بسم الله الرحمن الرحیم

دانشگاه شهید بهشتی
دانشکده مهندسی برق و کامپیوتر

ارائه یک مدل تصمیم‌گیری برای انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی خودتطبیق

پایان‌نامه کارشناسی ارشد مهندسی فناوری اطلاعات
گرایش معماری سازمانی

حمید رضا اقیری

استاد راهنما:
دکتر اسلام ناظمی

۱۳۹۲

دانشگاه شهید بهشتی

دانشکده مهندسی برق و کامپیوتر

پایان‌نامه کارشناسی ارشد مهندسی فناوری اطلاعات

گرایش معماری سازمانی

تحت عنوان:

ارائه یک مدل تصمیم‌گیری برای انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی خودتطبیق

در تاریخ پایان نامه دانشجو، حمیدرضا اقیری، توسط کمیته تخصصی داوران مورد بررسی و تصویب نهائی قرار گرفت.

۱- استاد راهنما اول: آقای دکتر اسلام ناظمی امضاء

۲- استاد داور (داخلی): آقای دکتر حسن حقیقی امضاء

۳- استاد داور (خارجی): آقای دکتر جعفر حبیبی امضاء

۴- نماینده‌ی تحصیلات تکمیلی: آقای/ خانم دکتر امضاء

با سپاس و تقدیر از تمامی اساتید، خبرگان و دوستانی که در تدوین این گزارش من را همراهی نمودند، به ویژه استاد ارجمند، دکتر اسلام ناظمی که راهنمایی‌های موجز و دقیق ایشان، راه‌گشایی کلیدی بود.

کلیه حقوق مادی مترتب بر نتایج مطالعات،

ابتکارات و نوآوری‌های ناشی از تحقیق موضوع

این پایان نامه متعلق به دانشگاه شهید بهشتی

می باشد.

به نام خدا

نام ونام خانوادگی: حمیدرضا اقیری
عنوان پایان نامه: ارائه یک مدل تصمیم‌گیری برای انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی خودتطبیق
استاد راهنما: دکتر اسلام ناظمی

اینجانب حمیدرضا اقیری تهیه کننده پایان نامه کارشناسی ارشد حاضر خود را ملزم به حفظ امانت داری و قدردانی از زحمات سایر محققین و نویسندگان بنا بر قانون Copyright می دانم. بدین وسیله اعلام می نمایم که مسئولیت کلیه مطالب درج شده با اینجانب می باشد و در صورت استفاده از اشکال؛ جداول، و مطالب سایر منابع، بلافاصله مرجع آن ذکر شده و سایر مطالب از کار تحقیقاتی اینجانب استخراج گشته است و امانتداری را به صورت کامل رعایت نموده ام. در صورتی که خلاف این مطلب ثابت شود، مسئولیت کلیه عواقب قانونی با شخص اینجانب می باشد.

نام و نام خانوادگی دانشجو:

امضاء و تاریخ:

فهرست مطالب

۱- فصل اول: کلیات ۱

۱-۱- مقدمه ۲

۱-۲- هدف ۴

۱-۳- تعریف مسئله ۴

۱-۴- مراحل پژوهش ۴

۱-۵- محدودیت‌های پژوهش ۵

۱-۶- ساختار تحقیق ۵

۲- فصل دوم: ادبیات پژوهش ۷

۲-۱- مقدمه ۸

۲-۲- تعاریف و مفاهیم اولیه‌ی موضوع ۸

۲-۲-۱- چارچوب نرم‌افزاری ۸

۲-۲-۲- معماری نرم‌افزار ۸

۲-۲-۳- پردازش خودمختار ۸

۲-۲-۴- سامانه خودمدیریت ۹

۲-۲-۵- حلقه‌ی کنترل ۱۱

۲-۲-۶- مؤلفه خودتطبیق ۱۲

۲-۲-۷- عنصر مدیریت شونده ۱۳

۲-۲-۸- حسگر ۱۳

۲-۲-۹- عملگر ۱۳

۲-۳- قوانین و سیاست‌های تطبیق ۱۳

۲-۴- طیف‌های خودتطبیقی ۱۴

۲-۴-۱- گرایش به مهندسی نرم‌افزار و نظریه کنترل ۱۴
۲-۴-۲- روش‌های داخلی و خارجی ۱۵
۲-۵- مدل تصمیم‌گیری ۱۶
۲-۵-۱- فرآیند تصمیم‌گیری ۱۶
۲-۵-۲- دسته‌بندی مدل‌های تصمیم‌گیری ۱۸
۲-۵-۳- تصمیم‌گیری گروهی ۱۹
۲-۶- سامانه‌های سازمانی ۱۹
۲-۷- کارهای مشابه ۲۱
۲-۸- نتیجه‌گیری ۲۳

