم انتخاب مناسب‌ترین CCSM

در این فاز ذخیره‌سازی مدل‌های مرتبط با نیاز درخواستی کاربر برای استفاده مجدد، ذخیره می‌شوند. این ذخیره‌سازی شامل CCSM نهایی، ACSMمتناظر آن و IM اولیه خواهد بود، همچنین در این فاز سایر مدل‌های CCSM و ACSM متناظر با شاخص ارزیابی قابل قبول به عنوان کاندیدا در مخزن مدل‌ها ذخیره می‌شوند. هدف از این کار بالا رفتن قابلیت رسیدگی به خطا[۱] و افزایش تحمل خطا[۲] برای مرحله اجرای سرویس مرکب است، زیرا در صورت پیش آمدن خطایی در مرحله اجرای سرویس مرکب انتخابی، سایر مدل‌های CCSM متناظر با سرویس‌های مرکب کاندیدا به عنوان گزینه‌های دیگر برای پاسخگویی به نیاز کاربر، در صورتی که سرویس‌های اتمیک موجود در سرویس مرکب انتخابی در زمان اجرا قابل دسترس نباشند، قابل استفاده هستند.

مزیت این کار نسبت به سایر روش‌های اجرای سرویس مرکب در این است که ترجیحات و محدودیت‌های کاربر در شرایطی است که سرویسی در زمان اجرا در دسترس نباشند در انتخاب سایر مدل‌های کاندیدا نیز در نظر گرفته شده است در حالی که در روش‌های کنونی، در صورتی که روش تمهیداتی برای افزایش تحمل خطا داشته باشد، در این شرایط تنها جستجو برای آن سرویس اتمیک خاص بدون توجه به محدودیت‌ها و نیازمندی‌های اولیه کاربر صورت می‌گیرد. علاوه بر این با فرض اجرای موفق سرویس مرکب انتخابی، لازم است فعالیت مرکبی متناظر با آن ایجاد و به همراه سایر عناصر مرتبط با آن بر اساس ساختار مدل متاداده برای استفاده مجدد به *مخزن اصلی* چارچوب افزوده شود. مؤلفه‌های درگیر در این فاز شامل *مخزن اصلی*، *مخزن مدل‌ها*، *موتور جستجو ذخیره* و *موتور انتخاب* هستند.

۶.۲.۴ فاز پنجم: ساخت سرویس مرکب قابل اجرا (تبدیل CCSM به ECSM)

در این فاز لازم است مدل مستقل از فناوری CCSM،که در فاز قبل به عنوان مناسب‌ترین مدل برای نیازمندی درخواستی و ترجیحات کاربر انتخاب شد، بر اساس فناوری انتخابی به مدل وابسته به فناوری ECSM، تبدیل شود.

پس از مطالعات وسیع برای یافتن راهکاری مؤثری به منظور امکان ترکیب انواع سرویس‌های وب در یک سرویس مرکب، ابتدا تمهیداتی برای توصیف و ذخیره انواع سرویس‌های وب در *مخزن اصلی* چارچوب در نظر گرفته شد و مدل متاداده ای مورد استفاده در چارچوب به صورتی طراحی شد که ویژگی‌های مورد نظر برای هر دو نوع سرویس را در برگیرد و در فازهای توسعه نیز امکان استفاده از هر دو نوع سرویس وب در نظر گرفته شد، سپس در فاز انتهایی که نیاز است مدل واقعی سرویس مرکب که

---

[۱] Error Handling

[۲] Fault Tolerance

ممکن است شامل هر دو نوع سرویس وب باشد، به مدلی قابل اجرا از سرویس مرکب تبدیل گـردد و همچنـین امکـان ارائـه هـر دو نوع سرویس وب به عنوان خروجی نهایی چارچوب وجود داشته باشد، لزوم اسـتفاده از روشـی بـود کـه ضـمن اسـتاندارد بـودن، دو قابلیت زیر را داشته باشد:

۱. قابلیت فراخوانی مستقیم هر دو نوع سرویس وب در خلال فرآیند سرویس مرکب.

۲. امکان تبدیل فرآیند اجرایی سرویس مرکب به هر دو نوع سرویس وب (یعنی نتیجه اجرای فرآیند اجرایی هم بتواند سرویس وب RESTful و هم سرویس وب مبتنی بر SOAP باشد).

از بین راه حل‌های ممکن، تنها گزینه‌ای که هر دو قابلیت نام بـرده را بـه صـورت مناسـبی پشـتیبانی مـی‌کنـد، روش BPEL for REST[۲۲]، [۴۴] است. این روش که توسط آقای Cesare Pautasso در سال ۲۰۰۹ مطرح شده است و ویژگی‌های آن به طـور کامل در فصل ۲ بخش ۱۷.۱.۳.۲ تشریح گردیده است، در واقع توسعه‌ای برای زبان BPEL ارائه مـی‌دهـد کـه امکـان پشـتیبانی از قابلیت‌های نام برده را فراهم کند. با توجه به قابلیت روش مورد استفاده برای تولیـد انـواع سـرویس‌هـای وب بـه صـورت خروجـی، تعیین نوع سرویس خروجی از نظر RESTful یا مبتنی بر SOAP بودن در چارچوب پیشنهادی هم می‌تواند از طرف کاربر تعیـین گردد و هم با توجه به ویژگی‌های سرویس مرکب و نوع سرویس‌های درگیر در آن، با توجه به قوانین زیر بدون دخالت کـاربر، تعیـین شود:

- در صورتی که تمامی سرویس‌های موجود در CCSM از نوع سرویس‌های مبتنی بر SOAP باشند، خروجی نهایی می‌تواند با استفاده از BPEL از نوع سرویس SOAP است.
- در صورتی که تمامی سرویس‌های موجود در CCSM از نوع سرویس‌های RESTful باشند، در این صورت خروجی نهایی نیز با استفاده از توسعه BPEL for REST سرویس RESTful خواهد بود.
- در صورتی که CCSM هم شامل سرویس‌های RESTful و هم شامل سرویس‌های مبتنی بر SOAP باشد، از توسعه BPEL for REST بدین منظور استفاده می‌شود. در این حالت خروجی هر دو نوع سرویس می‌تواند باشد.

مؤلفه‌های درگیر در این فاز شامل *ECSM-G*، *مخزن اصلی* و *موتور جستجو ذخیره* است. لازم به ذکر است که الگوریتم مربـوط بـه تبدیل CCSM به ECSM در محدوده‌ی تمرکز این تحقیق نبوده است و به عنوان کارهای آتی این تحقیق به شمار می‌آید.

۳.۴ جمع‌بندی مطالب فصل

در فصل ۳ مبانی و مفاهیم مورد استفاده در روش پیشنهادی شامل مدل‌ها توصیف شد و معماری چارچوبی که روش پیشنهادی در بستر آن ترکیب سرویس را انجام می‌دهد به همراه معرفی وظیفه‌مندی مؤلفه‌های درگیر در آن بیان گردید. در ادامه معرفی روش پیشنهادی در فصل۴، به توصیف فرآیند توسعه ترکیب سرویس وب، به همراه الگوریتم‌های مورد استفاده در هر فاز پرداختیم. بیان شد که فرآیند توسعه ترکیب سرویس در چارچوب MDCHeS طبق ایده معماری مدل‌رانه از فازهایی برای تبدیل مدل‌های معرفی شده به یکدیگر تشکیل شده است؛ این فرآیند شامل پنج فاز است که عبارتند از:

۱. توصیف سرویس مرکب درخواستی (تبدیل نیازمندی به مدل ورودی)
۲. کشف فعالیت متناظر با وظیفه‌مندی درخواستی (تبدیل مدل ورودی به مدل انتزاعی سرویس مرکب)
۳. ساخت سرویس مرکب واقعی (تبدیل مدل انتزاعی سرویس مرکب به مدل واقعی سرویس مرکب)
۴. انتخاب سرویس مرکب واقعی نهایی (انتخاب مدل واقعی سرویس مرکب نهایی)
۵. ساخت سرویس مرکب قابل اجرا (تبدیل مدل واقعی سرویس مرکب به مدل اجرایی سرویس مرکب)

۵ فصل پنجم –ارزیابی روش پیشنهادی

۱.۵ مقدمه

در فصل‌های قبلی روش و چارچوب MDCHeS و فازهای توسعه ترکیب سرویس در این روش با کمک سناریوی "برنامه ریزی سفر" تشریح شد. بیان شد که روش MDCHeS ترکیب سرویس را در پنج فاز انجام می‌دهد که عبارتند از: ۱. توصیف سرویس درخواستی۲. کشف فعالیت متناظر با وظیفه‌مندی درخواستی ۳. ساخت سرویس مرکب ۴. انتخاب سرویس مرکب نهایی و ۵. ساخت سرویس مرکب قابل اجرا. همچنین مدل‌های مورد استفاده در فرآیند توسعه به همراه مدل متاداده ای به منظور معرفی نحوه ارتباط عناصر ترکیب توصیف گردید.

در این فصل تلاش می‌شود تا چارچوب و فرآیند توسعه پیشنهادی مورد ارزیابی قرار گیرد. در ابتدا نحوه‌ی ارزیابی تشریح می‌گردد و سپس با معرفی شاخص‌های ارزیابی در ادامه فصل به ارزیابی روش پیشنهادی می‌پردازیم.

۲.۵ نحوه‌ی ارزیابی روش پیشنهادی

روش پیشنهادی از دو وجه لازم است مورد ارزیابی قرار گیرد، در وجه اول ابتدا لازم است فازهای فرآیند توسعه در چارچوب MDCHeS ارزیابی شود و در وجه دوم می‌بایست کلیت فرآیند توسعه و قابلیت‌های چارچوب مورد بررسی و ارزیابی قرار گیرند.

از بین فازهای پنج‌گانه معرفی شده برای فرآیند توسعه ترکیب در روش پیشنهادی، همان‌طور که قبلاً نیز گفته شد، فناروی مورد استفاده در فاز آخر این فرآیند، توسعهBPEL for REST [۴۴] است که پیش‌تر درستی آن از طریق محققان و توسعه دهندگانش تأیید گردیده است[۱]. پس در وجه اول، تنها ارزیابی چهار فاز اولیه مد نظر است که برای ارزیابی ابتدا شاخص‌های ارزیابی معرفی می‌شود و طبق آن‌ها ارزیابی با استفاده از داده‌های آزمون مرتبط با سناریوهای کاربردی "برنامه ریزی سفر" و "استخدام نیرو در سازمان " صورت می‌گیرد. در خصوص ارزیابی کلیت فرآیند و چارچوب پیشنهادی، تلاش می‌شود تا با استفاده از نظر خبرگان در خصوص میزان تطابق و ارتباط ویژگی‌های ادعا شده در چارچوب مورد انتظار با چارچوب پیشنهادی و تحلیل نتایج حاصل از این نظرسنجی، به این هدف دست یابیم.

---

[۱] این فناروی در ژرنال "Data & Knowledge Engineering" در سال ۲۰۰۹ توسط آقای Cesare Pautasso به چاپ رسیده است.

در پایان نیز مقایسه‌ی روش پیشنهادی با روش‌های مشابه موجود بر اساس شاخص‌های معرفی شده، صورت می‌گیرد.

۳.۵ شناخت شاخص‌های ارزیابی

شاخص‌های متعددی برای ارزیابی یک روش ترکیب سرویس در منابع بیان شده‌اند که در ادامه شاخص‌هایی که در این تحقیق مد نظر قرار گرفته است، معرفی می‌شود.

۱. **راهبرد ترکیب** [۶۲]

راهبرد ترکیب شامل استفاده از دیدگاه‌هایی است که به ترکیب سرویس کمک می‌کنند. راهبردهای ممکن برای استفاده در ترکیب سرویس عبارتند از: ترکیب مدل‌رانه، ترکیب معنایی، ترکیب صوری[۱]، ترکیب پویا، ترکیب جنبه گرا و ترکیب اعلانی. ممکن است روشی از چند راهبرد به صورت همزمان نیز استفاده کند. پس در نهایت این شاخص معین می‌کند که روش ترکیب، با کدامیک از راهبردهای نام برده ترکیب سرویس را انجام می‌دهد.

۲. **سطح خودکار سازی**[۶۳] و [۶۴]

با گرفتن مجموعه‌ای از سرویس‌های در دسترس و یک توصیف از سرویس درخواستی کاربر، مسأله ترکیب سرویس، بر روی ایجاد سرویس مرکب جدید به صورت تولید توصیفی از چگونگی هماهنگی سرویس‌های در دسترس برای تحقق بخشیدن به نیاز کاربر متمرکز می‌شود. ایجاد چنین توصیفی به سه طریق ممکن خواهد بود: ۱. به صورت خودکار: با استفاده از ابزاری که به طور کامل الگوریتم ترکیب را پیاده‌سازی کند. به طور کلی استفاده از توصیفات معنایی یا هستان‌شناسی، خودکار سازی فرآیند ترکیب سرویس را ممکن می‌سازد. ۲. به صورت نیمه خودکار: در مواردی که کاربر در طی فاز ترکیب با یک ابزار قابل تعامل، انتخاب‌های خود را وارد نماید در این حالت ترکیب به صورت نیمه خودکار انجام می‌گیرد. ۳. به طور کاملاً دستی: ترکیب به طور کامل توسط کاربر انجام شود.

همچنین بسته به هدف روش ترکیب، ممکن نیازمندی‌های مختلفی ممکن است برای شروع فرآیند ترکیب دریافت شود. هدف از این شاخص تعیین میزان خودکار بودن روش ترکیب سرویس است. باید توجه داشت که الزاماً تمام خودکار بودن روش ترکیب

---

[۱] Formal

سرویس نمی‌تواند به عنوان مزیت روش مطرح شود، به طور مثال گرفتن بازخورد از کاربر در طی فرآیند ترکیب منجر به نتـایج بهتـر و نزدیک‌تر به نیازمندی کاربر خواهد شد، اما نیاز است که روش ترکیب ارائـه شـده، راه‌هـایی بـرای خودکـار شـدن فازهـای فرآینـد ترکیب نیز در نظر داشته باشد تا در صورتی که کاربر قادر به تعامل در طی روال ترکیب نبود، امکان انجام ترکیـب بـه طـور خودکـار وجود داشته باشد. در واقع بهتر است تعامل با کاربر تنها از جنبه تأییـد خروجـی‌هـای حاصـل از هـر مرحلـه بـرای بهبـود کیفیـت خروجی فرآیند ترکیب باشد.

۳. منطق ترکیب [۶۴]

این شاخص به نحوه تعامل سرویس‌های شرکت‌کننده در فرآیند ترکیب ارتباط دارد. به بررسی اینکه جریان کنترل و داده در مکانیزم ترکیب چگونه نمایش داده می‌شود. دسته‌بندی این شاخص به صورت فرآیند محور[۱]، قانون محور[۲] و ترکیبی[۳] است. در مکانیزم فرآیند محور، ترکیب از یک تعریف فرآیند استفاده می‌کند که در آن منطق حرفه به صورت مدل فرآیندی که نشان‌دهنده چگونگی تعامل سرویس‌های وب شرکت‌کننده در ترکیب است، بیان می‌شود. در دیدگاه قانون محور، قوانین حرفه به عنوان منطق ترکیب مورد استفاده قرار می‌گیرند. در دیدگاه ترکیبی، منطق حرفه به بخش‌های اصلی با نام فرآیندهای حرفه شکسته می‌شود که به طور مستقل از قوانین حرفه تشکیل شده‌اند. استفاده از منطق مناسب برای ترکیب، در ساده شدن روال ترکیب مؤثر خواهد بود.

۴. تطبیق و انتخاب سرویس [۶۳]

این شاخص به نحوه‌ی کشف و انتخاب سرویس‌هایی که در ترکیب سرویس مورد استفاده قرار می‌گیرد اشـاره دارد. سـرویس‌هـای شرکت‌کننده در ترکیب بر اساس مجموعه‌ای از نیازمندی‌های موجود در درخواست سرویس، انتخـاب مـی‌شـوند. کشـف و تطبیـق ممکن است بر اساس اطلاعات مختلفی نظیر، اهداف سـرویس[۴]، ورودی- خروجـی- پـیش شـرط- تـأثیر (IOPE) و یـا کلمـات کلیدی باشد. در انتهای مرحله کشف سرویس، لیستی از سرویس‌های کشف شده حاصل می‌شود. این مجموعه نیز، خود مـی‌توانـد به وسیله شاخص‌های انتخاب سرویس، فیل‌تر گردد. انتخاب رویکرد مناسب در این مرحله، تأثیر بسزایی در کیفیـت نتـایج خروجـی

---

[۱] Process Driven
[۲] Driven Rule
[۳] Hybrid
[۴] Service Goals

فرآیند ترکیب خواهد داشت. استفاده از رویکردهایی نظیر اهداف سرویس و IOPE تنها در صورت استفاده از توصیفات معنایی در روش ترکیب و سرویس‌های مورد استفاده ممکن خواهد بود. برای کاهش خطا در مرحله کشف، لازم است تمهیدات مناسبی در توصیف سرویس‌ها و الگوریتم تطبیق در نظر گرفته شود.

۵. قابلیت ترکیب سرویس‌های موجود

برای روش‌هایی که توصیف نیازمندی‌ها را به صورت توصیفات کاملی از کاربر دریافت می‌کنند ترکیب سرویس به معنی عینیت بخشیدن[۱] به نیازمندی‌های درخواستی با استفاده از سرویس‌های موجود و در دسترس است. در شرایطی که امکان پاسخ‌گویی به نیازمندی کاربر با سرویس‌های موجود نباشد، برخی از روش‌های ترکیب، ناموفق بودن ترکیب را اعلام می‌کنند زیرا تنها به جنبه تحقق بخشی نیازمندی با سرویس‌های موجود پرداخته‌اند، اما برخی روش‌ها، با استفاده از تکنیک‌هایی برای پاسخ‌گویی به نیاز درخواستی با ترکیب سرویس‌های موجود تلاش می‌کنند، در واقع این روش‌ها ترکیب را نیز علاوه بر تحقق سازی انجام می‌دهند.

۶. زمان ترکیب [۶۳]

به لحظه‌ای که هر دیدگاه ترکیب سرویس را انجام می‌دهد اشاره دارد، دو زمان ممکن عبارتند از : زمان طراحی و زمان اجرا . همچنین یک طرح ترکیب اولیه می‌تواند در زمان طراحی تعریف گردد و در زمان اجرا به صورت پویا تکمیل گردد. اگرچه ترکیب در زمان اجرا به پویایی ترکیب کمک خواهد کرد و در واقع احتمال شکست در اجرای سرویس مرکب کمتر خواهد شد اما پیچیدگی روش ترکیب برای روش‌هایی که ترکیب را در زمان اجرا در نظر می‌گیرند، بیشتر خواهد بود. همچنین یک روش که ترکیب سرویس را در زمان طراحی انجام می‌دهد، می‌تواند با منتقل کردن زمان انقیاد به سرویس‌ها به آخرین مرحله طراحی، به بخشی از پویایی مورد نظر دست یافته و یا با در نظر گرفتن تمهیدات لازم امکان پویایی زمان اجرا را نیز فراهم آورد. به طور مثال با در نظر گرفتن سرویس‌های کاندیدا برای سرویس مرکب نهایی، تحمل خطا در روش ترکیب را افزایش دهد.

۷. پشتیبانی از ترکیب سرویس‌های نا همگن [۶۵]، [۴۲]، [۱۹] و [۱۷]

منظور از ناهمگن بودن سرویس‌ها در این تحقیق، ناهمگن بودن در سطح فناروی و پروتکل‌های مورد استفاده است. بر این اساس، دو نوع سرویس وب مبتنی بر SOAP و RESTful وجود دارد. در نتیجه هدف از ترکیب سرویس‌های ناهمگن، پشتیبانی از ترکیب همزمان این دو نوع سرویس در یک سرویس مرکب است.