۳- فصل سوم: چارچوب‌ها ۲۴

۳-۱- مقدمه ۲۵
۳-۲- طبقه‌بندی چارچوب‌ها ۲۵
۳-۳- چارچارچوب گارلان و دیگران (Ranibow) ۲۶
۳-۳-۱- طرح کلی چارچوب ۲۸
۳-۴- چارچوب جورجس و تیلور ۲۹
۳-۴-۱- طرح کلی چارچوب ۳۰
۳-۴-۲- سیاست‌های انطباق در پایگاهدانش ۳۲
۳-۴-۳- پاسخهای تطبیق ۳۳
۳-۵- چارچوب AOA ۳۳
۳-۵-۱- طرح کلّی چارچوب ۳۳
۳-۶- چارچوب DSCB ۳۶
۳-۶-۱- طرح کلی چارچوب ۳۷
۳-۶-۲- تطبیق در گسترهی خوشه ۳۹

۳-۶-۳- تطبیق در گسترهی سامانه ۴۰
۳-۶-۴- حلقههای کنترل ۴۰
۳-۷- چارچوب MOCAS ۴۲
۳-۷-۱- طرح کلی چارچوب ۴۲
۳-۷-۲- مدل مؤلفهها ۴۳
۳-۷-۳- حلقهی کنترل در MOCAS ۴۶
۳-۸- چارچوب کریشنامورتی و بابو ۴۹
۳-۸-۱- طرح کلی چارچوب ۴۹
۳-۹- چارچوب یانگ و همکاران ۵۰
۳-۹-۱- طرح کلی چارچوب ۵۱
۳-۱۰- نتیجهگیری ۵۳
۴- فصل چهارم: مدل پیشنهادی ۵۴
۴-۱- مقدمه ۵۵
۴-۲- صفتیابی چارچوبها ۵۵
۴-۲-۱- روش صفتیابی ۵۶
۴-۲-۲- نتایج صفتیابی چارچوبها ۶۱
۴-۲-۳- خلاصهی نتایج صفتیابی ۶۶
۴-۳- تصمیمگیرندگان ۶۶
۴-۴- معیارها ۶۷
۴-۵- فرآیند ۷۰
۴-۶- روش تصمیمگیری ۷۲
۴-۷- مطالعهی موردی آزمایشگاهی ۱ ..... ۷۸
۴-۸- مطالعهی موردی آزمایشگاهی ۲ ..... ۸۴

۵- فصل پنجم: جمع‌بندی ۹۰

۵-۱- مقدمه ۹۱

۵-۲- خلاصه‌ی پژوهش ۹۱

۵-۳- نتیجه‌گیری ۹۲

۵-۴- مخاطبان و موارد استفاده ۹۳

۵-۵- کارهای آتی ۹۳

پیوست ۱: روش‌های تصمیم‌گیری چند شاخصه ۹۵

مراجع ۱۰۳

فهرست اشکال

شکل ۱-۱: ساختار و اجزای کلی یک سامانه‌ی خودتطبیق [۴] ۳
شکل ۲-۱: طیف مهندسی نرم‌افزار و نظریه کنترل ۱۴
شکل ۲-۲: طیف روش‌های داخلی و خارجی ۱۶
شکل ۳-۱: مثال مدل معماری مشتری – خدمتگذار [۳۷] ۲۷
شکل ۳-۲: طرح کلی چارچوب RAINBOW [۳۷] ۲۸
شکل ۳-۳: اجزا و جریان اطلاعات در چارچوب جرجس و تیلور [۳۶] ۳۱
شکل ۳-۴: عناصر اصلی چارچوب AOA [۳۷] ۳۴
شکل ۳-۵: خوشه‌ها و فدراسیون [۴۰] ۳۹
شکل ۳-۶: حلقه‌ی کنترل چارچوب DSCB [۱۲] ۴۱
شکل ۳-۷: نگهدارنده‌ی MOCAS [۱۱] ۴۴
شکل ۳-۸: مؤلفه‌ی خودمختار MOCAS [۱۱] ۴۶
شکل ۳-۹: رفتار یک حسگر MOCAS [۱۱] ۴۷
شکل ۳-۱۰: رفتار ارزیاب MOCAS [۱۱] ۴۸
شکل ۳-۱۱: نمای کلی چارچوب کریشنامورتی و بابو [۵] ۵۰
شکل ۳-۱۲: نمای کلی چارچوب فازی سامانه خودتطبیق [۱] ۵۳
شکل ۴-۱: فرآیند تصمیم‌گیری انتخاب چارچوب ۷۱
شکل ۴-۲: بازه‌ی اهمیت معیارها ۷۸

فهرست جداول

جدول۳-۱: خلاصه‌ی نتایج صفت‌یابی چارچوب‌های سامانه‌های خودتطبیق ۶۶
جدول ۴-۱: معیارها و تصمیم‌گیرندگان مدل تصمیم‌گیری ۶۹
جدول ۴-۲: معیارهای حوزه‌ی کارفرما و توسعه‌دهنده ۷۱
جدول ۴-۳: مزایا و معایب روش‌های تصمیم‌گیری ۷۳
جدول ۴-۴: مشارکت‌کنندگان در نظرسنجی انتخاب روش تصمیم‌گیری ۷۴
جدول ۴-۵: مقایسه‌ی روش‌های تصمیم‌گیری. ۷۴
جدول ۴-۶: جزئیات مدل تصمیم‌گیری ۷۷

چکیده

سامانه‌های سازمانی، در بیشتر موارد در مقیاسی بزرگ توسعه می‌یابند و ممکن است نیازمندی‌هایی که پیشران توسعه‌ی آن‌ها بوده در دراز مدت تغییر یابد. دوره‌ی استفاده از این سامانه‌های سازمانی با توجه به ماهیت راهبردی بسیاری از آن‌ها برای سازمان‌ها، طولانی مدت بوده و در طول مدت فعالیت خود ممکن است با سایر سامانه‌ها یکپارچه شده و یا ارتباط برقرار نمایند. نکاتی مانند این، باعث افزایش نیاز به ایجاد تغییر در این سامانه‌ها و تطبیق آن‌ها با شرایط جدید می‌شود.

یکی از مهمترین راهکارهایی که امروزه برای مقابله با چنین مشکلی ارائه شده است، استفاده از سامانه‌های خودتطبیق می‌باشد. یک سامانه‌ی خودتطبیق می‌تواند رفتار و کارکردهای خود را در زمان اجرا و با توجه به اهداف از پیش تعریف شده از سوی کاربران تغییر دهد و بدین ترتیب با انعطاف‌پذیری بالاتر و صرف هزینه و تلاش کمتری اهداف مورد نظر کاربران خود را تأمین نماید [۱]. تاکنون چارچوب‌های مختلفی برای توسعه‌ی سامانه‌های خودتطبیق ارائه شده است که هرکدام با توجه به طبقه‌بندی‌های مختلف دارای مزایا، معایب و ویژگی‌های خاص خودشان می‌باشند؛ ولی هیچ‌کدام از این چارچوب‌ها دارای راهنمایی‌های مشخصی به منظور استفاده و بهره‌برداری نمی‌باشند. در این راستا سؤالات زیر قابل طرح می‌باشند:

۱- هر کدام از این چارچوب‌ها دارای چه ویژگی‌ها و مزایایی نسبت به سایر چارچوب‌ها می‌باشند؟
۲- سازمان‌های توسعه‌دهنده چگونه می‌توانند از میان انواع چارچوب‌های معرفی شده، چارچوب مناسبی را جهت بهره‌برداری انتخاب نمایند؟

در این پژوهش یک مدل تصمیم‌گیری جهت انتخاب چارچوب توسعه‌ی مناسب برای تولید سامانه‌های سازمانی خودتطبیق ارائه شده است. توسعه‌ی این مدل نیازمند تعیین صفات و ویژگی‌های چارچوب‌های توسعه‌ی نرم‌افزارهای خودتطبیق و در نهایت ارائه یک شیوه‌ی تصمیم‌گیری و انتخاب چارچوب - ازمیان چارچوب‌های رایج موجود - می‌باشد.