---

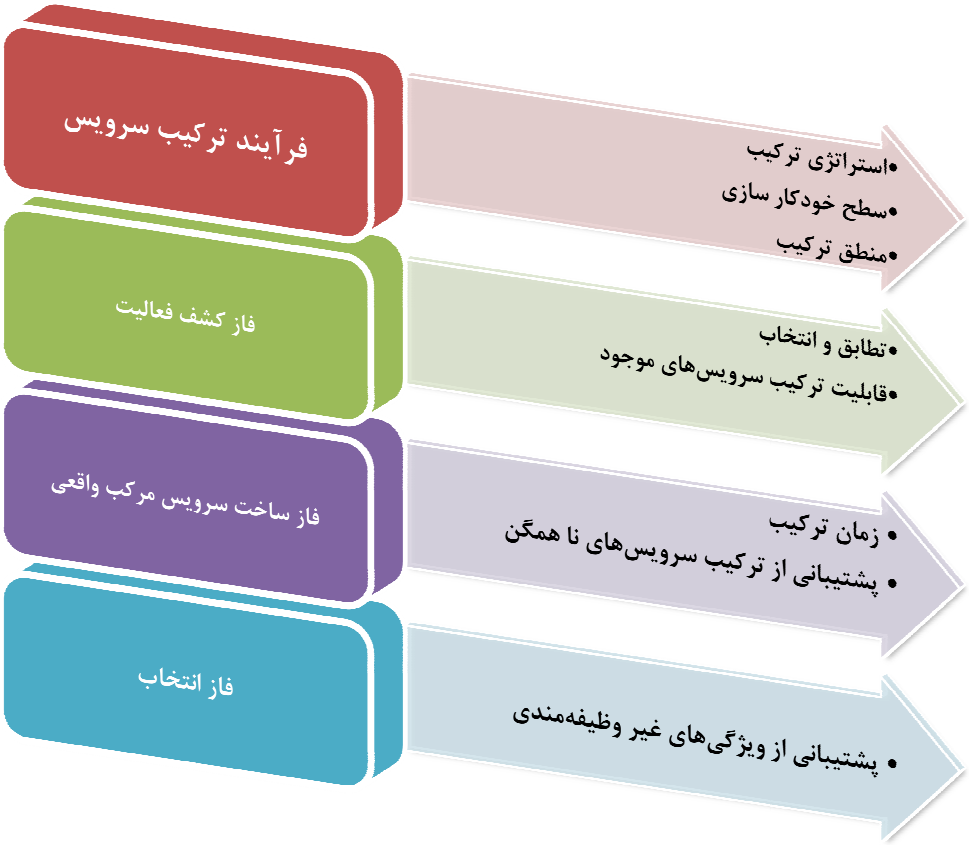
[۱] Realization

این شاخص از آنجا اهمیت دارد که با مطرح شدن سبک جدیدی از معماری سرویس‌های وب با نام REST و رواج استفاده از سرویس‌های RESTful که مبتنی بر قواعد این سبک هستند و مزایایی نظیر سادگی استفاده، سبک‌وزن بودن و راحتی توسعه‌ی آن‌ها، چالشی جدید در ترکیب همزمان این سرویس‌ها مطرح شد. استفاده از این نوع سرویس‌ها در سازمان‌ها به دلیل مزایای مطرح شده، رو به افزایش است و از سوی دیگر سرویس‌های موجود در مخازن سازمان‌ها اغلب سرویس‌های مبتنی بر SOAP هستند و این سرویس‌ها به عنوان دارایی‌های سازمان حائز اهمیت هستند، لذا پشتیبانی از این دو نوع سرویس در روش ترکیب از شاخص‌های مهم برای یک روش ترکیب به شمار می‌آیند. برای پشتیبانی از این ترکیب انواع سرویس‌ها هم لازم است امکان ذخیره انواع سرویس‌ها در مخزن فراهم شود و هم در روال ترکیب امکان استفاده از هر دو نوع سرویس وجود داشته باشد.

۸. پشتیبانی از ویژگی‌های غیر وظیفه‌مندی [۶۳]، [۶۲] و [۶۴]

ویژگی کیفی سرویس، برای بیان ویژگی‌های غیر عملکردی نظیر کیفیت، کارایی، قابلیت در دسترس بودن، هزینه، امنیت و غیره استفاده می‌شود. در محیط کنونی سرویس‌های وب، از آنجا که یک سرویس وب می‌تواند توسط تأمین‌کنندگان بسیاری فراهم شود و به طور پویا از طریق وب فراخوانی شود، اهمیت پشتیبانی و مدنظر قرار دادن این شاخص در روال ترکیب سرویس برای روش‌های پیشنهادی بیشتر شده است. نیازمندی‌های لازم برای پشتیبانی از این شاخص در ابتدا، پشتیبانی زبان توصیف سرویس وب از این ویژگی‌های غیر وظیفه‌مندی سرویس و سپس استفاده از این شاخص در روال ترکیب و انتخاب سرویس‌ها و در نهایت نظارت بر ویژگی‌های غیر وظیفه‌مندی سرویس‌های وب در حین اجرای سرویس است.

شکل ۵-۱ نشان می‌دهد که هر یک از شاخص‌های معرفی شده در این بخش، در چه فازی از فرآیند توسعه‌ی در روش ترکیب پیشنهادی دخیل هستند. همچنین برخی از این شاخص‌ها در کل فرآیند ترکیب سرویس دخالت دارند نظیر سه شاخصی که در ابتدا معرفی شدند و در شکل نیز تحت عنوان شاخص‌های فرآیند ترکیب سرویس نشان داده شده‌اند. این شاخص‌ها در ادامه برای تحلیل نتایج حاصل از انجام سناریوهای کاربردی استفاده می‌شوند. همچنین در بخشی که مربوط به مقایسه روش پیشنهادی، با روش‌های موجود است مقایسه بر اساس این شاخص‌ها صورت می‌گیرد.



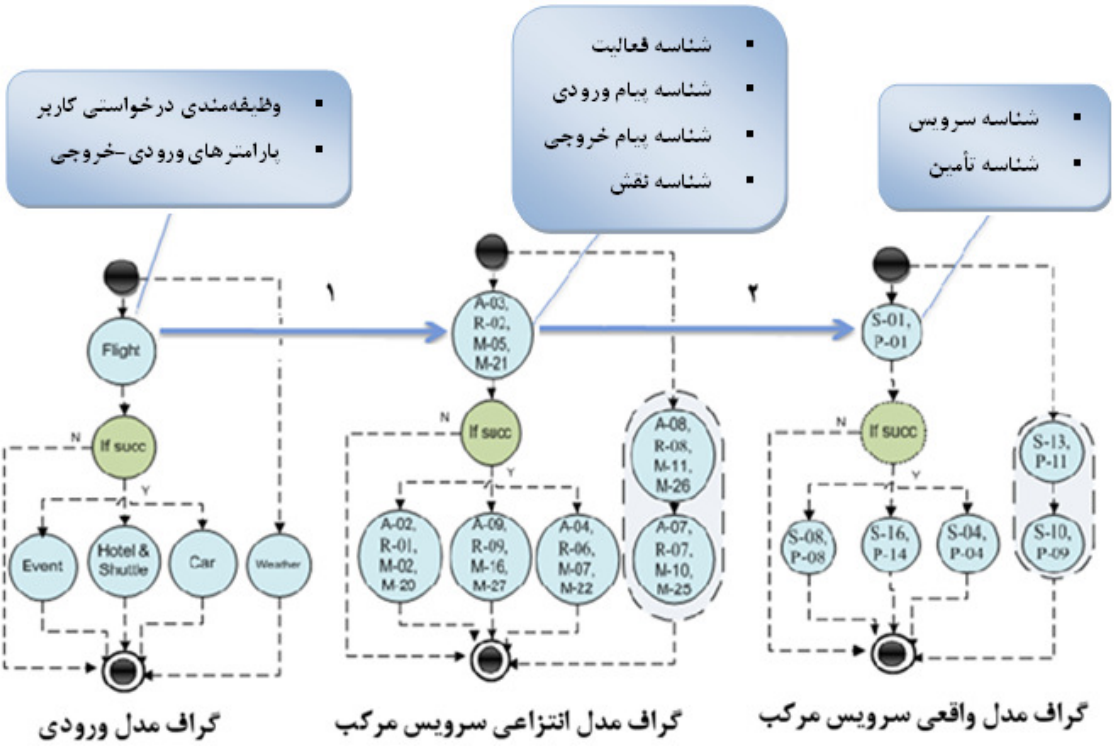
شکل ۵-۱: شاخص‌های ارزیابی به تفکیک فازهای فرآیند ترکیب پیشنهادی

۴.۵ سناریوهای کاربردی

در این بخش برای ارزیابی فرآیند توسعه در چارچوب پیشنهادی و بررسی امکان دست‌یابی به شاخص‌های معرفی شده در بخش ۳.۵، از دو سناریوی کاربردی بهره گرفته و نحوه اجرای سناریوها بر اساس فرآیند توسعه در چارچوب پیشنهادی در ادامه تشریح می‌شود. قبل از انجام سناریوها لازم است در مورد نحوه شبیه‌سازی بخش‌هایی از چارچوب که برای اجرای سناریوهای کاربردی مورد نیاز بود، توضیحاتی بیان شود. یکی از مواردی که لازم است قبل از اجرای فرآیند توسعه ترکیب سرویس برای پاسخگویی به سرویس مرکب درخواستی، فراهم شده باشد، ذخیره سازی عناصر ترکیب بر اساس مدل متاداده ای که در فصل ۳ معرفی گردید، در مخزن اصلی چارچوب است. لذا قبل از انجام سناریوها لازم بود این عناصر در پایگاه داده‌ای با عنوان مخزن اصلی چارچوب برای سناریوهای مورد استفاده جمع‌آوری و ذخیره گردند. بدین منظور، ابتدا سرویس‌های وب متناظر با سناریوها جمع‌آوری شدند (از آنجا که دست‌یابی به توصیف‌های برخی از

سرویس‌ها در یک زمینه خاص فراهم نبود در برخی از آن‌ها، شبیه‌سازی‌هایی نیز صورت گرفته است)، پس از جمع‌آوری اطلاعات مورد نیاز از سرویس‌ها، عناصر واقعی مدل شامل سرویس و تأمین‌کننده در مخزن در جداول مربوط به سرویس و تأمین‌کننده ذخیره و همچنین عناصر انتزاعی مدل، شامل فعالیت، نقش و پیام متناظر با عناصر سرویس و تأمین‌کننده ایجاد و به جداول متناسب اضافه شدند و ارتباط‌های متناظر نیز شکل گرفت. برای پوشش شاخص‌های معرفی شده، لازم بود عناصری در هر سناریو موجود باشد که بیانگر نشان دادن تمامی قابلیت‌های روش پیشنهادی در طی اجرای سناریوی کاربردی باشند. حاصل انجام این مراحل در عناصر موجود در جدول ۷-۱ در پیوست ۱، به تفکیک عناصر مدل متاداده ای و به عنوان شبیه‌سازی مخزن اصلی چارچوب نشان داده شده‌اند. همچنین در شکل ۷-۱ در پیوست ۱، ارتباط عناصر ترکیب در پایگاه داده‌ای که مخزن اصلی در آن شبیه‌سازی شده است، نشان داده شده است. اجرای سناریوها و عناصر استفاده شده در آن‌ها، بر اساس مخزن معرفی شده هستند. در ادامه دو سناریوی به منظور ارزیابی فرآیند ترکیب در روش پیشنهادی را بررسی خواهیم کرد. . ابتدا شرح سناریو بیان می‌گردد، سپس اجرای سناریو بر اساس فرآیند توسعه ترکیب در روش پیشنهادی تشریح و در پایان نیز هدف از اجرای سناریو و میزان تحقق اهداف و شاخص‌هایی که از اجرای این سناریوی مد نظر بوده، تحلیل و بررسی می‌شود.

لازم به ذکر است که سناریوی دوم (برنامه ریزی سفر) برای تشریح فازهای توسعه و مدل‌های مورد استفاده در فصل‌های ۳ و۴ مورد استفاده قرار گرفته و در اینجا تنها جنبه‌ها و اهدافی که از اجرای آن سناریوی مد نظر بوده بررسی می‌شود. شکل ۵-۲، گراف‌های متناظر با این سناریو را به همراه نحوه تبدیل وظیفه‌مندی‌هایی که فعالیت متناظر آن‌ها در مخزن اصلی موجود بوده نشان می‌دهد. شناسه عناصر ترکیب بر اساس جدول ۷-۱ در پیوست ۱ هستند. همچنین مدل‌های مورد استفاده در ترکیب این سناریو نیز به ترتیب در شکل ۳-۴(IM)، شکل ۳-۵ (ACSM) و شکل ۳-۶ (CCSM) نشان داده شده‌اند.



شکل ۵-۲: گراف‌های متناظر با سناریوی " برنامه ریزی سفر"

۱.۴.۵ سناریوی کاربردی " استخدام نیرو در سازمان"

سناریوی استخدام نیرو در یک سازمان یک سناریوی کاربردی-سازمانی است که شرح سناریو از منبع [۱۸] برگرفته شده است.

۱.۱.۴.۵ شرح سناریو

"مدیر بخش منابع انسانی یک سازمان قصد دارد تا برای تسهیل استخدام و جذب نیرو در سازمان، درخواست توسعه یک سرویس مرکب دهد. این سرویس مرکب، ابتدا موقعیت‌هایی از سازمان که نیاز به جذب نیرو دارد را در نظر گرفته و سپس به بررسی رزومه‌های دریافتی سازمان از متقاضیان کار و رزومه‌های برخط [۱] افراد روی اینترنت می‌پردازد و افرادی که دارای شرایط لازم برای موقعیت‌های خالی در سازمان هستند را پیدا می‌کند. همچنین نتایج یافت شده، روی نقشه نمایش داده می‌شوند تا موقعیت جغرافیایی افراد در مقایسه با محل کار آن‌ها به خوبی مشخص باشد. مدیران بخش منابع انسانی در سازمان، با حجم وسیعی از رزومه‌ها روبرو هستند که مستقیماً به سازمان فرستاده می‌شوند. با این حال، تعداد زیادی از کاندیداهای شغلی وجود دارند که رزومه

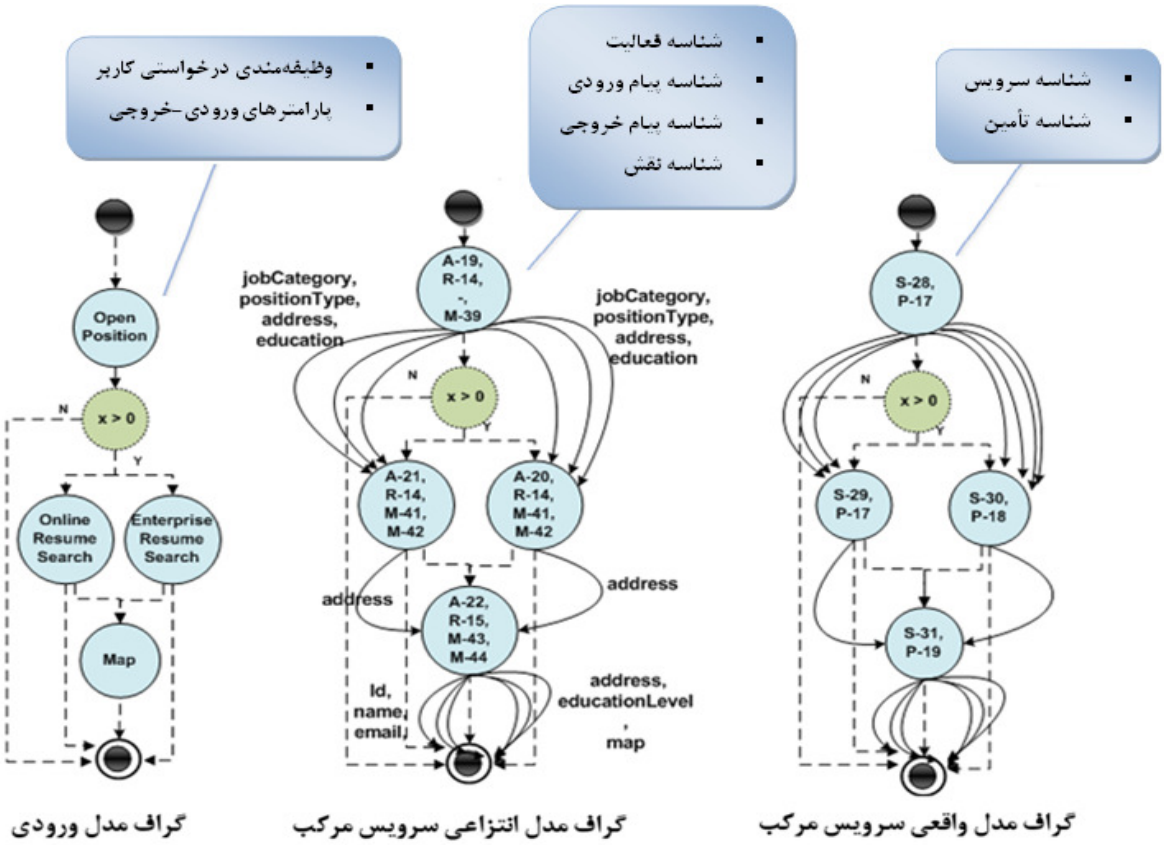
[۱] Online

خود را به صورت برخط و در وب سایت‌های مربوط بـه کاریـابی بـرخط ماننـد Monster.com و یـا شـبکه‌هـای اجتمـاعی ماننـد Linkedin.com درج می‌کنند. ایجاد یک سرویس مرکب که اطلاعات موقعیت‌های خالی در سازمان را با اطلاعات سـرویس‌هـای رزومه خارجی ترکیب کند، به طور قابل توجهی عمل جذب نیرو در سازمان را تسهیل می‌بخشد و ایـن فرصـت را بـرای سـازمان بـه وجود می‌آورد که نگاهی بلادرنگ به بازار افراد با استعداد برای جذب در سازمان داشته باشد."

در ادامه روند توسعه سرویس مرکب برای این سناریو بر اساس روش پیشنهادی تشریح می‌گردد.

۲.۱.۴.۵ اجرای سناریو بر اساس فرآیند توسعه در چارچوب MDCHeS

برای انجام این سناریو و ایجاد سرویس مرکب درخواستی با استفاده از روش MDCHeS، بر اساس فازهای معرفـی شـده در فصـل ۴، سناریو را پیش می‌بریم و مدل‌های خروجی هر فاز بر اساس الگوریتم‌های پیشنهاد شده را نمایش می‌دهـیم. شـکل ۵-۳، گـراف‌هـای متناظر با این سناریو را نشان می‌دهد. هر یک از این گراف‌ها، در بخش توصـیف جریـان داده-کنتـرل از مـدل‌هـای متنـاظر بـا ایـن سناریو قـرار خواهنـد گرفـت. در ایـن شـکل از شناسـه‌هـای عناصـر، بـر اسـاس جـدول ۷-۱ در پیوسـت ۱، اسـتفاده شـده اسـت



شکل ۵-۳:گراف‌های متناظر با سناریوی " استخدام نیرو در سازمان"

مدل ورودی متناظر با این سناریو در شکل ۵-۴ و گراف آن در شکل ۵-۳ با عنوان گراف مدل ورودی، نشان داده شده‌اند که به عنوان خروجی فاز اول که فاز توصیف سرویس درخواستی است، ایجاد خواهند شد. در فاز دوم که کشف فعالیت است، لازم است به ازای تمامی وظیفه‌مندی‌هایی که در بخش توصیف وظیفه‌مندی (Functionaloty Difinition) از مدل ورودی به همراه طبقه استاندارد و همچنین پارامترهای ورودی-خروجی متناظر با هر وظیفه‌مندی در مدل ورودی وجود دارد، فعالیت و پیام ورودی-خروجی متناسب، با استفاده از الگوریتم تبدیل IM به ACSM که در فصل ۴ معرفی شد، کشف شده و در نهایت مدل ورودی به مدل انتزاعی سرویس مرکب تبدیل گردد. هدف از وظیفه‌مندی‌های موجود در نیازمندی درخواستی برای این سرویس مرکب (نشان داده شده در مدل ورودی شکل ۵-۳) عبارتند از:

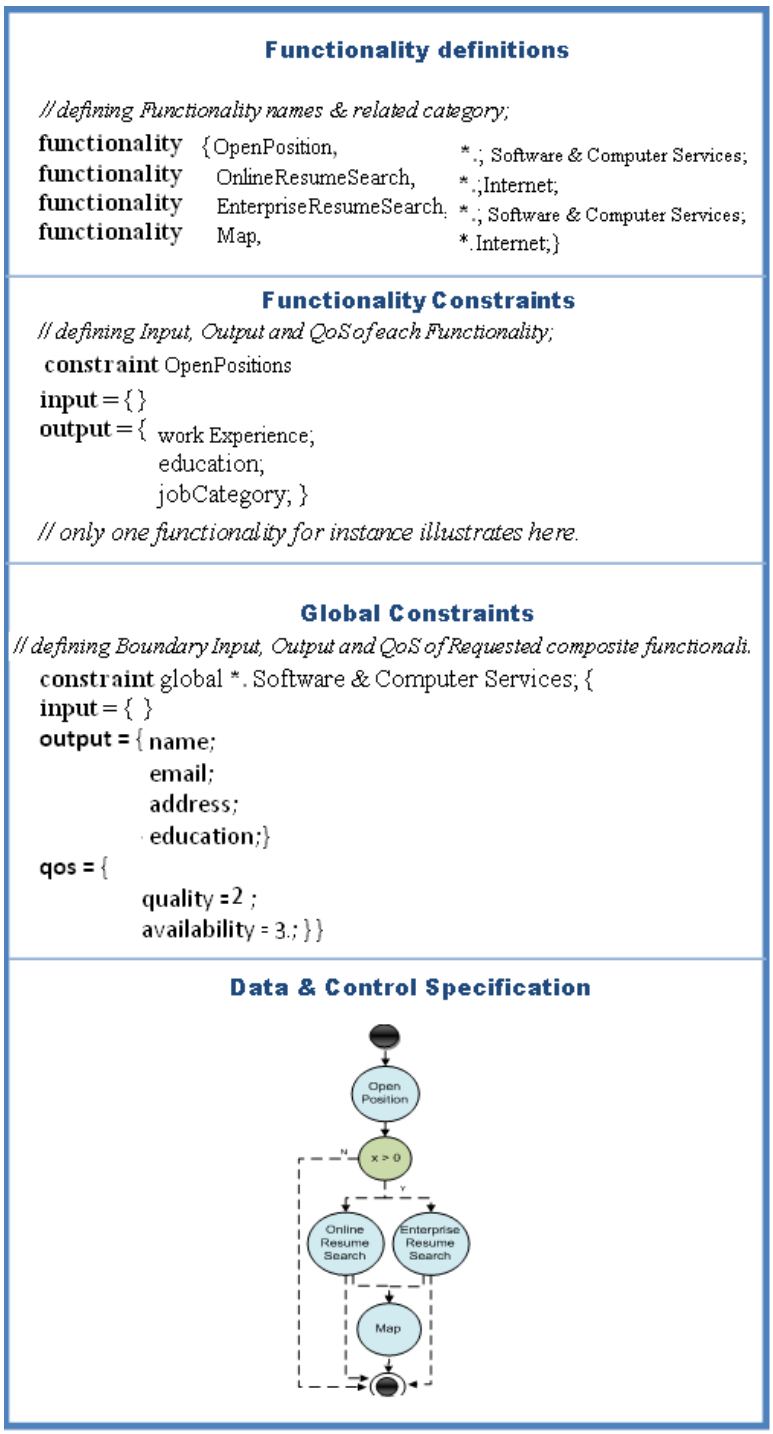
- Open Position : هدف از این وظیفه‌مندی یافتن موقعیت‌هایی است که در سازمان به نیرو نیاز دارند.
- OnlineResumeSearch: هدف از این وظیفه‌مندی جستجوی رزومه‌های مناسب با موقعیت‌های باز سازمان در وب است.
- EnterpriseResumeSearch: هدف از این وظیفه‌مندی جستجوی رزومه‌های مناسب با موقعیت‌های باز سازمان در بین رزومه‌های ارسالی به سازمان است.
- Map: این وظیفه‌مندی با هدف نمایش مکان رزومه‌ی مناسب برای موقعیت باز سازمان در نقشه، درخواست شده است.

در این سناریو به ازای تمامی وظیفه‌مندی‌های نام برده، بر اساس عناصر ترکیب موجود در مخزن اصلی چارچوب، فعالیت متناظر کشف شده و نیازی به اجرای الگوریتم ترکیب فعالیت نیست. شکل ۵-۵ مدل انتزاعی سرویس مرکب متناظر با این سناریو را نشان می‌دهد. برای ساخت سرویس مرکب این نیازمندی، برخی سرویس‌های سازمانی نظیر سرویس "جستجوی موقعیت‌های خالی سازمان[۱]" (با شناسه‌ی فعالیت A-19 و شناسه سرویس S-28 که از نوع سرویس مبتنی بر SOAP است) و برخی سرویس‌های خارج از سازمان نظیر نقشه گوگل[۲] (با شناسه فعالیت A-22 و شناسه سرویس S-31 که از نوع سرویس‌های RESTful است) کشف می‌شوند. در فاز بعدی که فاز ساخت سرویس مرکب واقعی است، بر اساس فعالیت‌ها و پیام‌های ورودی-خروجی متناظر با آن‌ها که در مدل انتزاعی سرویس مرکب مشخص شده بود و همچنین بر اساس محدودیت‌هایی که به صورت

[۱] Find Enterprise Open Position

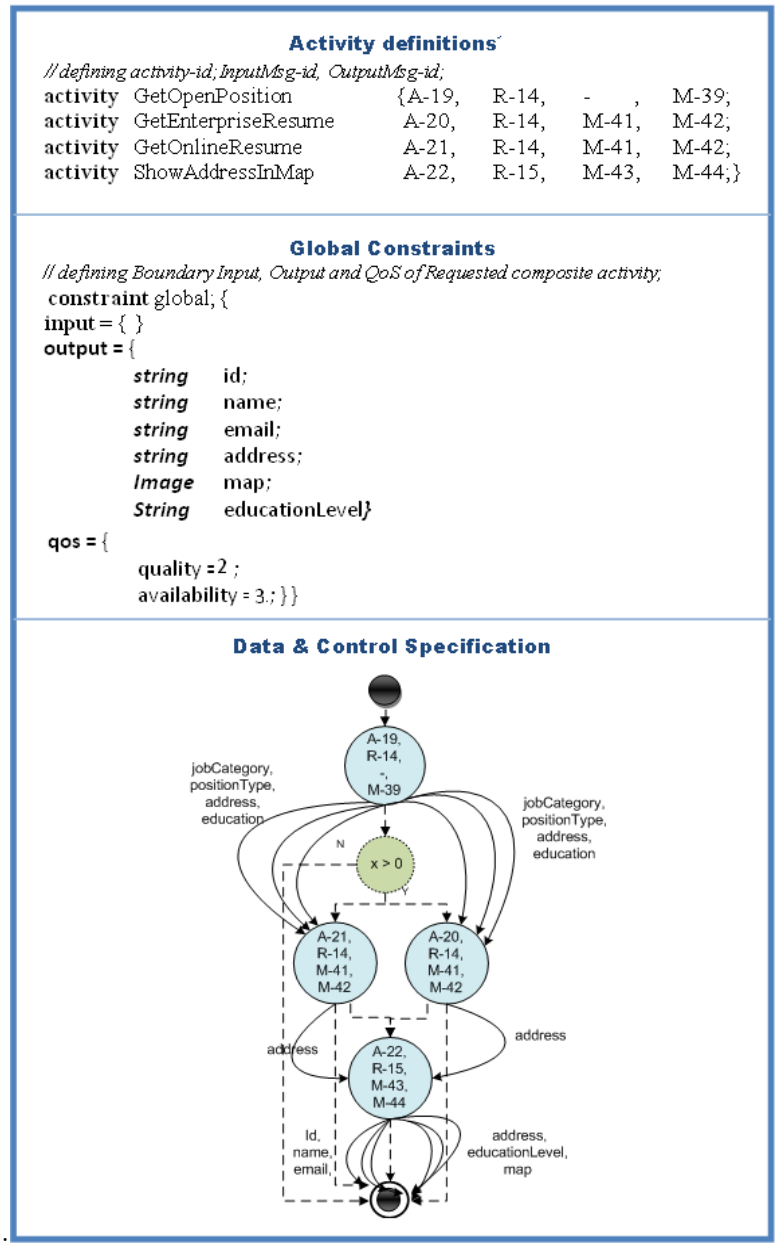
[۲] GoogleMap

عمومی برای سرویس مرکب در نظر گرفته شده است، عناصر تأمین‌کننده و سرویس متناظر از طرق ارتباط‌های موجود بـین عناصـر مخزن اصلی انتخاب می‌شوند. شکل ۵-۶، مدل واقعی سرویس مرکب متناظر را نشان می‌دهد.



شکل ۵-۴: IM در سناریوی" استخدام نیرو در سازمان"

همان‌طور که در بخش محدودیت‌های سراسری (Global Constraints) در مدل ورودی شکل ۵-۴ دیده می‌شود، کاربر برای ویژگی‌های کیفی سرویس مرکب، میزان کیفیت ۲ (متوسط) و میزان در دسترس بودن ۳ (بالا) را درخواست کرده است، لذا لازم است در فاز ساخت از سرویس‌هایی استفاده گردد که در مجموع میزان ویژگی کیفی سرویس مرکب حاصل مساوی با بیشتر از میزان درخواستی کاربر باشد. همان‌طور که در CCSM شکل ۵-۶ مشاهده می‌شود، بر اساس ویژگی کیفی سرویس‌های انتخابی در مجموع کیفیت سرویس مرکب ایجاد شده برابر ۲.۵ (متوسط به بالا) و میزان در دسترس بودن آن ۳ (بالا) است لذا قابل قبول است.



شکل ۵-۵: ACSM در سناریوی" استخدام نیرو در سازمان"

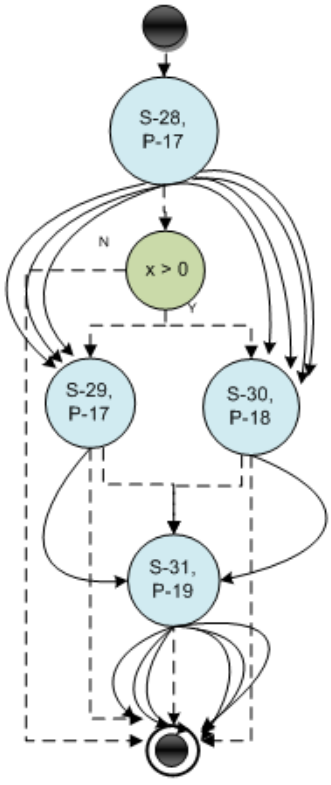
Service definitions

```
// defining Service-id, Role-id, Provider-id and Service-type;
service {S-28,   P-17,   SOAP-based;
service  S-29,   P-17,   SOAP-based;
service  S-30,   P-18,   RESTful;
service  S-31,   P-19,   RESTful; }
```

Global Constraints

```
// defining total QoS of Requested composite service;
constraint global {
qos = {
        quality =2.5;
        availabili=3; }
evaluationFactor=1 // it should be completed in Selection phase;}
```

Data & Control Specification



شکل ۵-۶: CCSM در سناریوی" استخدام نیرو در سازمان"

۲.۴.۵ تحلیل اهداف انجام سناریوها و میزان تحقق آن‌ها

در این بخش به بررسی اهدافی که از انجام دو سناریوی کاربردی " استخدام نیرو در سازمان" در این فصل و "برنامه ریزی سفر" در فصل‌های ۳ و۴ داشتیم، می‌پردازیم.

سناریوی " استخدام نیرو در سازمان" نمونه‌ای از یک سناریوی کاربردی است که درون سازمانی رخ می‌دهد و در این سناریو به ازای تمامی وظیفه‌مندی‌های درخواستی، فعالیت‌های متناظری در مخزن اصلی موجود بوده لذا بررسی شاخص قابلیت ترکیب سرویس‌های موجود از انجام این سناریو مد نظر نبوده است. اما سایر شاخص‌ها قابل بررسی هستند. سناریوی "برنامه ریزی سفر" که شرح آن در بخش ۱.۳ بیان شد، از جمله سناریوهای پرکاربرد در بررسی قابلیت‌های یک روش ترکیب سرویس است که در مقالات متعددی مورد استفاده قرار گرفته است. با انجام این سناریو و با توجه به عناصر ترکیب موجود در مخزن اصلی جدول ۷-۱ در پیوست۱ که برای این سناریو در نظر گرفته شده‌اند، امکان بررسی تمامی شاخص‌های معرفی شده، وجود دارد. در ادامه هر دو سناریو برای تحلیل شاخص مد نظر گرفته شده است. شاخص قابلیت ترکیب سرویس‌های موجود با این سناریوی بررسی می‌گردد.

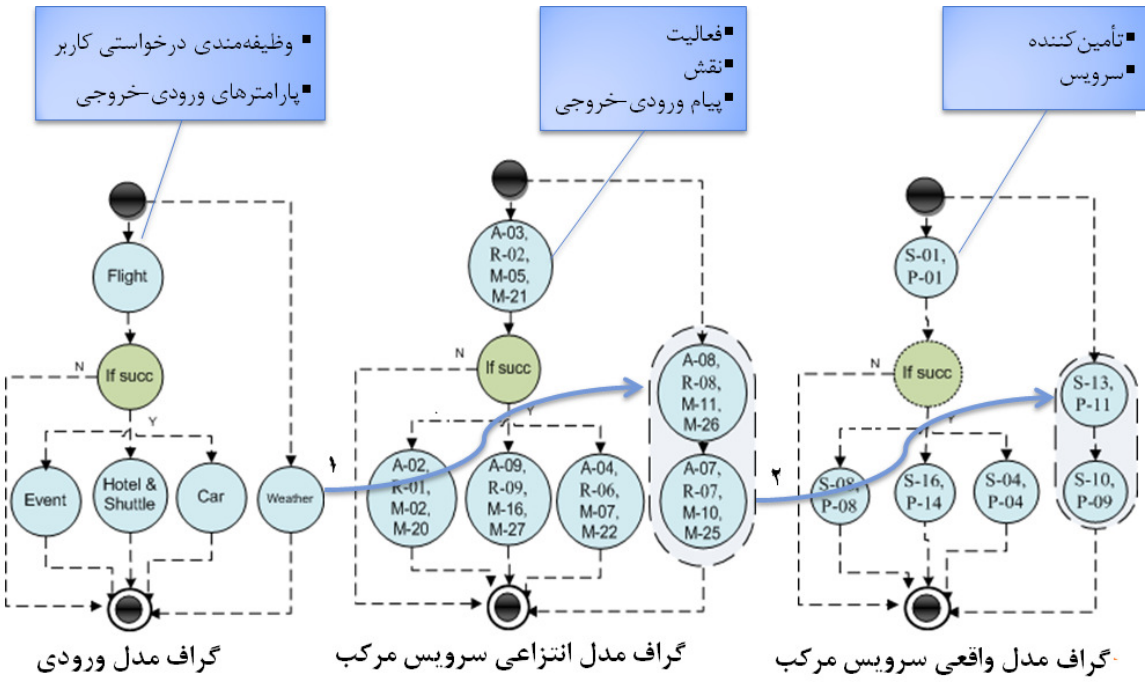
- **راهبرد ترکیب**: مدل‌رانه بودن ترکیب، بر اساس مدل ورودی، مدل انتزاعی سرویس مرکب و مدل واقعی سرویس مرکب که هر سه نوعی PIM به شمار می‌آیند، متناظر با سناریوها و فازهای مربوط به تبدیل مدل‌ها به یکدیگر که در واقع تبدیل از یک مدل PIM با سطح انتزاع بالا به مدل‌های PIM دیگر با سطوح انتزاع پایین‌تر و جزئیات بیشتر است، استفاده از راهبرد مدل‌رانه در روش ترکیب پیشنهادی، قابل مشاهده است.
- **سطح خودکار سازی**: از آنجا که در روش پیشنهادی در فازهای کشف، ساخت و انتخاب با تعامل با کاربر خروجی‌های هر فاز به تأیید کاربر می‌رسد لذا روش از نظر سطح خودکار سازی در سطح نیمه خودکار است. همچنین در صورتی که با توجه به محدودیت‌ها و نیازمندی‌های کاربر، خروجی یک فاز به طور مثال فاز کشف موفق نباشد برای بالا رفتن میزان موفقیت در برآورده ساختن نیاز کاربر، از کاربر درخواست تغییر در نیازمندی و یا پذیرش خروجی بدست آمده داده می‌شود. لازم به ذکر است که برای حذف برخی از تعاملات مورد نیاز با کاربر می‌توان با تعریف شاخص‌هایی اقدام کرد که لازم است برخی از این شاخص‌ها در کارهای آتی این تحقیق به طور دقیق‌تری بررسی شوند تا کیفیت خروجی هر فاز بدون دخالت و تأیید کاربر نیز در سطح مناسبی باشد. استفاده از این شاخص‌ها برای جلوگیری از تعامل با کاربر در این تحقیق مد نظر نبوده است.

- **منطق ترکیب:** در هر یک از سناریوها در ابتدای فرآیند ترکیب در فاز توصیف نیازمندی‌ها، کاربر جریان داده و کنترل ما بین وظیفه‌مندی‌های درخواستی خود را به صورت گراف معین شده در این تحقیق در بخش پایانی مدل ورودی، وارد می‌کند. از آنجا که کاربر فرآیند ترکیب را به صورت نحوه جریان مابین وظیفه‌مندی‌های درخواستی می‌دهد، لذا بر اساس دسته‌بندی‌های موجود برای منطق ترکیب، روش پیشنهادی در دسته فرآیند محور است. فرآیند ترکیب برای سناریوی استخدام نیرو در سازمان در شکل ۵-۳ در گراف مدل ورودی، نمایش داده شده است.
- **تطابق و انتخاب:** در فاز کشف فعالیت‌های متناظر با وظیفه‌مندی‌های درخواستی، ابتدا بر اساس بخش اول از مدل ورودی شکل ۵-۴که با عنوان Functionality Definition نشان داده شده است، جستجو بر اساس طبقه‌ی هر یک از چهار وظیفه‌مندی درخواستی، بر روی عناصر کلاس فعالیت در جدول ۷-۱ از پیوست ۱ انجام می‌شود. از طرف دیگر با استفاده از WordNet کلمات معادل هر یک از چهار وظیفه‌مندی درخواستی استخراج می‌گردد، پس به ازای هر وظیفه‌مندی دو مجموعه بدست آمده است، یکی مجموعه کلمات معادل با وظیفه‌مندی درخواستی و دیگری مجموعه فعالیت‌های هم طبقه با وظیفه‌مندی درخواستی، در ادامه ستون وظیفه‌مندی از جدول فعالیت‌ها با کلمات معادل مقایسه و فعالیت‌های مناسب انتخاب می‌شوند، سپس برای مقایسه پارامترهای ورودی-خروجی نیز همین کار با استفاده از Wordnet تکرار می‌گردد و در پایان با مقایسه میزان تشابه پارامترهای ورودی-خروجی وظیفه‌مندی با پیام‌های ورودی-خروجی فعالیت‌های کشف شده، چند تایی[۱] <فعالیت؛ پیام ورودی-خروجی؛ نقش> مناسب انتخاب می‌شوند و پس از تأیید کاربر، وارد فاز بعدی می‌گردند. در نتیجه می‌توان گفت که تطبیق و انتخاب در روش پیشنهادی در دسته بندی، انتخاب بر اساس کلمات کلیدی قرار می‌گیرد.
- **قابلیت ترکیب سرویس‌های موجود:** در سناریوی "برنامه ریزی سفر" برای وظیفه‌مندی درخواستی پیش بینی آب و هوا بر اساس پارامترهای ورودی-خروجی درخواستی، فعالیت متناظری در عناصر موجود در مخزن اصلی یافت نشد، لذا الگوریتم ترکیب سرویس در این سناریو مورد ارزیابی قرار گرفت، اجرای این الگوریتم به طور کامل در بخش ۳.۲.۴ و در فاز کشف، به همراه شکل برای این سناریو تشریح گردید و در شکل ۵-۷ نیز جایگزینی دو فعالیت به جای وظیفه‌مندی weather حاصل اجرای موفق الگوریتم ترکیب فعالیت در فاز کشف است مراحل ۱ و ۲ در

---

[۱] Tuple

شکل ۵-۷ نمایش این جایگزینی هستند. این الگوریتم ترکیب را تا سطح مشخصی که کاربر تعیین می‌کند پیش می‌برد تا نیازمندی درخواستی را بر آورده سازد.