با استفاده از این مدل تصمیم‌گیری امکان مشارکت روش‌مند سازمان‌های بزرگ در فرآیند توسعه‌ی نرم‌افزار فراهم می‌شود. همچنین بکارگیری این مدل، امکان ارائه رتبه‌بندی چارچوب‌های مختلف را - برای استفاده در توسعه‌ی سامانه‌های سازمانی خودتطبیق - با توجه به شرایط و نیازهای گوناگون سازمانی و فنی فراهم آورده و فرآیند طراحی معماری را شتاب می‌بخشد.

در انتها مدل تصمیم‌گیری طراحی شده به کمک دو مطالعه‌ی موردی آزمایشگاهی مورد ارزیابی قرار گرفته است. ارزیابی انجام گرفته نشان می‌دهد که مدل پیشنهادی امکان رتبه‌بندی چارچوب‌های مختلف را با توجه به نظر تصمیم‌گیرندگان فراهم می‌آورد و همچنین از اعمال نظر مدیریتی کارفرما در خصوص عدم استفاده از چارچوب‌های مشخصی پشتیبانی می‌نماید.

واژه‌های کلیدی – سامانه‌های خودتطبیق، خودتطبیقی، چارچوب، چارچوب نرم‌افزاری، مدل تصمیم‌گیری

۱- فصل اول: کلیات

۱-۱- مقدمه

امروزه ابعاد سامانه‌های سازمانی چنان توسعه یافته است که امکان توسعه‌ی این سامانه‌ها با استفاده از روش‌های سنتی وجود ندارد؛ چرا که با گسترش ابعاد این سامانه‌ها پیچیدگی آن‌ها و تعداد افراد درگیر در فرآیند توسعه، افزایش می‌یابد. این بدان معناست که نمی‌توان بطور یکنواخت و متمرکز و با تیم‌های - یکپارچه این سامانه‌ها را توسعه داد. استفاده از روش‌های سنتی مهندسی نرم‌افزار در این سامانه‌ها باعث ایجاد سامانه‌هایی غیر منعطف و شکننده خواهد شد. در این‌جا شکننده ناظر بر این ایده است که با توجه به ماهیت پویای این سامانه‌ها و محیط اجرایی آن‌ها، نیاز به تغییر رفتار یا عملکرد چنین سامانه‌هایی به تناوب رخ می‌دهد و سامانه در حال اجرا را مرتباً دچار نقص می‌نماید. این پیچیدگی سامانه‌های رایانه‌ای به حدی رسیده است که نگهداری از آن‌ها در جهت فعال بودن و کار کردن، نیازمند تلاش انسانی بسیار زیادی است که در حال خارج شدن از کنترل می‌باشد [۲].

یکی از مهم‌ترین جنبه‌های مربوط به این سامانه‌ها، انطباق با شرایط جدید است. بنابراین یکی از اهداف نوین مهندسی نرم‌افزار طراحی سامانه‌هایی است که در صورت نیاز به تغییرات با سرعت و سادگی قابل تغییر باشند. از این روی تکنیک‌هایی مثل استفاده از زبان‌های توصیف مشخصه‌ها، زبان‌های برنامه‌نویسی سطح بالا، تحلیل و طراحی شی‌گرا و معماری سرویس‌گرا تلاش‌هایی در این راستا می‌باشند.