شکل ۵-۷: نمایش ترکیب فعالیت‌ها در" سناریوی برنامه ریزی سفر"

- **زمان ترکیب:** از آنجا که فرآیند ترکیب در این چارچوب مرحله‌ی اجرا و مدیریت سرویس مرکب را در بر نمی‌گیرد لذا زمان ترکیب، زمان طراحی است. اما در روش پیشنهادی همان‌طور که در روال سناریو و ایجاد مدل‌ها نیز شاهد بودیم، در فاز ساخت سرویس مرکب واقعی است که فعالیت‌های متناظر با وظیفه‌مندی‌های درخواستی به سرویس‌های واقعی مقید[۱] می‌شوند و لذا نسبت به سایر روش‌ها که ترکیب را در زمان طراحی انجام می‌دهند، در این روش ترکیب به آخرین مرحله طراحی منتقل شده است تا پویایی روش ترکیب با استفاده از مدل‌های انتزاعی تأمین گردد.
- **پشتیبانی از ترکیب سرویس‌های نا همگن:** از آنجا که طبق مدل واقعی سرویس مرکب که در شکل ۵-۶ نشان داده شده است، سرویس‌های انتخابی هم شامل سرویس‌های مبتنی بر SOAP و هم شامل سرویس‌های

[۱] Bind

RESTful هستند. لذا این سناریو امکان‌پذیری این شاخص را نشان می‌دهد. همچنین در سناریوی "برنامه ریزی سفر" نیز همان‌طور که در شکل ۳-۶ ، نشان داده شده است، مدل واقعی سرویس مرکب شامل هر دو نوع سرویس است.

- **پشتیبانی از ویژگی‌های غیر وظیفه‌مندی:** برای پشتیبانی از ویژگی‌های غیر وظیفه‌مندی در روش پیشنهادی ابتدا لازم بود این ویژگی‌ها را برای سرویس‌های موجود در مخزن اصلی سرویس‌ها در نظر بگیریم، که این نیازمندی با در نظر گرفتن مشخصه‌های مورد نیاز برای عنصر سرویس در مدل متا داده‌ای پاسخ داده شد. سپس لازم بود به ازای هر سرویس مرکب درخواستی، میزان ویژگی‌های غیر وظیفه‌مندی مورد نظر کاربر در قالب مدل ورودی دریافت شود، که این نیازمندی با در نظر گرفتن دو بخش محدودیت‌های وظیفه‌مندی و محدودیت‌های سراسری در مدل ورودی پاسخ داده شده است، در بخش اول کاربر می‌تواند میزان ویژگی غیر عملکردی برای هر یک از سرویس‌ها را جداگانه قید کند و در بخش دوم این ویژگی‌ها را برای سرویس مرکب نهایی تعیین می‌کند. این دو بخش در سایر مدل‌ها نیز وجود دارند و مقادیر بدست آمده بر اساس سرویس‌های کشف شده‌ی متناظر با نیازمندی‌های درخواستی جایگزین می‌شوند، در مواردی که به ازای میزان درخواستی کاربر سرویسی یافت نشود برای تغییر این نیازمندی با کاربر تعامل می‌شود. این نیازمندی‌ها هم در فاز ساخت سرویس برای انتخاب هر سرویس و هم در فاز انتخاب برای محاسبه شاخص ارزیابی (evaluationFactor در بخش محدودیت‌های سراسری مدل واقعی سرویس مرکب) و انتخاب سرویس مرکب واقعی نهایی از بین سایر سرویس‌های مرکب کاندیدا مورد استفاده قرار می‌گیرد. در این تحقیق در حال حاضر پشتیبانی از دو ویژگی غیر عملکردی کیفیت و میزان در دسترس بودن پشتیبانی می‌شود، اما با توجه به در نظر گرفتن تمهیدات پشتیبانی از سایر ویژگی‌ها نیز به سادگی امکان پذیر خواهد بود که به عنوان کارهای آتی این تحقیق مطرح است.

۳.۴.۵ نظرسنجی از خبرگان

آخرین مرحله از فرآیند ارزیابی، بررسی قابلیت‌های چارچوب و فرآیند توسعه ترکیب سرویس MDCHeS، از دیدگاه خبرگان است. نظرات خبرگان از طریق پرسش‌نامه‌ای که شامل جدول ارزیابی نمایش داده شده در جدول ۵-۱ است، جمع‌آوری شد. برخی از نظرات پس از ارائه‌ای[۱] از روند توسعه ترکیب در چارچوب MDCHeS در طی جلسه گروه تحقیقاتیASER، جمع‌آوری شد و برخی به صورت ارسال فایلی از پرسش‌نامه به همراه توصیفات مورد نیاز جهت شناخت از چارچوب پیشنهادی از طریق فایل نظرسنجی که در پیوست ۳ آورده شده است، جمع‌آوری شد. خبرگانی که در تکمیل این پرسشنامه ما را یاری کردند تعداد ۱۱ نفر بودند که به دو گروه افراد دانشگاهی- آکادمیک و متخصصین حوزه صنعت تقسیم می‌شوند که آشنایی کافی در زمینه معماری سرویس گرا و ترکیب سرویس داشته‌اند.

۱.۳.۴.۵ نتایج ارزیابی

جهت اخذ نتایج ابتدا معماری و فرآیند توسعه در چارچوب پیشنهادی به همراه مثالی در این زمینه، در قالب ارائه‌ای کامل برای شرکت‌کنندگان در نظرسنجی، تشریح گردید. سپس پرسش‌نامه‌ای حاوی ۱۶ سوال در اختیار خبرگان قرار داده شد. سوالات این پرسش‌نامه دو جنبه قابلیت‌های کلی و عام چارچوب و قابلیت‌های خاص آن را مورد پرسش قرار می‌دادند. تلاش شده سوالات به گون‌های طراحی شوند که پاسخ‌دهندگان بتوانند نظراتشان را صریحاً بیان کنند. جدول ۵-۱ سؤالات نظر سنجی را نشان می‌دهد. خبرگان باید هر یک از سوالات را بر اساس اینکه موافق یا مخالف آن بودند از دامنه ۱ تا ۵ انتخاب می‌کردند. عدد ۱ به معنی کاملاً مخالف، عدد ۱ مخالف، عدد ۳ متوسط، عدد ۴ موافق و در نهایت عدد ۵ کاملاً موافق است. در ادامه پس از جمع آوری نتایج، از آزمون‌های آماری جهت تفسیر نتایج استفاده گردید. در جدول ۵-۱، $m^{۲}$ بیانگر میانگین اعداد، $sd^{۳}$ بیانگر انحراف معیار با دقت دو رقم اعشار و $np^{۴}$ بیانگر تعداد پاسخ‌های مثبت (انتخاب گزینه‌ی ۴ و ۵) است. ستون‌های ۱ تا ۵ نیز تعداد پاسخ‌های مربوط به هر گزینه را نشان می‌دهد. ممکن است همه شرکت‌کنندگان بر برخی سؤالات پاسخ نداده باشند مانند سؤال ۱۰.

---

[۱] Presentation

[۲] Mean

[۳] Standard Deviation

[۴] Number of Positive Responses

جدول ۵-۱: نتایج ارزیابی نظرات خبرگان در خصوص قابلیت‌های چارچوب پیشنهادی

ردیف	قابلیت‌های چارچوب ترکیب سرویس‌های وب MDCHeS	۱	۲	۳	۴	۵	np	m	sd
۱	کاربر درخواست‌کننده‌ی سرویس مرکب برای فرآیند توسعه ترکیب سرویس لازم نبست دانش فنی عمیقی داشته باشد.	۰	۰	۴	۵	۲	۷	۳.۸۱	۰.۷۵
۲	تعامل با چارچوب در کلیه فازها، برای توسعه دهنده سرویس مرکب، ساده و شفاف است.	۰	۰	۲	۶	۳	۹	۴.۰۹	۰.۷۰
۳	تعامل با کاربر تنها به منظور بهبود روال توسعه سرویس مرکب است.	۰	۰	۲	۷	۱	۸	۳.۹۰	۰.۵۶
۴	چارچوب نیازمندی‌های عملکردی و غیر عملکردی سازمان برای ایجاد سرویس‌های مرکب درخواستی را لحاظ می‌کند.	۰	۴	۲	۳	۲	۵	۳.۲۷	۱.۱۹
۵	چارچوب قابلیت استفاده از دارایی‌های سازمان و استفاده مجدد از سرویس‌های مرکب ایجاد شده در سازمان را دارد.	۰	۰	۰	۸	۳	۱۱	۴.۲۷	۰.۴۶
۶	چارچوب قابلیت استفاده از انواع سرویس وب خارج از سازمان، در سرویس مرکب درخواستی را دارد.	۰	۰	۱	۵	۵	۱۰	۴.۳۶	۰.۷۰
۷	چارچوب قالب مناسبی برای طرح نیازمندی در فاز توصیف سرویس مرکب درخواستی در نظر گرفته است.	۰	۱	۵	۲	۲	۴	۳.۵۰	۰.۸۶
۸	چارچوب فاز کشف فعالیت را به طور مناسب توسعه داده است.	۰	۰	۵	۳	۲	۵	۳.۷۰	۰.۸۲
۹	در صورت عدم کشف سرویس متناظر، چارچوب قابلیت ترکیب عناصر موجود به منظور پاسخ‌گویی به نیاز درخواستی را دارا می‌باشد.	۰	۰	۱	۶	۴	۱۰	۴.۲۷	۰.۶۴
۱۰	چارچوب فاز ساخت سرویس مرکب واقعی (CCSM) را به طور مناسب توسعه داده است.	۰	۰	۴	۷	۰	۷	۳.۶۳	۰.۵۰
۱۱	چارچوب فاز انتخاب سرویس نهایی را بر اساس شاخص‌های مناسبی توسعه داده است.	۰	۱	۰	۸	۲	۱۰	۴.۰۰	۰.۷۷
۱۲	چارچوب از فناوری مناسبی برای فاز ساخت سرویس مرکب قابل اجرا (ECSM) استفاده کرده است.	۰	۱	۳	۶	۱	۷	۳.۶۶	۰.۸۰
۱۳	استفاده از چارچوب فرآیند توسعه سرویس مرکب را تسهیل می‌بخشد.	۰	۰	۱	۸	۲	۱۰	۴.۰۹	۰.۴۷
۱۴	استفاده از چارچوب در سازمان‌ها عملیاتی است.	۰	۳	۵	۳	۰	۳	۳.۰۰	۰.۷۷
۱۵	چارچوب قابلیت تطبیق با تغییرات محیط سرویس‌های وب و پویایی ترکیب را دارا می‌باشد.	۰	۲	۵	۴	۰	۴	۳.۱۸	۰.۷۵
۱۶	درصد پاسخگویی موفق به نیازمندی درخواستی با استفاده مداوم از چارچوب به دلیل تکمیل مخزن اصلی، بیشتر می‌شود.	۰	۰	۰	۳	۷	۱۰	۴.۷۰	۰.۴۸

۲.۳.۴.۵ تحلیل نتایج ارزیابی قابلیت‌های چارچوب پیشنهادی

در بخش قبل نتایج ارزیابی قابلیت‌های چارچوب از دیدگاه خبرگان نشان داده شد. می‌توان سوالات را به دو دسته تقسیم بندی کرد، دسته‌ی اول سوال‌هایی که جنبه عام و کلان چارچوب را مورد ارزیابی قرار می‌دهند که عبارتند از سوالات ۱،۲، ۳، ۱۳، ۱۴، ۱۵و ۱۶ و دسته دوم سوالاتی که جنبه خاص چارچوب را مورد پرسش قرار می‌دهند شامل سوالات ۴، ۵، ۶، ۷، ۸، ۹، ۱۰، ۱۱و ۱۲.

در خصوص نتایج بدست آمده از سه پارامتر میانگین، انحراف معیار و پاسخ‌های مثبت، می‌توان نتیجه گرفت که آیا چارچوب از دیدگاه خبرگان در دو جنبه قابلیت‌های عام و خاص تأیید شده است یا خیر.

از آنجا که نمره ۱و ۲ مخالفت، نمره ۴ و ۵ موافقت و نمره ۳ مابین این دو را در قابلیت مورد پرسش نشان می‌دهد، لذا میانگین بالای ۳.۵ بیانگر تأیید خصوصیت مورد پرسش قرار گرفته از سوی خبرگان است، می‌توان اذعان نمود که از ۱۶ سوال بررسی شده تنها سوالات ۴، ۱۴و ۱۵ نمره کمتر از میانگین مورد قبول را کسب کرده‌اند. (هر سه سوال طبق جدول ۵-۱ دارای میانگین بالاتر ۳ و زیر ۳.۵ هستند که بیانگر عدم مخالفت و موافقت است.) سؤال ۴ در نظر گرفتن نیازمندی‌های عملکردی و غیر عملکردی سازمان برای ایجاد سرویس‌های مرکب درخواستی را مورد پرسش قرار می‌دهد، که ۵ نفر از شرکت کنندگان موافق و ۴ نفر مخالف این موضوع بوده‌اند و بقیه نظر متوسط داشته‌اند. این قابلیت در این مرحله از توسعه چارچوب تنها به دو ویژگی قابلیت دسترس‌پذیری و کیفیت از نیازمندی‌های غیر عملکردی پرداخته و تکمیل این بخش از جمله کارهای آینده برای چارچوب در نظر گرفته شده است. در نتیجه جامع نبودن این ویژگی برای تعریف دامنه دقیق برای نیازمندی‌های مورد پشتیبانی و همچنین عدم پشتیبانی از سایر ویژگی‌های غیر عملکردی، منجر به عدم تأیید کامل این قابلیت در بین خبرگان بوده است.

سؤال ۱۴ جنبه عملیاتی چارچوب را مورد پرسش قرار می‌دهد که ۳ نفر از افراد با این پرسش موافق هستند و تنها ۳ نفر مخالف هستند و بقیه نظر متوسط داشته‌اند. با توجه به اینکه چارچوب ارائه شده از جمله کارهای اولیه تحقیقاتی در این زمینه است و هنوز در این مرحله ابزاری برای آن توسعه داده نشده است، انتظار عملیاتی بودن آن در مراحل ابتدایی کمی دور از انتظار می‌نماید .به بیان دیگر، چارچوب ارائه شده اگر بخواهد در ابعاد سازمان‌ها و به صورت عملیاتی مورد استفاده قرار گیرد نیازمند تقویت فازهای پنج‌گانه در ابعاد مختلف پیاده‌سازی ابزاری جامع است که تکمیل این‌ها به عنوان کارهای آینده این تحقیق مطرح شده است.

سؤال ۱۵ قابلیت پویایی چارچوب و تطبیق با تغییر در محیط سرویس‌ها را نشان می‌دهد، این سؤال تعداد ۴ موافق و ۲ مخالف داشته است (سایر افراد نظر متوسط داده‌اند) از آنجا که این چارچوب ترکیب سرویس را در آخرین مرحله طراحی انجام می‌دهد و در این چارچوب به بخش مدیریت و اجرای سرویس مرکب نپرداخته‌ایم، پس پویایی زمان اجرا را شامل نمی‌شود و تنها به دلیل استفاده از مفاهیم انتزاعی و مستقل کردن مدل‌های اولیه‌ی سرویس مرکب از عناصر محیط واقعی نظیر تأمین‌کننده و سرویس، بخشی از پویایی را تأمین می‌کند. با توسعه چارچوب دیگری تحت عنوان چارچوب مدیریت و اجرای سرویس مرکب که به عنوان آخرین کار آینده مطرح شده است، قابلیت پویایی ترکیب سرویس بیشتر خواهد شد.

جالب توجه است که از بین سؤالات مطرح شده، سوال ۱۶ بیشترین امتیاز و سوالات ۶، ۵ و۷ به ترتیب امتیازهای بعدی را به خود اختصاص داده‌اند. این سؤالات در مورد قابلیت‌ها و ویژگی‌های اساسی چارچوب پیشنهادی بوده‌اند که امتیاز بسیار مناسبی با تعداد بیشتر یا مساوی ۱۰ موافق را کسب کرده‌اند. سایر نتایج این پرسش‌نامه از طریق موارد بدست آمده در جدول ۵-۱ قابل بررسی و مشاهده است.

۵.۵ مقایسه روش پیشنهادی با روش‌های مشابه

در این بخش با توجه به چارچوب‌های مشابه موجود که از نظر وظیفه‌مندی، عملکردی مشابه با چارچوب MDCHeS پیشنهاد شده در این تحقیق دارند و در بخش ۳.۲ معرفی شدند و با در نظر گرفتن شاخص‌های معرفی شده در بخش ۳.۵۴.۲، مقایسه‌ای با روش پیشنهادی در جدول ۵-۲، نشان داده شده است. لازم به ذکر است که برخی چارچوب‌های موجود در این جدول توسعه و اجرای سرویس مرکب را در بر می‌گیرند، اما از آنجا که حیطه وظیفه‌مندی چارچوب پیشنهادی تنها توسعه ترکیب سرویس است، شاخص‌های در نظر گرفته شده به مقایسه این بخش تمرکز دارند.

جدول ۵-۲: مقایسه چارچوب MDCHeS با چارچوب‌های مشابه موجود

چارچوب ترکیب سرویس	راهبرد ترکیب	سطح خود کارسازی	تطابق و انتخاب سرویس	منطق ترکیب	صحت ترکیب	زمان ترکیب	ترکیب سرویس ناهمگن	پشتیبانی از QoS
MDCHeS	مدل‌رانه	نیمه خودکار	کلمات کلیدی	فرآیند محور		طراحی	*	*
eFlow [۴۵]	پویا	نیمه خودکار-دستی	-	فرآیند محور		طراحی		
METEOR-S [۳۶]	معنایی	نیمه خودکار-دستی	-	فرآیند محور	*	طراحی-اجرا		*
Web Transact [۳۵]	پویا	نیمه خودکار-دستی	-	قانون محور	*	طراحی		
DynamiCoS [۴۱]	معنایی	خودکار	IOPE	فرآیند محور	*	اجرا		*

۶.۵ جمع‌بندی مطالب فصل

در این فصل به ارزیابی روش ارائه شده و فازهای مربوط به فرآیند توسعه در این چارچوب پرداختیم. از این رو ابتدا نحوه‌ی ارزیابی چارچوب معرفی شد و بیان شد که دو جنبه در ارزیابی روش پیشنهادی مد نظر است، در وجه اول ابتدا لازم است فازهای فرآیند توسعه در چارچوب ارزیابی گردد و درستی آن‌ها تأیید شود و در وجه دوم می‌بایست کلیت چارچوب مورد بررسی و ارزیابی قرار گیرد. برای شروع ارزیابی ابتدا با شاخص‌های ارزیابی برای یک روش ترکیب سرویس آشنا شدیم و سپس بر اساس دو سناریوی کاربردی "برنامه ریزی سفر" و " استخدام نیرو در سازمان" ارزیابی هر فاز و همچنین فرآیند توسعه ترکیب سرویس در چارچوب انجام شد. سپس برای ارزیابی کلیت چارچوب از نظرسنجی از خبره کمک گرفتیم و در ادامه نتایج حاصل از نظرسنجی مورد تحلیل قرار گرفت که تمامی این نتایج قابل قبول و رضایت بخش بود. در پایان نیز قابلیت‌های چارچوب پیشنهادی با سایر چارچوب‌های مشابهی موجود بر اساس شاخص‌های معرفی شده، مقایسه گردید.

۶ فصل ششم – خلاصه و نتیجه گیری

۱.۶ مقدمه

در فصل اول مسأله مورد مطالعه در این پایان نامه به همراه ویژگی‌های راه حل پیشنهادی معرفی شد، سپس در فصل ۲ ضمن بیان مفاهیم و اصطلاحات موجود در حیطه مسأله، روش‌های مشابهی موجود برای حل این مسأله، مورد بررسی و ارزیابی قرار گرفتند و نقاط قوت و ضعف روش‌های آن‌ها تشریح گردید. در ادامه و در فصل ۳ مبانی روش پیشنهادی مطرح گردید و سپس در فصل ۴ به توصیف فازهای توسعه ترکیب سرویس در روش پیشنهادی و الگوریتم‌های مربوط به هر فاز پرداختیم. در پایان نیز روش پیشنهادی با سناریوهای کاربردی و چارچوب پیشنهادی با نظرسنجی از خبره، مورد ارزیابی قرار گرفتند و از نظر توانایی حل مسأله و شاخص‌های مورد انتظار ارزیابی شدند. در این فصل ابتدا بار دیگر مسأله مورد مطالعه و ویژگی‌های راه حل مورد انتظار را مرور کرده و این ویژگی‌ها را با خصوصیات روش پیشنهادی مقایسه و نتایج حاصل از این تحقیق بیان خواهد شد. در پایان نیز زمینه‌های تحقیق برای تکمیل و بهبود روش پیشنهادی معرفی خواهند شد.

۲.۶ بازبینی میزان تحقق اهداف پایان نامه

در فصل سوم هدف اصلی این پایان نامه به صورت زیر تعریف گردید:

"ارائه روش ترکیبی جامع که به صورت مدل‌رانه قادر به توسعه نیمه خودکار ترکیب انواع سرویس‌های وب با در نظر گرفتن نیازمندی‌های غیر عملکردی برای سرویس مرکب درخواستی باشد."

با توجه به این تعریف ویژگی‌های اصلی روش مورد نیاز عبارتند از:

۱. قابلیت توصیف سرویس‌های وب RESTful و مبتنی بر SOAP به صورت همزمان با استفاده از مدل متاداده ای پیشنهادی.
۲. قابلیت تغییر در ویژگی‌های عناصر ترکیب، نظیر تأمین‌کنندگان و ویژگی سرویس‌های آن‌ها در حین ترکیب، به دلیل استفاده از مفاهیمی انتزاعی نظیر فعالیت، پیام و نقش در مدل متاداده ای روش پیشنهادی.
۳. قابلیت ارائه مدل قابل اجرا برای انواع سرویس‌های وب در خروجی بنا به درخواست کاربر یا با توجه به سرویس‌های شرکت‌کننده در سرویس مرکب نهایی.
۴. تأمین مکانیزمی جهت ترکیب بازگشتی به منظور افزایش قابلیت استفاده مجدد از سرویس‌ها. به طوری که یک سرویس مرکب، خود می‌تواند در روش پیشنهادی برای ترکیب‌های بعدی مورد استفاده قرار گیرد.

۵. استفاده از ایده معماری مدل‌رانه در روال توسعه ترکیب برای تبدیل مدل‌های مورد استفاده در ترکیب برای کنترل پیچیدگی روال ترکیب و توانایی پشتیبانی از تغییرات در محیط سرویس‌های وب.

۶. در نظر گرفتن ویژگی‌های غیر عملکردی سرویس‌ها در فاز ساخت و انتخاب سرویس و همچنین در توصیف سرویس‌های قابل دسترس در مخزن اصلی.

حال با توجه به اهداف مورد انتظار و ویژگی‌های روش MDCHeS به بررسی میزان دست یابی به این اهداف خواهیم پرداخت.

۱. **قابلیت توصیف انواع سرویس‌های وب**: همان‌طور که در فصل ۳ بیان شد، مدل متاداده ای پیشنهادی با در نظر گرفتن هر یک از ویژگی‌های مربوط به سرویس‌های وب مبتنی بر SOAP و RESTful، امکان توصیف هر دو نوع از سرویس‌های وب را فراهم آورده است. لذا در مخزن اصلی چارچوب پیشنهادی قابلیت ذخیره هر دو نوع سرویس وب وجود دارد و در روال ترکیب از هر یک از آن‌ها در صورت مناسب بودن برای پاسخگویی به نیازمندی درخواستی، استفاده خواهد شد. پشتیبانی از این ویژگی در عناصر موجود در مخزن اصلی برای سناریوهای کاربردی موجود در پیوست ۱ مشاهده می‌شود.

۲. **قابلیت تغییر در ویژگی‌های عناصر ترکیب**: در فصل ۳ بیان شد که هدف از وجود عناصر فعالیت، نقش و پیام افزودن انتزاع به مدل سرویس مرکب در حین مراحل ابتدایی ترکیب است. عنصر فعالیت در واقع مفهومی انتزاعی از سرویس‌های وب محیط واقعی است و استفاده از آن ضمن تسهیل فرآیند کشف، این قابلیت را به روش ترکیب می‌دهد که با داشتن مدل انتزاعی سرویس مرکب (ACSM) که از عناصر نام برده شده، تشکیل شده است، خود را از تغییر ویژگی‌های سرویس‌های وب محیط واقعی و سایر عناصر وابسته به آن‌ها نظیر تأمین‌کننده و غیره، مستقل کند.

۳. **قابلیت ارائه مدل قابل اجرا از انواع سرویس وب مرکب در خروجی**: همان‌طور که در مورد اول ویژگی‌های روش پیشنهادی مطرح شد، مدل متاداده ای معرفی شده در چارچوب، امکان توصیف و ذخیره انواع سرویس‌های وب در مخزن اصلی چارچوب را فراهم آورده است، همچنین در روال ترکیب نیز امکان استفاده از انواع سرویس‌های وب فراهم شده است، لذا در این مرحله تنها کافی است روشی برای تولید خروجی که مدلی قابل اجرایی از سرویس مرکب درخواستی است، انتخاب شود. پس از انجام مطالعات کافی استفاده از توسعه BPEL برای سرویس‌های RESTful (BPEL for REST)

بدین منظور پیشنهاد شد، که در بخش ۱۷.۱.۳.۲ به طور کامل تشریح شد، که در آن ضمن افزودن قابلیت فراخوانی انواع سرویس‌ها، با افزودن برچسب[۱]های مناسب امکان ایجاد فایل BPEL برای انواع سرویس‌های وب را نیز فراهم می‌کند.

۴. **پشتیبانی از ترکیب بازگشتی**: از آنجا که در روش MDCHeS پس از انتخاب نهایی مدل واقعی سرویس مرکب (CCSM)، فعالیت مرکب مربوط به این سرویس به عنوان فعالیتی مرکب به همراه سایر عناصر مرتبط ترکیب به مخزن اصلی و مدل‌های مرتبط با فرآیند توسعه ترکیب آن به مخزن مدل‌های چارچوب افزوده می‌شوند، لذا در سایر درخواست‌ها، این فعالیت می‌تواند جهت پاسخگویی به نیاز درخواستی مورد استفاده قرار گیرد. همچنین در مواردی که فعالیت متناظر با وظیفه‌مندی درخواستی کاربر، یافت نشود و الگوریتم مربوط به ترکیب فراخوانی شود، پس از ایجاد فعالیت مرکبی که پاسخگوی نیاز کاربر باشد، این فرآیند می‌تواند تا مرحله مدل اجرایی این سرویس مرکب ادامه یابد و برای استفاده مجدد به مخزن اصلی و مخزن مدل‌ها به عنوان فعالیت مرکب کاربردی، افزوده گردد. مانند فعالیت‌هایی با شناسه A-11 و A-13 در عناصر فعالیت جدول ۷-۱ از پیوست ۱.

۵. **استفاده از ایده معماری مدل‌رانه برای کنترل پیچیدگی**: هدف اصلی در استفاده از ایده مدل‌رانه، جدا کردن منطق ترکیب از فناوری‌ها و توصیفات ترکیب، به منظور بالا بردن انتزاع ترکیب و کنترل پیچیدگی روال ترکیب است. بالا بردن انتزاع منجر به افزایش سرعت توسعه بوده و عامل کنترل پیچیدگی روال توسعه ترکیب می‌گردد. در روش پیشنهادی ضمن معرفی مدل متاداده ای از چهار مدل در فازهای مختلف توسعه ترکیب استفاده شد و در واقع هر فاز تبدیل نیمه خودکار این مدل‌ها به یکدیگر را انجام می‌داد. مدل ورودی (IM)، مدل انتزاعی (ACSM) و مدل واقعی (CCSM) سرویس مرکب از نوع مدل PIM و مدل اجرایی سرویس مرکب از نوع مدل PSM هستند.

۶. **در نظر گرفتن ویژگی‌های غیر عملکردی سرویس‌ها**: ویژگی‌های غیر عملکردی قابل پشتیبانی در روش MDCHeS شامل میزان کیفیت و دسترس پذیری هستند. این دو ویژگی هم در توصیف و ذخیره سرویس‌های وب واقعی از ویژگی‌های عنصر سرویس در مدل متاداده هستند و هم به عنوان ویژگی‌هایی که در مدل IM توسط کاربر برای سرویس مرکب درخواستی توصیف می‌شود. این ویژگی‌ها در دو مرحله تأثیر گذار هستند، یکی در فاز کشف فعالیت مناسب برای

---

[۱] Tag

وظیفه‌مندی درخواستی که تناسب ویژگی درخواستی احتمالی کاربر برای جستجو مدنظر قرار داده می‌شود و تأثیر اصلی آن در فاز انتخاب CCSM مناسب برای سرویس مرکب درخواستی بر اساس ویژگی غیر عملکردی درخواستی کاربر برای سرویس مرکب مورد نظر است.

۳.۶ ویژگی نوآوری‌های تحقیق

در بخش قبل ویژگی‌های روش ارائه شده را بیان کردیم، برخی از این ویژگی‌ها از نوآوری‌های این تحقیق به شمار می‌آید اما برخی با استفاده از ایده‌های ارائه شده، بدست آمده است. در این بخش ویژگی نوآوری‌هایی این تحقیق در حیطه ترکیب سرویس وب ارائه می‌گردد.

- ارائه چارچوبی مدل‌رانه و جامع برای پوشش فازهای توسعه ترکیب سرویس وب.
- ارائه فازهای توسعه ترکیب سرویس وب مبتنی بر ایده معماری مدل‌رانه در چارچوب پیشنهادی.
- طراحی مدل متاداده ای برای افزایش قابلیت تغییرپذیری در اضافه شدن و کم شدن سرویس، تأمین‌کننده و تغییر در ویژگی‌های هر کدام و به نوعی افزودن خاصیت پویایی بیشتر به فرآیند ترکیب سرویس.
- پشتیبانی از نیازمندی‌های مربوط به توصیف و ذخیره انواع سرویس‌های وب در مخزن و استفاده از آن‌ها در سرویس‌های مرکب ایجاد شده.
- عدم نیاز به توصیفات معنایی و هستان شناسی برای کاربردی و صنعتی کردن روش و چارچوب پیشنهادی.
- استفاده از دیدگاه فازبندی شده و مدل‌رانه برای غلبه بر پیچیدگی روال ترکیب سرویس و افزایش قابلیت مقیاس‌پذیری روش و چارچوب پیشنهادی.
- در نظر گرفتن ترجیحات و محدودیت‌های کاربر برای انتخاب سرویس‌های دخیل در ترکیب.
- قابلیت استفاده مجدد از سرویس‌های مرکب تولید شده توسط چارچوب و امکان انجام ترکیب به صورت بازگشتی.
- قابلیت ترکیب سرویس‌های موجود برای پاسخ‌گویی به وظیفه‌مندی که سرویس متناظر آن در مخزن موجود نیست با ارائه الگوریتم ترکیب فعالیت به شکل پایین به بالا.

۴.۶ محدودیت‌ها

یکی از محدودیت‌های این تحقیق عدم دسترسی کامل به سرویس‌های وب واقعی در یک سازمان و استخراج توصیف این سرویس‌ها برای ذخیره در مخزن اصلی چارچوب بوده است. از اینرو انجام ترکیب سرویس برای ارزیابی روش پیشنهادی با سناریوی کاربردی، نیاز به برخی شبیه‌سازی‌های سرویس‌ها با توجه به منابع موجود، برای تکمیل مخزن اصلی داشت که این کار برای سناریوهای کاربردی معرفی شده در فصل ۵ انجام شد.

با توجه به اینکه تمرکز اصلی این تحقیق ساخت سرویس مرکب است اما فراگیر بودن متدلوژی پیشنهادی برای فاز ساخت به سایر فازهای توسعه ترکیب سرویس نیاز به ارائه چارچوبی جامع و توصیف تمامی فازهای توسعه را در پی داشت، لذا افزایش حیطه و گستردگی تحقیق، منجر به عدم داشتن زمان کافی برای پیاده‌سازی ابزار مورد نیاز برای چارچوب و روش پیشنهادی گردید و سناریوی کاربردی به صورت دستی با استفاده از فازها و شبه کد الگوریتم‌های معرفی شده، انجام شد.

۵.۶ نتیجه‌گیری

با توجه به مطالب بخش ۲.۶ می‌توان نتیجه گرفت روش MDCHeS تمامی اهداف مورد نظر در این پایان نامه را برآورده می‌سازد. این روش به صورت نیمه خودکار ترکیب سرویس‌های وب را انجام می‌دهد. این روش با استفاده از دانش معمار در مراحلی که نیاز به تعامل با کاربر است بهترین انتخاب‌های ممکن را از طریق بازخوردهای مناسب کاربر، انتخاب می‌کند.

علاوه بر این، روش MDCHeS با استفاده از این دیدگاه فازبندی شده و مدل‌های معرفی شده، ترکیب انواع سرویس‌های وب را با کنترل پیچیدگی‌های ممکن در روال ترکیب انجام می‌دهد. مخزن اصلی و مخزن مدل‌ها در چارچوب پیشنهادی، پس از مدتی به پایگاه‌های داده جامعی از سرویس‌های وب مرکب و اتمیک و ویژگی‌های مورد نیاز آن‌ها برای روش ارائه شده تبدیل خواهد شد و به مرور زمان، به درخواست سرویس مرکب با درصد موفقیت بالاتری با استفاده از سرویس‌های موجود پاسخ داده می‌شود. همچنین قابلیت ترکیب بازگشتی در روش ارائه شده به پاسخگویی هر چه بیشتر به نیازمندی درخواستی کمک خواهد کرد.

۶.۶ کارهای آینده

جهت‌های پژوهشی مختلفی برای بهبود چارچوب پیشنهادی و یا تکمیل و توسعه آن وجود دارد که در این بخش به عنوان کارهای آینده معرفی خواهد شد. به طور کلی کارهای آینده این تحقیق را می‌توان به سه دسته زیر تقسیم کرد:

- دسته اول مربوط به بخش‌هایی است که در این تحقیق مورد تمرکز قرار گرفته و کارهای آینده پیشنهادی منجر به بهبود این بخش‌ها خواهد شد، این موارد عبارتند از :
  - ارائه مکانیزمی خودکار به عنوان پیش‌نیازهای شروع فرآیند توسعه ترکیب سرویس، به منظور وارد کردن خودکار اطلاعات مورد نیاز سرویس‌های اتمیک اعلان شده به مخزن چارچوب بر اساس ساختار معرفی شده بدون نیاز به عامل انسانی.
  - پیاده‌سازی ابزار مرتبط با مؤلفه‌های معرفی شده در معماری چارچوب پیشنهادی.
  - بهبود الگوریتم‌های مربوط به فاز کشف فعالیت برای بالا بردن دقت نتایج حاصل از تطبیق.
  - بهبود الگوریتم‌های مربوط به فاز انتخاب سرویس نهایی با ارائه دقیق فرمول برای محاسبه فاکتور ارزیابی.
  - پشتیبانی از ویژگی‌های کیفی بیشتر برای سرویس درخواستی کاربر و سرویس‌های اعلان شده توسط تأمین کنندگان برای انتخاب سازگارتر با نیازمندی درخواستی.
  - ارزیابی فرآیند توسعه ترکیب بر اساس روش‌های فرمال[۱].
  - اضافه کردم مکانیزمی به منظور بروز رسانی خودکار ویژگی‌های سرویس‌های موجود در مخزن و همچنین بروز رسانی مجدد سرویس‌های مرکب ایجاد شده بر اساس ویژگی‌های تغییر یافته سرویس‌های موجود در مخزن (مثلاً مجدداً روال ترکیب انجام شود).

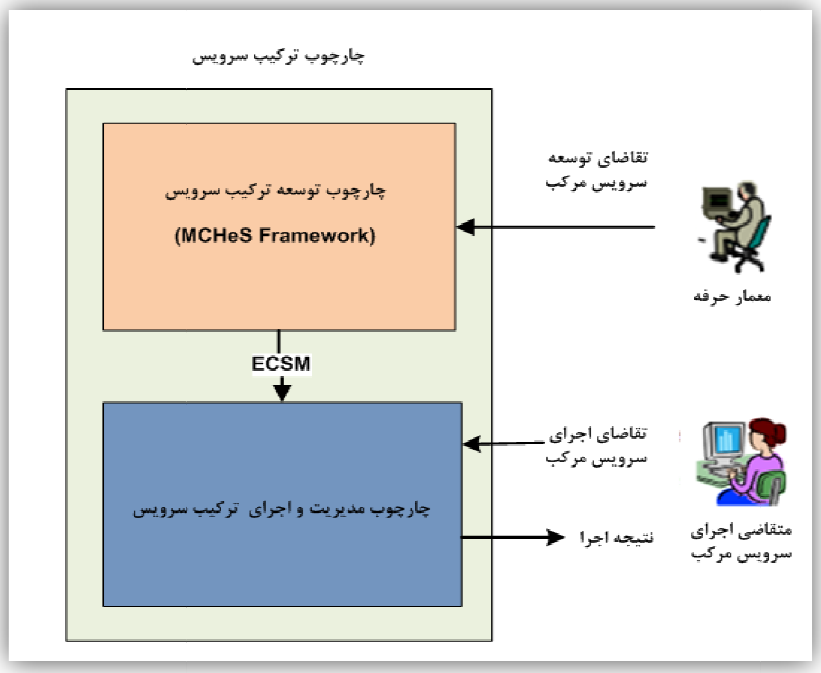
- دسته دوم مربوط به بخش‌هایی از چارچوب است که به صورت پیشنهادات مختصری در این تحقیق اشاره شده‌اند اما از آنجا که در محدوده تمرکز این تحقیق نبوده‌اند نیازمند تکمیل شدن هستند. نظیر موارد مطرح شده در زیر:

---

[۱] Fromal Methods

- توسعه مؤلفه‌ای به منظور افزودن قابلیت دریافت ورودی به زبان سطح بالاتر به طور مثال جملات زبان طبیعی به چارچوب با هدف کاربر پسند تر کردن چارچوب MDCHeS.
- ارائه الگوریتم تبدیل خودکار CCSM به ECSM برای فاز ساخت سرویس مرکب قابل اجرا.

- دسته سوم از کارهای آینده، با هدف جامع‌تر کردن روال ترکیب سرویس است. همان‌طور که در فصل اول مطرح گردید، ترکیب سرویس را می‌توان به دو مورد مجزا شامل توسعه ترکیب سرویس، و مدیریت و اجرای سرویس مرکب تقسیم نمود. در این تحقیق تنها به مورد اول پرداخته شد و چارچوب پیشنهادی در صورت تکمیل شدن، پاسخگویی مناسبی برای این مورد است. اما مورد دوم می‌تواند جهت تحقیقاتی مناسبی برای کارهای آینده این تحقیق به شمار آید. همان‌طور که در شکل ۶-۱ مشاهده می‌شود، این مورد نیز می‌تواند با پیشنهاد چارچوبی برای مدیریت و اجرای ترکیب سرویس و استفاده از خروجی چارچوب MDCHeS (که در این تحقیق پیشنهاد شد) پاسخ داده شود و با تکیه بر قابلیت‌های دیدگاه استفاده شده در چارچوب MDCHeS منجر به یک راه‌حل صنعتی و کامل برای استفاده در سازمان‌های کشور به منظور ترکیب سرویس، گردد.

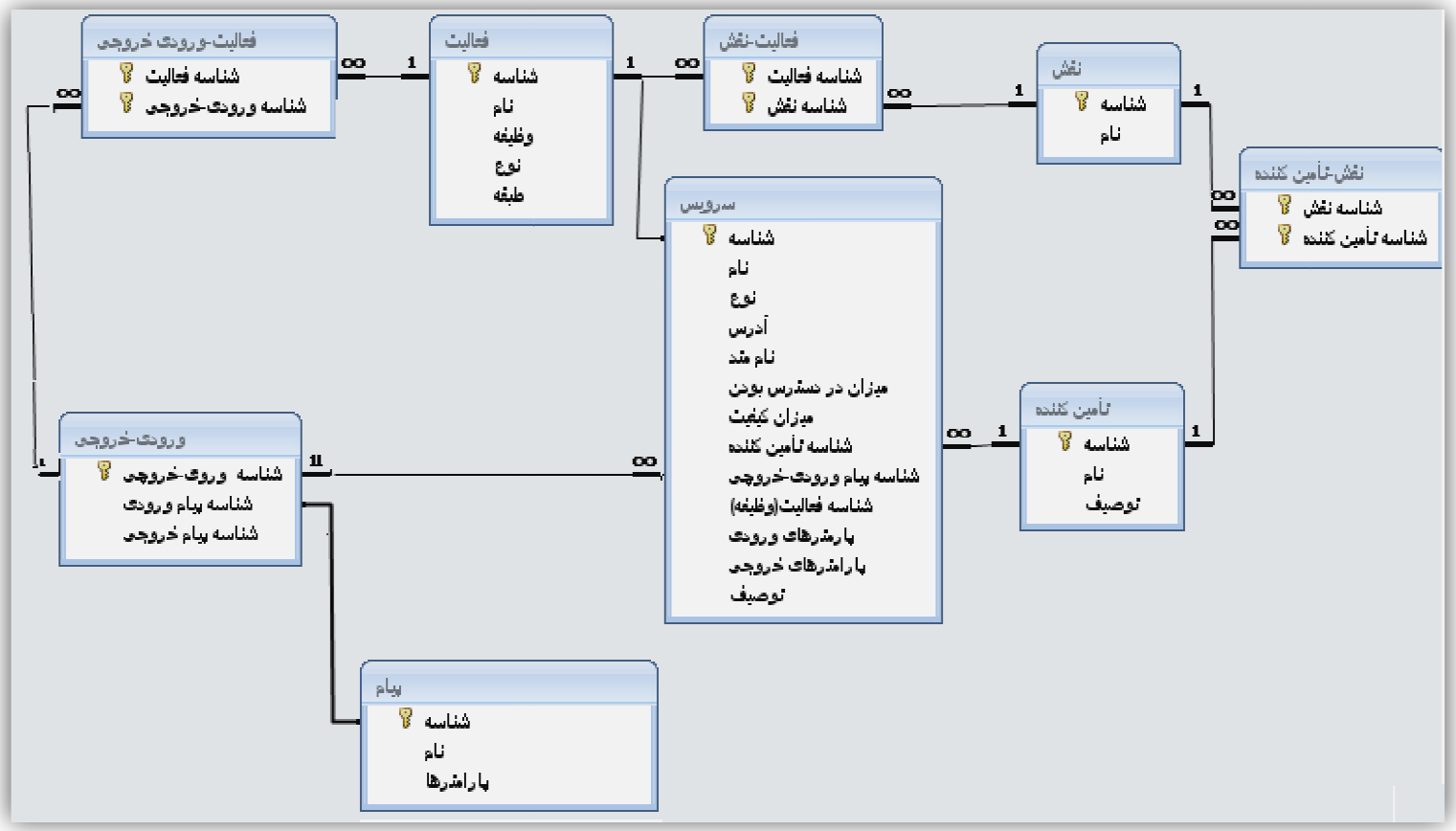


شکل ۶-۱: چارچوب جامع توسعه و مدیریت ترکیب سرویس

۷ پیوست ۱ – عناصر مخزن اصلی چارچوب MDCHeS

مقدمه

برای انجام دقیق‌تر و جامع‌تر سناریوهای کاربردی استفاده شده در این تحقیق، مخزن اصلی چارچوب در پایگاه داده Access ایجاد شد و عناصر هر سناریوی در این پایگاه داده ذخیره گردید که این عناصر در جدول ۷-۱ آورده شده‌اند. در شکل ۷-۱ شمایی از ارتباط کلاس‌های ترکیب (جداول پایگاه داده) نمایش داده شده است. لازم به ذکر است که کلاس‌هایی که در شکل ۷-۱ علاوه بر عناصر ترکیب معرفی شده در مدل متاداده ای وجود دارد به منظور نمایش ارتباط چند به چند (n به m) سایر کلاس‌ها در نظر گرفته شده‌اند، به طور مثال کلاس فعالیت-نقش، نگه‌دارنده ارتباط چند به چند دو جدول نقش و فعالیت است. جدول ۷-۱ نیز عناصر موجود در مخزن اصلی را به تفکیک هر کلاس نمایش می‌دهد. این عناصر برای پاسخگویی به دو سناریوی"برنامه ریزی سفر" و "استخدام نیرو در سازمان" ایجاد شده‌اند.



شکل ۷-۱: شمای ارتباط کلاس‌های ترکیب در پایگاه داده Access

جدول ۷-۱: عناصر موجود در مخزن اصلی چارچوب برای انجام سناریوهای کاربردی

عناصر کلاس فعالیت					
شناسه	نام	وظیفه	نوع	طبقه	شناسه نقش
A-01	HotelReservation	Hotelbooking	Atomic	Hotels	R-04
A-02	EventRegistration	EventRegistering	Atomic	Travel & Tourism	R-01
A-03	PlaneReservation	PlaneBooking	Atomic	Airlines	R-02
A-04	CarReservation	CarRenting	Atomic	Automobiles & Parts	R-06
A-05	ShuttleReservation	ShuttleBooking	Atomic	Hotels	R-05
A-06	WeatherGetting	CityWeatherGetting	Atomic	Internet	R-07
A-07	WeatherForcasting	CityWeatherForcasting	Atomic	Internet	R-07
A-08	CityGeographicalGetting	CityGeographicalGetting	Atomic	Internet	R-08 , R-12
A-09	HotelShuttleReservation	HotelShuttleBooking	Composite	Hotels	R-09
A-11	HotelCarReservation	HotelCarBooking	Composite	Hotels	R-11
A-13	WeatherForcasting	CityWeatherForcasting	Composite	Internet	R-07
A-19	GetOpenPosition	EnterpriseOpenPositionGetting	Atomic	Software & Computer Services	R-14
A-20	GetEnterpriseResume	GetResume	Atomic	Software & Computer Services	R-14
A-21	GetOnlineResume	GetResume	Atomic	Internet	R-14
A-22	**ShowAddressInMap**	**ShowAddress**	**Atomic**	**Internet**	**R-15**

عناصر کلاس تأمین کننده		
شناسه	نام	توصیف
P-01	MahanAir	MahanAirlineCompany
P-02	AsemanAir	InternationalAsemanAirlineCompany
P-03	GoldCar	Gold Car Rental Company
P-04	LuxeCar	LuxeCar Rental Company
P-05	GoTrip	GoTripHotel
P-06	DreanHotel	DreanHotel
P-08	InternationalEyes	InternationalEventRegistrationCompany
P-09	WeatherOnline	WeatherForcastingSystem
P-10	Oklahoma	OklahomaWeatherForecastCompany
P-11	OnileGeographical	CityGeographical
P-12	YourShuttle	Shuttle and Hotel Reservation Provider
P-14	MDCHeS	MDCHeS-Framework
P-17	PA-Company	PeykasaProvider
P-18	LinkedIn	OnlineResumeProvider
P-19	**Google**	**GoogleMap**

فعالیت- ورودی خروجی	
شناسه ورودی-خروجی	شناسه فعالیت
IO-01	**A-01**
IO-02	**A-01**
IO-03	**A-02**
IO-04	**A-02**
IO-05	**A-03**
IO-06	**A-03**
IO-07	**A-04**
IO-08	**A-04**
IO-09	**A-05**
IO-10	**A-06**
IO-11	**A-06**
IO-12	**A-07**
IO-14	**A-08**
IO-15	**A-09**
IO-16	**A-09**
IO-17	**A-09**
IO-18	**A-11**
IO-13	**A-13**
IO-25	**A-19**
IO-26	**A-20**
IO-26	**A-21**
IO-27	**A-22**

عناصر کلاس پیام		
شناسه	نام	پارامترها
M-01	HotelBookingInputMsg	String hotelName; Date dateArrive; Date dateBack
M-02	EventRegisteringInputMsg	String event_name, Date start_registr_date
M-03	EventRegisteringInputMsg2	String event_name, Date start_registr_date, String duration
M-04	PlaneBookingInputMsg	String origination, String to, Date departeDate, Number passenger
M-05	PlaneBookingInputMsg2	String from, String to, Date departDate, Date returnDate; Boolean journey type;
M-06	CarBookingInputMsg	String pickupCity, Date pickupDate, String dropoffLocation, Date dropoffDate
M-07	CarBookingInputMsg2	string depature_address, Date date_leave, String arrive_address, Date date_back
M-08	ShuttleBookingInputMsg	String originCity,String DestinationCity,Date date
M-10	WeatherForcastingInputMsg	Num citylatitude, Num citylongitude, Date startdate, String duration
M-11	CityMsg	String cityName
M-13	HotelBookingInputMsg2	String hotel_city, Date date_arrive, Date date_back
M-14	HottlShuttleBookingInputMsg	String name_hotel, String hotel_city, Date date_arrive, Date date_back
M-15	HottlShuttleBookingInputMsg2	String name_hotel, String hotel_city, Date date_arrive, Date date_back , String originCity, Date destinationCity date, time, String shuttle_Type
M-16	HottlShuttleBookingInputMsg3	String hotel_name, String hotel_city, Date date_arrive, String originCity
M-18	HotelCarBookingInputMsg	String hotel_name, String hotel_city, Date date_arrive, Date date_back, String dropOffLocation, Date drop

عناصر کلاس نقش	
شناسه	نام
R-01	EventRegisterationAgent
R-02	Airline
R-04	HotelAgent
R-05	ShuttleAgent
R-06	CarRentingAgent
R-07	WeatherAgent
R-08	GeographicalInfoAgent
R-09	HottleShuttleAgent
R-11	HotelCarAgent
R-12	WeatherForcastingAgent
R-14	JobProfileRule
R-15	**AddressingRule**

		OffDate
M-19	HotelBookingOutputMsg	String hotel Booking _ID
M-20	EventRegisteringOutputMsg	String EventParticipant _ID
M-21	PlaneBookingOutputMsg	Num seatNo, Num flightNo, String flightBooking_ID
M-22	CarBookingOutputMsg	String carBooking _ID, String pickupAddress
M-23	ShuttleBookingOutputMsg	String ShuttleBooking _ID
M-24	WeatherGettingOutputMsg	String weatherOfCity
M-25	WeatherForcastingOutputMsg	String forcasted WeatherOfCity
M-26	CityGeographicalGettingMsg	Num citylatitude, Num citylongitude
M-27	HottlShuttleBookingOutputMsg	String hotel Booking _ID, String shuttleBooking _ID
M-28	HotelCarBookingOutputMsg	String carBooking _ID, String hotel Booking _ID
M-39	GetOpenPositionOutputMsg	String Job, String educationLevel, String experienceYear
M-40	GetResumeInputMsg	String jobCategory, string positionType, Address address, String education
M-41	GetResumeOutputMsg	String name, String email, Address address, String educationLevel
M-42	MapShowAddressInputMsg	Address address
M-43	**MapShowAddressOutputMsg**	**Image Map**

ورودی-خروجی		
شناسه ورودی-خروجی	**شناسه پیام ورودی**	**شناسه پیام خروجی**
IO-01	M-01	M-19
IO-02	M-13	M-19
IO-03	M-02	M-20
IO-04	M-03	M-20
IO-05	M-04	M-21
IO-06	M-05	M-21
IO-07	M-06	M-22
IO-08	M-07	M-22
IO-09	M-08	M-23
IO-10	M-26	M-24
IO-11	M-11	M-24
IO-12	M-10	M-25
IO-13	M-11	M-25
IO-14	M-11	M-26
IO-15	M-14	M-27
IO-16	M-15	M-27
IO-17	M-16	M-27
IO-18	M-18	M-28
IO-18	M-18	M-28
IO-25	-	M-39
IO-26	M-41	M-42
IO-27	M-43	M-44

عناصر کلاس سرویس											
شناسه	نام	نوع	آدرس	نام متد	میزان کیفیت	میزان در دسترس بودن	پارامترهای ورودی	پارامترهای خروجی	آدرس فایل توصیف	شناسه فعالیت	شناسه تأمین کننده
S-01	MahanAirBooking	SOAP-based	http://	Booking	Average	High	String from, String to, Date departDate, Date returnDate; Boolean journey type; String cabin	Num seatNo, Num flightNo, String flightBooking_ID	http://	A-03	P-01
S-02	AsemanAir Booking	SOAP-based	http://	Booking	High	Average	String origination, String to, Date departeDate, Number passenger	Num seatNo, Num flightNo, String flightBooking_ID	http://	A-03	P-01
S-03	Gold CarRenting	RESTful	http://	POST	Average	High	String pickupCity, Date pickupDate, String dropoffLocation, Date dropoffDate	String carBooking _ID, String pickupAddress	http://	A-04	P-03
S-04	LuxeCar Renting	RESTful	http://	POST	High	Average	string depature_address, Date date_leave, String arrive_address, Date date_back	String carBooking _ID, String pickupAddress	http://	A-04	P-04
S-05	GoTrip ShuttleBooking	SOAP-based	http://	Shuttle Booking	Average	Low	String originCity,String DestinationCity,Date date	String ShuttleBooking _ID	http://	A-05	P-05
S-06	GoTrip HotelBooking	SOAP-based	http://	Hotel Booking	Average	Average	String hotelName; String hotelCity; Date dateArrive; Date dateBack	String hotel Booking _ID	http://	A-01	P-05
S-07	DreanHotelBooking	SOAP-based	http://	Booking	High	High	String hotel_city, Date date_arrive, Date date_back	String hotel Booking _ID	http://	A-01	P-06
S-08	EventRegistering1	RESTful	http://	POST	High	High	String event_name, Date start_registr_date, Date end_registr_date	String EventParticipant _ID	http://	A-02	P-08
S-09	EventRegistering2	RESTful	http://	POST	Average	Average	String event_name, Date start_registr_date, String duration	String EventParticipant _ID	http://	A-02	P-08
S-10	WeatherOnlineForcasting	SOAP-based	http://	Weather Forcasting	High	High	Num citylatitude, Num citylongitude, Date startdate, String duration	String forcasted WeatherOfCity	http://	A-07	P-09
S-11	WeatherOnlineGetting	RESTful	http://	GET	Average	Average	Num citylatitude, Num citylongitude	String weatherOfCity	http://	A-06	P-09
S-12	OklahomaWeatherGetting	RESTful	http://	GET	High	Average	String cityName	String weatherOfCity	http://	A-06	P-10
S-13	OnlineCityGeogra	RESTful	http://	GET	High	High	String cityName	Num citylatitude, Num	http://	A-08	P-11

	phicalGetting							citylongitude			
S-14	GoTrip-HotelShuttle Booking	SOAP-based	http://	HottleShuttleBooking	Average	Average	String name_hotel, String hotel_city, Date date_arrive, Date date_back	String hotel Booking _ID, String shuttleBooking _ID	http://	A-09	P-05
S-15	Dream-HotelShuttleBooking	SOAP-based	http://	HottleShuttleBooking	High	High	String name_hotel, String hotel_city, Date date_arrive, Date date_back , String originCity, Date destinationCity date, time, String shuttle_Type	String hotel Booking _ID, String shuttleBooking _ID	http://	A-09	P-06
S-16	MDCHeS-HotelShuttleBooking	SOAP-based	http://	HottleShuttleBooking	Average	Low	String hotel_name, String hotel_city, Date date_arrive, String originCity	String hotel Booking _ID, String shuttleBooking _ID	http://	A-09	P-14
S-17	MDCHeS-Weather Forcasting	RESTful	http://	GET	High	Average	String cityName	String forcasted WeatherOfCity	http://	A-13	P-14
S-18	MDCHeS-HotelCarBooking	SOAP-based	http://	HotelCar Booking	High	Average	String hotel_name, String hotel_city, Date date_arrive, Date date_back, String dropOffLocation, Date drop OffDate	String carBooking _ID, String hotel Booking _ID	http://	A-11	P-14
S-28	FindEnterprise OpenPositions	SOAP-based		FindOpen Position	Average	Average	-	String id, String positionType, String JobCategory, String educationLevel	http://www.peykasa.ir/Service/FindOpenPosition.ws	A-19	P-17
S-29	FindEnterprise Resume	SOAP-based		FindResume	High	High	String jobCategory, string positionType, Address address, String education	String id, String name, EmailAddress email, Address address	http://www.peykasa.ir/Service/FindOpenPosition.wsd	A-20	P-17
S-30	GetOnlineResume	RESTful	http//linkedIn.com/Services/OnlineResume/[Position	POST	Average	High	String jobCategory, string positionType, Address address, String education	String id, String name, EmailAddress email, Address address		A-21	P-18
S-31	GoogleMap	RESTful	http://maps.google.com/Service/ShowAddresess	POST	High	High	Address address	Image Map		A-22	P-19

۸ پیوست ۲ – جدول کلمات مخفف مورد استفاده در الگوریتم‌ها

جدول ۸-۱ : فهرست کلمات مخفف مورد استفاده در الگوریتم‌های معرفی شده در فصل ۴

مخفف	توصیف لاتین	توصیف فارسی
F_i	Functionality(i)	وظیفه‌مندی درخواستی کاربر شماره i
A_i	Activity(i)	فعالیت شماره i
$A_i.IO$	Related Input and Output of activity(i)	ورودی-خروجی متناظر با فعالیت شماره i
$F_i.IO$	Category of Functionality(i)	طبقه استاندارد وظیفه‌مندی شماره i
S_{Fi}	Set of proper graph (contained one or several activities) and IO(s) related to Functionality(i)	مجموعه‌ای از گراف‌ها شامل یک فعالیت یا ترتیبی از فعالیت‌های متناظر با وظیفه‌مندی شماره i به همراه ورودی-خروجی‌ها
$S_{ACSM/CCSM}$	Set of ACSM/ CCSM	مجموعه‌ای متشکل از ACSM/CCSM
$F_i/A_i.name$	The name of Functionality/Activity(i)	نام وظیفه‌مندی / فعالیت شماره i
$F_i/A_i.category$	The standard category of Functionality/Activity(i)	طبقه استاندارد وظیفه‌مندی / فعالیت شماره i
$S_{Approved}$	Set of activities that Approved by user for a Functionality	مجموعه فعالیت‌های تأیید شده توسط کاربر برای یک وظیفه‌مندی
$S_{NotApproved}$	Set of activities that NotApproved by user for a Functionality	مجموعه فعالیت‌های تأیید نشده توسط کاربر برای یک وظیفه‌مندی
$S_{Activities}$	Set of Activities	مجموعه‌ای از فعالیت‌ها
$S_{FunctionalityNames}$	Set of WordNet similar words for a Functionality name	مجموعه‌ای از نام‌های مشابه از نظر WordNet برای نام یک وظیفه‌مندی
IM.FD	Functionality Definition of IM	اطلاعات وظیفه‌مندی‌ها در مدل ورودی
IM.FC	Functionality Constraints of IM	محدودیت وظیفه‌مندی‌ها در مدل ورودی
IM.GC	Global Constraints of IM	محدودیت‌های عمومی وظیفه‌مندی درخواستی در مدل ورودی
IM.DCG	Data Control Graph of IM	گراف داده‌ای-کنترلی بر اساس وظیفه‌مندی‌ها و پارامترهای ورودی-خروجی در مدل ورودی
ACSM.AD	Activity Definition of ACSM	اطلاعات فعالیت‌ها در مدل انتزاعی سرویس مرکب
ACSM.GC	Global Constraints of ACSM	محدودیت‌های عمومی فعالیت درخواستی در مدل انتزاعی سرویس مرکب

ACSM.DCG	Data Control Graph of ACSM	گراف داده‌ای- کنترلی بر اساس فعالیت‌ها، نقش‌ها و پارامترهای ورودی-خروجی در مدل انتزاعی سرویس مرکب
CCSM.SD	Service Definition of CCSM	اطلاعات سرویس‌ها در مدل واقعی سرویس مرکب
CCSM.GC	Global Constraints of CCSM	محدودیت‌های عمومی سرویس درخواستی در مدل واقعی سرویس مرکب
CCSM.DCG	Data Control Graph of CCSM	گراف داده‌ای- کنترلی بر اساس سرویس‌ها، تأمین‌کنندگان در مدل واقعی سرویس مرکب

۹ پیوست ۳ – متن کامل نظرسنجی قابلیت‌های چارچوب MDCHeS

نظرسنجی قابلیت‌های چارچوب مدل‌رانه ترکیب انواع سرویس‌های وب

تهیه کننده: سوده فرخی

استاد راهنما: دکتر فریدون شمس

۱. مقدمه

این مستند با هدف ارزیابی بخشی از کار پایان نامه‌ای با عنوان "ارائه روشی به منظور نیمه خود کارسازی مدل‌رانه ترکیب سرویس‌های وب" تهیه شده است. بخش مورد نظر چارچوبی نیمه خودکار جهت ترکیب مدل‌رانه سرویس‌های وب است که به عنوان پایان‌نامه کارشناسی ارشد انجام شده است. اعضای محترم گروه تحقیقاتی ASER مخاطب این نظرسنجی‌اند. در این مستند ابتدا توصیف مختصری از چارچوب MDCHeS[۱] و مؤلفه‌های آن آورده شده است و سپس فرآیند توسعه ترکیب سرویس معرفی می‌گردد. لازم به ذکر است که توصیف چارچوب و فرآیند توسعه برای آن دسته از مخاطبانی است که با روال ترکیب در این چارچوب آشنا نیستند. در بخش پایانی ۱۶ سؤال در مورد قابلیت‌های چارچوب پیشنهادی مطرح می‌گردد. افرادی که در طی جلسات گروه ASER با چارچوب MDCHeS و روال توسعه در آن آشنا هستند، نیازی به مرور چارچوب و فرآیند توسعه ترکیب سرویس در آن ندارند، و تنها کافی است به ۱۶ سؤال پرسش‌نامه پاسخ دهند.

پیشاپیش از همکاری اساتید محترم و دانش‌پژوهان گرامی تشکر می‌کنم.

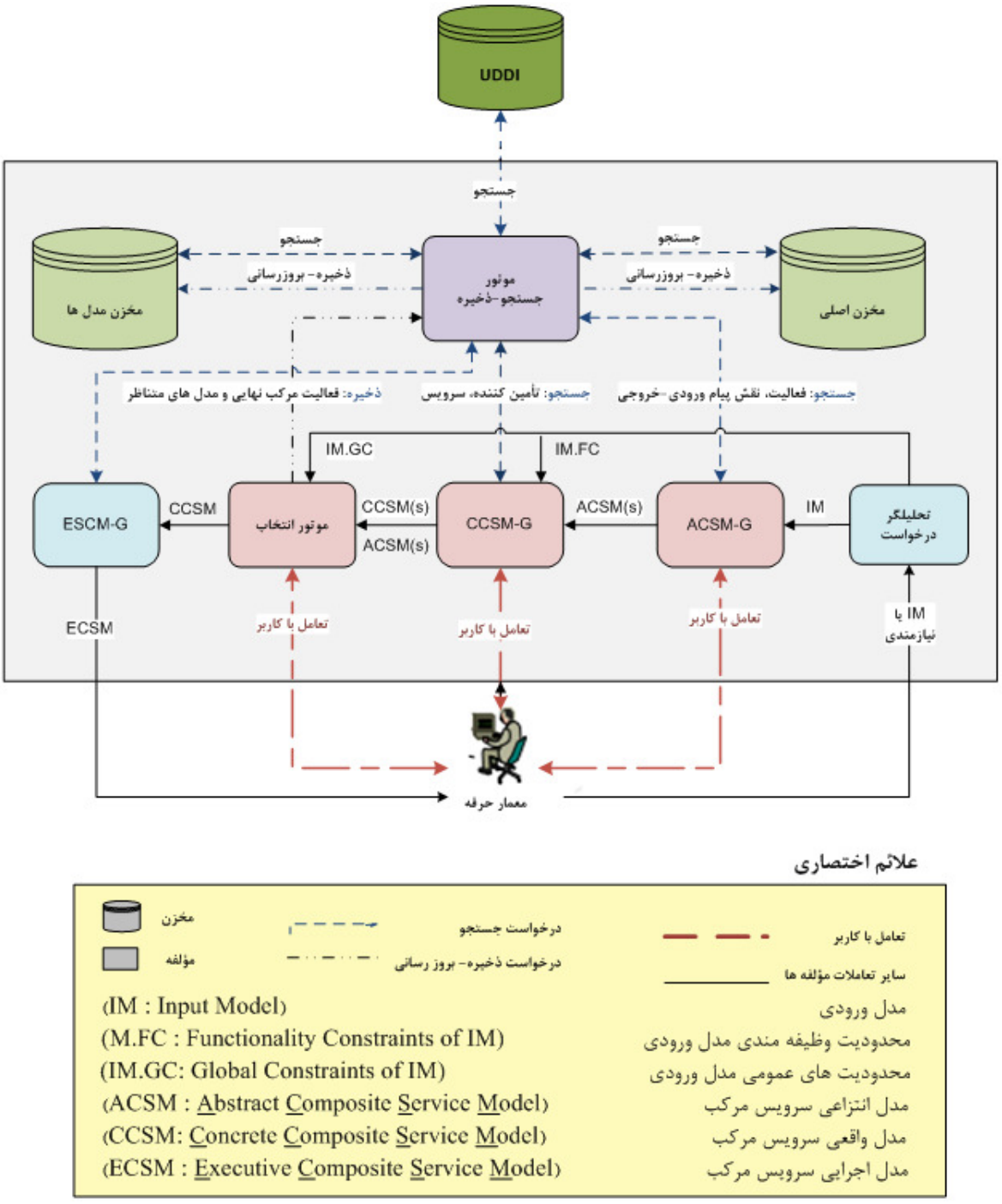
سوده فرخی - دی ماه ۸۹

---

[۱] Model driven Composition of Heterogeneous Service

۲. معماری چارچوب MDCHeS

در شکل۱ معماری چارچوب پیشنهادی نشان داده شده است و در ادامه عناصر این معماری و وظیفـه‌منـدی هـر کـدام معرفی می‌گردند.



"شکل ۱: معماری چارچوب"

۳. وظیفه‌مندی عناصر معماری چارچوب

- **مؤلفه تحلیل‌گر درخواست**: این مؤلفه نیازمندی‌های کاربر را به عنوان ورودی دریافت می‌کند و پس از تحلیل و تأیید، در خروجی آن‌ها را به صورت IM به مؤلفه *ACSM-G* می‌دهد.
- **مخزن اصلی**: همه عناصر ترکیب در *مخزن اصلی* چارچوب بر اساس ساختار مدل متاداده ای که در شکل ۲ نمایش داده‌شده است، ذخیره شده‌اند. لازم است *مخزن اصلی* بر اساس سرویس‌های اعلان شده در UDDI با مکانیزمی (خودکار یا دستی توسط معمار) بر اساس ساختار مدل متاداده ای قبل از شروع فرآیند توسعه سرویس مرکب، تکمیل شود، همچنین از آنجا که ویژگی‌های سرویس‌های اعلان شده در وب قابل تغییر است، نیاز به مکانیزمی برای بروز رسانی دوره‌ای ویژگی‌های سرویس‌های ذخیره شده در مخزن چارچوب است. سرویس‌های مرکبی که در این چارچوب توسعه داده می‌شوند نیز در پایان فرآیند توسعه و پس از اجرای آن به عنوان فعالیتی مرکب، به مخزن اصلی چارچوب جهت استفاده مجدد، افزوده خواهند شد. دسترسی به *مخزن اصلی* تنها از طریق مؤلفه *موتور جستجو-ذخیره* امکان پذیر خواهد بود.
- **مخزن مدل‌ها**: این مخزن در برگیرنده مدل‌های مرتبط با سرویس مرکب درخواستی است که شامل همه مدل‌های (IM، ACSM, CCSM, ECSM) شرکت‌کننده در فرآیند ایجاد یک سرویس مرکب هستند. از آنجا که سرویس‌های مرکب توسعه داده شده توسط چارچوب، در پایان به عنوان فعالیت مرکب به *مخزن اصلی* چارچوب افزوده می‌شوند، در این صورت لازم است مدل‌های مرتبط با توسعه آن نیز ذخیره گردند تا در صورتی که نیاز به استفاده مجدد از این فعالیت شد، اطلاعات مربوط به محدودیت‌ها و ویژگی‌های کیفی درخواستی کاربر با شرایطی که سرویس طبق آن توسعه داده شده، مقایسه گردد.
- **مؤلفه موتور جستجو_ ذخیره**: این مؤلفه وظیفه ارتباط با *مخزن اصلی* و *مخزن مدل‌ها* در چارچوب را بر عهده دارد. کلیه تراکنش‌های مورد نیاز سایر مؤلفه‌های چارچوب با هر یک از مخازن چارچوب به وسیله درخواست به این مؤلفه انجام می‌شود. این مؤلفه یکی از مؤلفه‌های اصلی چارچوب MDCHeS محسوب می‌شود. وظیفه‌مندی‌های این مؤلفه عبارتند از:
  - درخواست مؤلفه *ACSM-G* به منظور جستجوی فعالیت، نقش و پیام‌های ورودی_خروجی متناسب با وظیفه‌مندی درخواستی کاربر، توسط این مؤلفه انجام می‌شود. همچنین در صورتی که فعالیت مرکبی

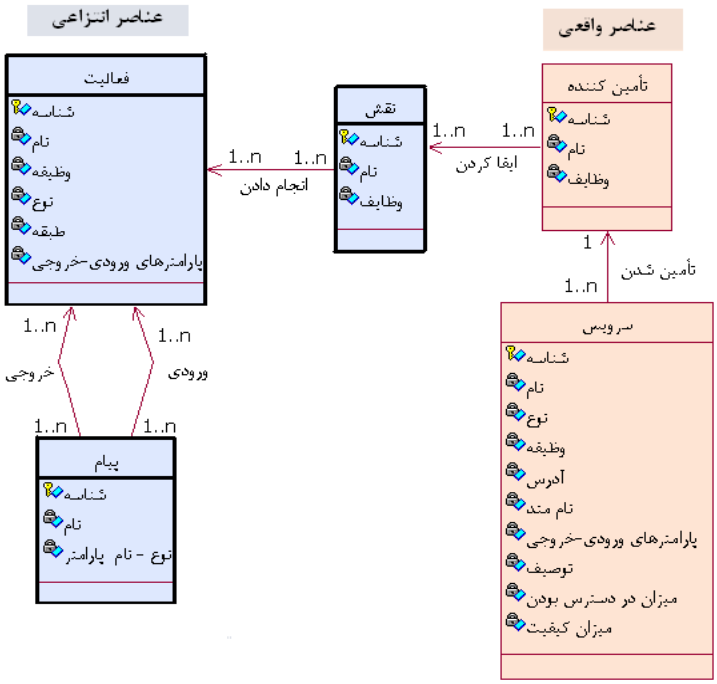
برای یک وظیفه‌مندی یافت شود، از طریق ارتباط این مؤلفه با *مخزن مدل‌ها* کلیه مدل‌های مورد نیاز برای فرآیند توسعه استخراج می‌شود.

- مؤلفه *CCSM-G* درخواست جستجوی تأمین‌کننده و سرویس متناظر با هر فعالیت، نقش و پیام ورودی- خروجی را به این مؤلفه داده و نتیجه را دریافت می‌کند.
- مؤلفه *موتور انتخاب* نیز ذخیره مدل‌های مرتبط با سرویس مرکب توسعه داده شده را با ارتباط با این مؤلفه در *مخزن مدل‌ها* ذخیره می‌کند.
- همچنین این مؤلفه به منظور تکمیل و بروز رسانی ویژگی‌های عناصر موجود در *مخزن اصلی* و همچنین افزودن سرویس‌های جدید به *مخزن اصلی* چارچوب، با UDDI در ارتباط است.

- **مؤلفه ACSM-G**: این مؤلفه با ارتباط با مؤلفه *موتور جستجو-ذخیره* نیازمندی‌های مربوط به ایجاد مدل‌های ACSM مرتبط با IM که به عنوان ورودی دریافت کرده است را استخراج می‌کند. خروجی این مؤلفه یک یا بیش از یک ACSM است. این مؤلفه برای تأیید فعالیت‌های بدست آمده به ازای هر فعالیت و انتخاب مدل‌های ACSM با کاربر تعامل می‌کند. همچنین در صورتی که به ازای یک وظیفه‌مندی فعالیت مناسبی در جستجو کشف نشود، این مؤلفه وظیفه ساخت فعالیت مرکبی که پاسخ‌گویی این وظیفه‌مندی باشد را بر عهده دارد. تعاملات لازم با مخزن اصلی در این رابطه نیز از طریق مؤلفه *موتور جستجو-ذخیره* انجام می‌شود.
- **مؤلفه CCSM-G**: این مؤلفه وظیفه تبدیل مدل‌های ACSM ورودی به مدل‌های CCSM متناظر را با ارتباط با مؤلفه *موتور جستجو-ذخیره* بر عهده دارد. در این ارتباط عناصر تأمین‌کننده و سرویس متناظر با هر فعالیت بدست می‌آید. همچنین این مؤلفه به عنوان ورودی، محدودیت‌های درخواستی کاربر مرتبط با هر وظیفه‌مندی اولیه را نیز دریافت می‌کند تا درخواست جستجو عناصر نام برده را با توجه به ترجیحات کاربر انجام دهد. این ترجیحات می‌تواند محدود کردن با تأمین کننده خاص و یا سطح خاصی برای ویژگی کیفی سرویس و یا سایر محدودیت‌های پشتیبانی شده در چارچوب باشند. این مؤلفه نیز به منظور تأیید مدل‌های CCSM تولید شده با کاربر تعامل دارد.
- **مؤلفه موتور انتخاب**: این مؤلفه بر اساس ورودی‌های دریافتی و الگوریتم انتخاب تعریف شده در چارچوب با محاسبه شاخص ارزیابی برای هر CCSM بر اساس بخش محدودیت عمومی از مدل ورودی، میزان ویژگی‌های کیفی در بخش محدودیت عمومی از هر مدل انتزاعی سرویس مرکب و همچنین نوع سرویس‌ها در بخش تعریف

سرویس‌ها از هر مدل واقعی سرویس مرکب، شاخص ارزیابی را برای هر CCSM محاسبه کرده و با تکمیل این پارامتر در بخش محدودیت عمومی از هر مدل واقعی سرویس مرکب، مدل‌های CCSM کاندیدا را بر اساس این شاخص مرتب می‌کند. سپس از بین مدل‌های CCSMکاندیدا با تعامل با کاربر، مناسب‌ترین گزینه برای پاسخ‌گویی به سرویس درخواستی کاربر را انتخاب و به عنوان خروجی به مؤلفه *ECSM-G* تحویل می‌دهد. این مؤلفه در پایان مرحله انتخاب کلیه مدل‌های مرتبط با مدل انتخابی شامل IM، ACSM، CCSM برای پاسخگویی به این سرویس درخواستی را به منظور استفاده مجدد از طریق مؤلفه *موتور جستجو*ذخیره در *مخزن مدل‌ها* ذخیره می‌کند.

- **<u>مؤلفه ECSM-G</u>**: این مؤلفه CCSM ورودی را به زبان BPEL for REST که زبانی قابل اجرا برای سرویس مرکب درخواستی است تبدیل کرده و به منظور اجرا و مدیریت سرویس مرکب به کاربر تحویل می‌دهد. خروجی این مؤلفه به عنوان خروجی نهایی فرآیند توسعه ترکیب سرویس مرکب در چارچوب MDCHeS است.



"شکل ۲ : مدل متا داده‌ای"

۴. فرآیند توسعه در چارچوب MDCHeS

فرآیند توسعه ترکیب سرویس در چارچوب MDCHeS طبق ایده معماری مدل‌رانه از فازهایی برای تبدیل مدل‌های معرفی شده به یکدیگر تشکیل شده است؛ این فرآیند شامل پنج فاز است که عبارتند از:

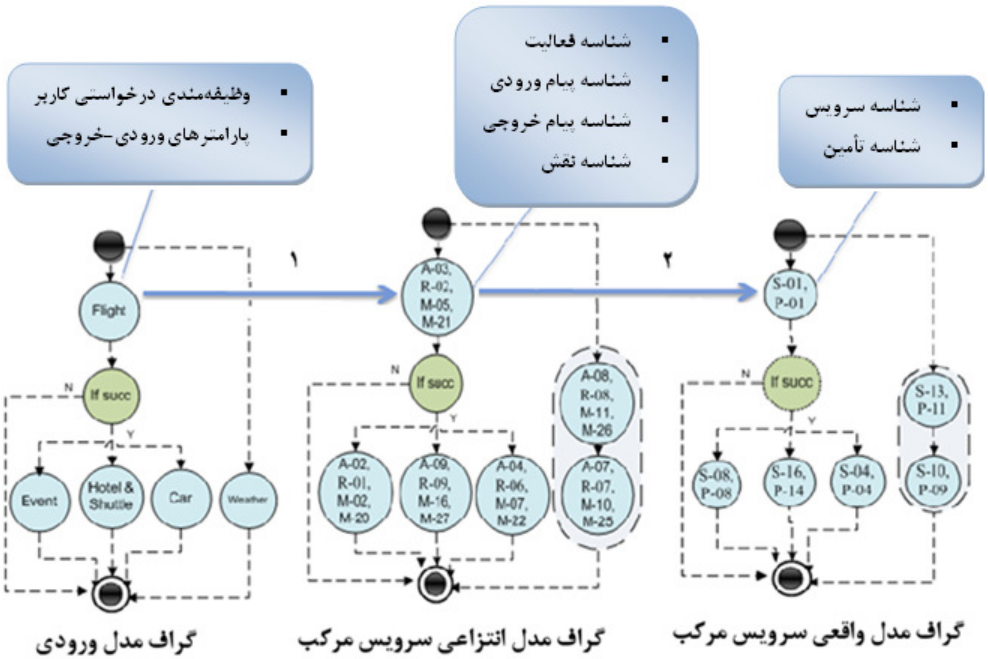
۱. توصیف سرویس مرکب درخواستی (تبدیل نیازمندی درخواستی کاربر در قالب مدل ورودی)
   - در شکل۳ نمونه‌ای از مدل ورودی برای سناریوی برنامه‌ریزی سفر نشان داده شده است. در صورتی که کاربر با الگوی مورد استفاده در مدل ورودی آشنا باشد، خود کاربر IM را ایجاد می‌کند.
۲. کشف فعالیت متناظر با وظیفه‌مندی درخواستی (تبدیل مدل ورودی به مدل(های) انتزاعی سرویس مرکب)
   - در صورت موفق نبودن الگوریتم جستجو ساخت فعالیت مرکب برای پاسخگویی به وظیفه‌مندی مورد نظر با استفاده از الگوریتم ترکیب فعالیت
   - تعامل با کاربر برای انتخاب فعالیت‌های مناسب از مجموعه فعالیت‌های کشف‌شده برای هر وظیفه‌مندی
   - ایجاد تمامی مدل‌های ممکن ACSM با مجموعه تأیید شده فعالیت‌های هر وظیفه‌مندی
۳. ساخت سرویس مرکب واقعی (تبدیل مدل انتزاعی سرویس مرکب به مدل واقعی سرویس مرکب)
   - کشف تأمین‌کننده و سرویس متناظر با فعالیت با در نظر گرفتن ترجیحات کاربر
   - محاسبه ویژگی کیفی هر مدل CCSM
   - تولید مدل‌های CCSM
   - تعامل با کاربر برای تأیید مدل‌های تولید شده
۴. انتخاب سرویس مرکب واقعی نهایی (انتخاب مدل واقعی سرویس مرکب نهایی)
   - محاسبه شاخص ارزیابی برای هر CCSM بر اساس ویژگی‌های کیفی، محدودیت‌های کاربر و همگن بودن نوع سرویس .
   - تکمیل پارامتر شاخص ارزیابی هر CCSM
   - مرتب‌سازی CCSMها بر اساس شاخص ارزیابی
   - در صورت تمایل کاربر، تعامل با کاربر برای انتخاب نهایی، در غیر این صورت انتخاب CCSM با بالاترین شاخص
   - ذخیره سازی IM، ACSMها، CCSMها در مخزن مدل‌ها به منظور استفاده‌ی مجدد.

۵. ساخت سرویس مرکب قابل اجرا (تبدیل مدل واقعی سرویس مرکب به مدل اجرایی سرویس مرکب)

- تبدیل به مدل اجرایی با استفاده از توسعه ارائه شده BPEL با نام BPEL for REST با دو قابلیت زیر:
  - توانایی فراخوانی مستقیم یک سرویس وب RESTful درون یک فرآیند BPEL
  - نمایش دادن بخشی از فرآیند اجرایی BPEL به عنوان یک منبع

۵. سناریوی کاربردی برنامه ریزی سفر

سناریوی **برنامه ریزی سفر** به شرح زیر است: " کاربر قصد برنامه‌ریزی کردن سفری به منظور شرکت در یک رویداد را دارد، این رویداد می‌تواند شرکت در یک کنفرانس، جلسه کاری و یا یک سفر توریستی باشد. بدین منظور او لازم است در این رویداد ثبت نام کرده و برای مکان و نحوه رفت و آمد برنامه ریزی کند. او همچنین نیاز به دانستن پیش بینی آب و هوای شهر مقصد در زمان حضور در آن رویداد، را دارد." در شکل ۳ تبدیل گراف‌های سه مدل معرفی شده در چارچوب پیشنهادی برای مثال برنامه‌ریزی سفر نشان داده شده است. فلش‌های نشان داده شده روال وظیفه‌مندی رزرو بلیط هواپیما تا یافتن سرویس متناظر را نشان می‌دهد. لازم به ذکر است که این مثال بر اساس عناصر موجود در مخزن اصلی چارچوب تکمیل گردیده است. همچنین در شکل ۴ مدل ورودی این سناریو نمایش داده شده است.



"شکل۳ : گراف‌های مربوط به سناریوی برنامه ریزی سفر"

Functionality definitions

//defining Functionality names & related category;

```
functionality     {planBooking,          *.Airlines;
functionality      hotelShuttleBooking,  *.Hotels;
functionality      eventRegistrating,    *.Travel & Tourism;
functionality      carBooking,           *.Automobiles & Parts;
functionality      weatherForcasting,    *.Internet;}
```

Functionality Constraints

// defining Input, Output and QoS of each Functionality;

```
constraint planBooking

input = { depaturePlace;
          dateLeave;
          arrivePlace;
          dateBack; }

output = { planBookingId; }
```

// only one functionality for instance illustrates here.

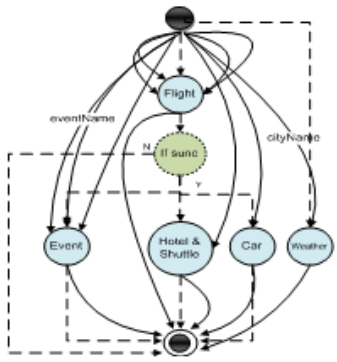
Global Constraints

// defining Boundary Input, Output and QoS of Requested composite functionality;

```
constraint global *.Travel & Tourism; {
input = {  startDate;
           finishDate;
           destination;
           source;
           eventName; }

output = {  planBooking_id;
            hotelBooking_id;
            shuttleBooking_id;
            carRental_id:
            carRental_id;
            weatherForcasted;
            carPickupAddress;}
qos = {
           quality = 2;
           availability = 2; }
```

Data & Control Specification



''شکل ۴: نمونه‌ای از مدل ورودی در سناریوی ''برنامه ریزی سفر''

۶. پرسش‌نامه

عدد ۱ به معنی کاملاً مخالف، عدد ۲ مخالف، عدد ۳ متوسط، عدد ۴ موافق و نهایتاً عدد ۵ کاملاً موافق است. لطفاً با انتخاب یکی از ۵ گزینه، پاسخ خود را مشخص کنید.

ردیف	قابلیت‌های چارچوب ترکیب سرویس‌های وب MDCHeS	۱	۲	۳	۴	۵
۱۷	کاربر درخواست‌کننده‌ی سرویس مرکب برای فرآیند توسعه ترکیب سرویس لازم نبست دانش فنی عمیقی داشته باشد.					
۱۸	تعامل با چارچوب در کلیه فازها، برای توسعه دهنده سرویس مرکب، ساده و شفاف است.					
۱۹	تعامل با کاربر تنها به منظور بهبود روال توسعه سرویس مرکب است.					
۲۰	چارچوب نیازمندی‌های عملکردی و غیر عملکردی سازمان برای ایجاد سرویس‌های مرکب درخواستی را لحاظ می‌کند.					
۲۱	چارچوب قابلیت استفاده از دارایی‌های سازمان و استفاده مجدد از سرویس‌های مرکب ایجاد شده در سازمان را دارد.					
۲۲	چارچوب قابلیت استفاده از انواع سرویس وب خارج از سازمان، در سرویس مرکب درخواستی را دارد.					
۲۳	چارچوب قالب مناسبی برای طرح نیازمندی در فاز توصیف سرویس مرکب درخواستی در نظر گرفته است.					
۲۴	چارچوب فاز کشف فعالیت را به طور مناسب توسعه داده است.					
۲۵	در صورت عدم کشف سرویس متناظر، چارچوب قابلیت ترکیب عناصر موجود به منظور پاسخ‌گویی به نیاز درخواستی را دارا می‌باشد.					
۲۶	چارچوب فاز ساخت سرویس مرکب واقعی (CCSM) را به طور مناسب توسعه داده است.					
۲۷	چارچوب فاز انتخاب سرویس نهایی را بر اساس شاخص‌های مناسبی توسعه داده است.					
۲۸	چارچوب از فناوری مناسبی برای فاز ساخت سرویس مرکب قابل اجرا (ECSM) استفاده کرده است.					
۲۹	استفاده از چارچوب فرآیند توسعه سرویس مرکب را تسهیل می‌بخشد.					
۳۰	استفاده از چارچوب در سازمان‌ها عملیاتی است.					
۳۱	چارچوب قابلیت تطبیق با تغییرات محیط سرویس‌های وب و پویایی ترکیب را دارا می‌باشد.					
۳۲	درصد پاسخگویی موفق به نیازمندی درخواستی با استفاده مداوم از چارچوب به دلیل تکمیل مخزن اصلی، بیشتر می‌شود.					

منابع تحقیق

[1] B. Orriens, J. Yang, and M.P. Papazoglou, “Model driven service composition,” *Service-Oriented Computing-ICSOC 2003*, 2003, pp. 75–90.

[2] F. Shams and P. Jamshidi, *Proposal of ASOCF, Technical Report, Shahid Beheshti University*, *Automated Software Engineering Research Group* .http://aser.sbu.ac.ir, 2010.

[3] S. Farokhi, *A Servey on web service composition, Seminar Report, Shahid Beheshti University*, 2010.

[4] “Oasis: SOA Adoption Blueprint” Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-blueprints.

[5] B. Borges, K. Holley, and A. Arsanjani, “Service Oriented Architecture,” *IBM* Available: http://searchsoa.techtarget.com/news/1006206/Service-oriented-architecture.

[6] J.H.D. Santos, “Public service improvment using runtime service composition strategies , M.S. Thesis, University of Twente, Enschede the Netherlands,” *Revista Democracia Digital e Governo Eletrônico*, vol. 2, 2010.

[7] “XML,” *Wikipedia* Available: http://en.wikipedia.org/wiki/XML.

[8] “Web service,” *Wikipedia* Available: http://en.wikipedia.org/wiki/Web_service.

[9] X. Liu, Y. Hui, W. Sun, and H. Liang, “Towards service composition based on mashup,” *Services, 2007 IEEE Congress on*, 2007, pp. 332–339.

[10] P. Leitner, “The Daios Framework-Dynamic, Asynchronous and Message-oriented Invocation of Web Services,” 2007.

[11] “Web Services Description Language,” *Wikipedia* Available: http://en.wikipedia.org/wiki/Web_Services_Description_Language.

[12] “Web Service Description Language,” *W3C (World Wide Web Consortium)* Available: http://www.w3.org/TR/wsdl.

[13] "SOAP," *IBM* Available: http://publib.boulder.ibm.com/infocenter/cicsts/v3r1/index.jsp?topic=/com.ibm.cics.ts31.doc/dfhws/concepts/soap/dfhws_message.htm.

[14] "SOAP," *Wikipedia* Available: http://en.wikipedia.org/wiki/SOAP.

[15] "Representational State Transfer," *Wikipedia* Available: http://en.wikipedia.org/wiki/Representational_State_Transfer.

[16] R.L. Costello, "Building Web Services the REST Way" Available: http://www.xfront.com/REST-Web-Services.html.

[17] H. Zhao and P. Doshi, "Towards Automated RESTful Web Service Composition," *2009 IEEE International Conference on Web Services*, 2009, pp. 189-196.

[18] A. Khalili, "Semi automatic Creation of Enterprise Mashups Using Semantic Descriptions," MS Thesis. Khaje Nasir Toosi University, 2009.

[19] H. Zhao, "Scalable Composition of Web Services Under Uncertainty, PHD Thesis,Georgia University" Available: http://ugakr.libs.uga.edu/handle/10724/11685.

[20] C. Pautasso, "On Composing RESTful Services," *IEEE Internet Computing*, 2008, pp. 24-31.

[21] C. Pautasso, "Composing restful services with JOpera," *Software Composition*, Springer, 2009, pp. 142–159.

[22] C. Pautasso, "Bpel for rest," *Business Process Management, Springer*, 2008, pp. 278–293.

[23] N. Milanovic and M. Malek, "Current solutions for web service composition," *IEEE Internet Computing*, 2004, p. 51–59.

[24] L. Zeng, A.H.H. Ngu, B. Benatallah, R. Podorozhny, and H. Lei, "Dynamic composition and optimization of Web services," *Distrib Parallel Databases Springer*, vol. 24, 2008, pp. 45-72.

[25] S.P. Sivasubramanian, E. Ilavarasan, and G. Vadivelou, “Dynamic Web Service Composition: Challenges and techniques,” *2009 International Conference on Intelligent Agent & Multi-Agent Systems*, 2009, pp. 1-8.

[26] “BPEL, Active Endpoints” Available: http://www.activevos.com/bpel.php.

[27] Y. Wang and K. Taylor, “A Model-Driven Approach to Service Composition,” *Service-Oriented System Engineering, 2008. SOSE’08. IEEE International Symposium on*, IEEE, 2008, pp. 8–13.

[28] J.D. Haan, “Model Driven Architecture, basic concepts” Available: http://www.theenterprisearchitect.eu/archive/2008/01/16/mda_model_driven_architecture.

[29] A. Kleppe, J. Warmer, and W. Bast, *MDA Explained: The Model Driven Architecture™: Practice and Promise*, 1997.

[30] C. Zhao, Z. Duan, and M. Zhang, “A Model-Driven Approach for Dynamic Web Service Composition,” *WRI World Congress on Software Engineering*, 2009, pp. 273-277.

[31] P.P.W. Chan and M.R. Lyu, “Dynamic Web Service Composition: A New Approach in Building Reliable Web Service,” *22nd International Conference on Advanced Information Networking and Applications*, Ieee, 2008, pp. 20-25.

[32] F. Casati, S. Ilnicki, L.J. Jin, V. Krishnamoorthy, and M.C. Shan, “Adaptive and dynamic service composition in eFlow,” *Advanced Information Systems Engineering*, Springer, 2000, pp. 13–31.

[33] M.D.A.M. Benatallah, B., “SELF-SERV: a platform for rapid composition of web services in a peer-to-peer environment,” *28th VLDB Conference.*, Hong Kong, China. 2002.

[34] E.A. Zeng, L., “Quality Driven Web Services Composition,” *Proceedings of the 12th international conference on World Wide Web, ACM*, 2003.

[35] P.F. Pires, *WEBTRANSACT: A Framework for Specifying and coordinating reliable web services compositions. Technical Report ES-578/02, Federal University of Rio De Janerio*, 2002.

[36] J. Aggarwal, R., Verma, K., Miller, J., Milnor, *Dynamic Web Service Composition in METEOR-S. Technical report, LSDIS Lab, Univerisity of Georgia, Athens*, 2004.

[37] D. Roman, "Web Service Modeling Ontology," *Applied Ontologies*, 2005, pp. 77-106.

[38] T. Fujii, K., Suda, "Semantics-based dynamicWeb service composition," *International Journal of Cooperative Information Systems*, 2006, pp. 293–324.

[39] A.V.H. Pathak, J., S. Basu, "Modeling Web Services by Iterative Reformulation of Functional and Nonfunctional Requirements," *4th International Conference on Service-Oriented Computing (ICSOC)*, 2006, pp. 314-326.

[40] S. Topouzidou, *SODIUM, Service-Oriented Development In a Unified framework in Final report ISTFP 6-004559.*, 2007.

[41] E.G. da Silva, F. Pires, and M.V. Sinderen, "Supporting dynamic service composition at runtime based on end-user requirements," *Knowledge Creation Diffusion Utilization*, 2009.

[42] Y.-Y. Peng, M. Shang-Pin, and J. Lee, "REST2SOAP: A Framework To Integrate SOAP Services And RESTful," *IEEE International Conference on Service-Oriented Computing and Applications (SOCA' 09)*, 2009, pp. 106-109.

[43] K. Boumhamdi and Z. Jarir, "Yet another approach for dynamic web service composition," *Internet Technology and Secured Transactions*, London: 2010, pp. 1 - 5.

[44] C. Pautasso, "RESTful Web service composition with BPEL for REST," *Data & Knowledge Engineering Journal, ISSN: 0169-023X*, vol. 68, Sep. 2009, pp. 851-866.

[45] F. Casati, S. Ilnicki, L. Jin, V. Krishnamoorthy, and M. Shan, "eFlow: a platform for developing and managing composite e-services," 2000, pp. HPL-2000-36.

[46] K. He, “Integration and orchestration of heterogeneous services,” *Pervasive Computing (JCPC), 2009 Joint Conferences on*, IEEE, 2009, pp. 467–470.

[47] J. Lim and K.-H. Lee, “Constructing composite web services from natural language requests,” *Journal of Web Semantics: Science, Services and Agents on the World Wide Web, ISSN: 1570-8268*, vol. 8, Mar. 2010, pp. 1-13.

[48] F. Rosenberg, P. Leitner, A. Michlmayr, and S. Dustdar, “Integrated Metadata Support for Web Service Runtimes,” *Enterprise Distributed Object Computing Conference Workshops, 2008 12th*, IEEE, 2009, pp. 361–368.

[49] M.H. Rick Hull, Bell Labs, “Usage Scenario: e-Service Composition in a Behavior based Framework” Available: http://www.ai.sri.com/~daml/services/use-cases/language/swsl-usecase/Composition.htm.

[50] L. Liu, Z. Wu, Z. Ma, and W. Wei, “Functionality Semantic Indexing and Matching Method for RESTful Web Services Based on Resource State Descriptions,” *2009 WRI World Congress on Software Engineering*, May. 2009, pp. 138-142.

[51] W. Junye, M. Lirui, and C. Hongming, “A REST-Based Approach to Integrate Enterprise Resources,” *International Forum on Computer Science-Technology and Applications*, Ieee, 2009, pp. 219-223.

[52] M.A. Talib, Z. Yang, and Q.M. Ilyas, “A framework towards web services composition modelling and execution,” *International Journal of Web and Grid Services*, vol. 2, 2006, pp. 25.

[53] “Standard Industrial Classification,” *Wikipedia* Available: http://en.wikipedia.org/wiki/Industry_Classification_Benchmark.

[54] “Data Flow Diagram,” *Wikipedia* Available: http://en.wikipedia.org/wiki/Data_Flow_Diagram.

[55] “Control flow graph,” *Wikipedia* Available: http://en.wikipedia.org/wiki/Control_flow_graph.

[56] J. Pathak, S. Basu, R. Lutz, and V. Honavar, "MOSCOE: An Approach For Composing Web Services Through Iterative Reformulation Of Functional Specifications," *International Journal on Artificial Intelligence Tools*, vol. 17, 2008, pp. 109-122.

[57] A. Kim, M. Kang, C. Meadows, E. Ioup, and J. Sample, "A Framework for Automatic Web Service Composition," *(CHACS), Naval Research Lab Washington DC Center For High Assurance Computing Systems*, 2009.

[58] J. Rao and X. Su, "A survey of automated web service composition methods," *Semantic Web Services and Web Process Composition Springer*, 2005, pp. 43–54.

[59] F. Rosenberg, P. Leitner, A. Michlmayr, P. Celikovic, and S. Dustdar, "Towards Composition as a Service - A Quality of Service Driven Approach," *2009 IEEE 25th International Conference on Data Engineering*, pp. 1733-1740.

[60] R. Karimpour and F. Taghiyareh, "Conceptual discovery of Web services using WordNet," *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific*, IEEE, 2010, pp. 440–444.

[61] "Word Net" Available: http://wordnet.princeton.edu/wordnet/.

[62] S. Dustdar and W. Schreiner, "A survey on web services composition," *International Journal of Web and Grid Services*, vol. 1, 2005, pp. 1–30.

[63] R.M. Pessoa, E. Silva, M. van Sinderen, D.A.C. Quartel, and L.F. Pires, "Enterprise interoperability with SOA: a survey of service composition approaches," *Enterprise Distributed Object Computing Conference Workshops, 2008 12th*, IEEE, 2009, pp. 238–251.

[64] R. Khadka and B. Sapkota, "An Evaluation of Dynamic Web Service Composition Approaches," *4th International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing - ACT4SOC 2010*, 2010, pp. 67-79.

[65] Y. Peng, "Design and Implementation of a Framework to Integrate SOAP Services and RESTful Services, MS Thesis, Computer Science and Information Engineering China," 2009.

Abstract

Frequent changes in customer requirements and business environments are the main challenges in developing ultra-large-scaled software systems. Service-Oriented Architecture (SOA) has beeb proposed as an applicable solution to overcome these challenges. On the other hand, service composition is one of the central and most important tenets of service-oriented computing. Web Service Composition (WSC) provides an opportunity for enterprises to increase their ability to adapt to frequent changes in user demands via integrating existing services.

Lack of supporting for the simultaneous composition of SOAP-based and RESTful Web service in a composite service, inadequate coverage of service composition phases and non-consideration of requested non-functional requirements for service composition development and inadequate use of enterprise assets along with extra Web services in composition methods are the shortcomings of current service composition methods and approaches.

Hence, the main goal of this thesis is semi-automate the process of Web service composition based on user requested functional and non-functional requirements by using the idea of the Model Driven Architecture in the form of introducing a framework, composition process phases and the models used in each phase. The proposed method has tried to support RESTful and SOAP-based web services by using a Meta data model for describing these kinds of services simultaneously. To this aim, we have studied relevant existing methods and have tried to address their shortcomings as well as utilize their strengths. Therefore, we propose a method called “Model-driven Dynamic Composition of Heterogeneous Service” (MDCHeS) and the related framework along with relevant composition development phases.

Keywords. Service-Oriented Architecture, Web Service Composition, RESTful Web Service, SOAP-based Web Service, Model Driven Architecture.

Shahid Beheshti University
Department of Electrical & Computer Engineering

Semi-Automated Model Driven Web Service Composition

Supervisor:
Dr. Fereidoon Shams

A THESIS SUBMITTED
FOR THE DEGREE OF
MASTER OF SCIENCE

For Partial Fulfillment in MSc Degree in Shahid Beheshti University

By
Soodeh Farokhi

Advisor:
Dr. Fereidoon Shams

January 2011