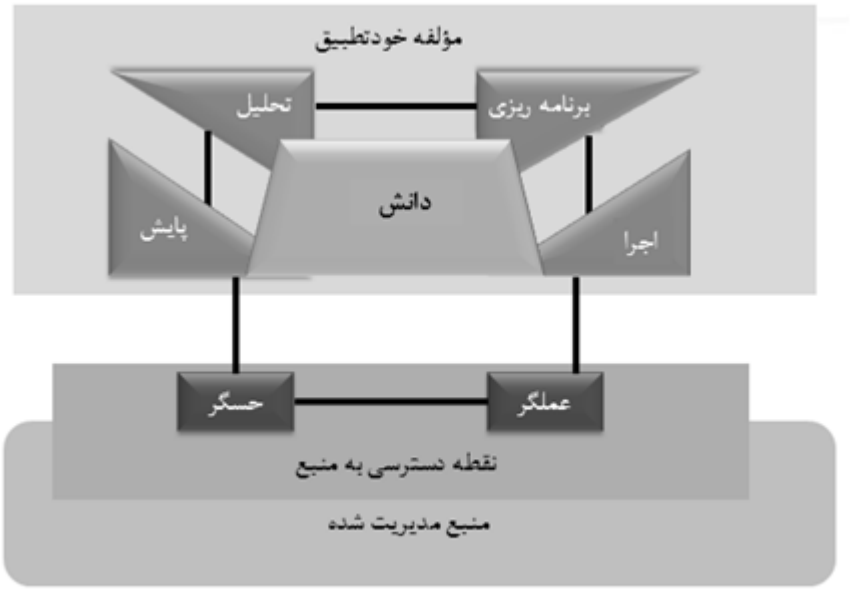
در سال‌های اخیر نرم‌افزارهای خودتطبیق به عنوان راه‌حلی مناسب برای مشکلات نگهداری سامانه مطرح شده‌اند [۳]. کاهش وابستگی شدید به منابع انسانی و واکنش سریع به رویدادهای مختلف از جمله مزایای سامانه‌های خودتطبیق به شمار می‌روند. این مشکلات می‌توانند طیف گسترده‌ای را از خطاهای داخلی زمان اجرا تا حملات خارجی و تغییر زمینه‌ی[۱] نرم‌افزار را دربر بگیرد. سامانه‌های خودتطبیق قادرند تا رفتار [۴] و پیکربندیشان را در پاسخ به تغییرات در محیط عملیاتی (هر آنچه که برای یک سامانه خودتطبیق قابل نظارت باشد مثل ورودی‌های کاربر و یا

---

[۱] Context

داده‌های دریافتی از حسگرها[۱] یا کاوشگرها[۲]) تغییر دهند. این تغییرات در پیکربندی می‌تواند شامل افزودن، حذف و یا جایگزینی مؤلفه، تغییر قوانین سامانه، تغییر در مدل استقرار برنامه و موارد مختلف دیگری باشد. تغییر در مدل استقرار تغییری در سامانه است که در آن ساختار منطقی برنامه ثابت مانده ولی نگاشت بین مؤلفه‌های نرم‌افزاری و سخت‌افزاری تغییر می‌یابد. این ویژگی نشان می‌دهد با بزرگ شدن مقیاس نرم‌افزارها تمرکز توسعه‌دهندگان از الگوریتم‌ها و ساختمان‌های داده به معماری‌ها [۵] و چارچوب‌ها منتقل شده است.

هر سامانه خودتطبیق به طور کلی از چهار بخش یا عنصر اصلی تشکیل شده است که اطلاعات یا دانش خاص تطبیق مانند زمان و ماهیت تطبیق سامانه را به اشتراک می‌گذارند که در (شکل ۱-۱: ساختار و اجزای کلی یک سامانه‌ی خودتطبیق) نشان داده شده‌اند. این دانش معمولاً توسط توسعه‌دهنده یا مدیر[۳] سامانه به عنوان سیاست‌های تطبیق سطح بالا تعیین می‌شود [۶].



شکل ۱-۱: ساختار و اجزای کلی یک سامانه‌ی خودتطبیق [۴]

---

[۱] Sensors
[۲] Probes
[۳] Administrator

اجزای مشخص شده در شکل در فصل آینده مورد بررسی قرار خواهند گرفت.

۱-۲- هدف

در این پژوهش سعی شده است تا چارچوب‌هایی که برای ایجاد قابلیت خودتطبیقی مورد استفاده قرار می‌گیرند از منظرهای متفاوت بررسی و تشریح شده و در انتها مدلی برای انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی خودتطبیق ارائه گردد. در این فصل مسئله، پرسش‌ها، اهداف و ساختار تحقیق معرفی می‌شوند.

۱-۳- تعریف مسئله

مسئله‌ی حاضر عبارت است از طراحی یک مدل تصمیم‌گیری برای انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی خودتطبیق. همانطور که پیشتر نیز مورد اشاره قرار گرفت، یکی از امیدوارکننده‌ترین راهکارها برای مواجهه با افزایش پیچیدگی و نگهداری سامانه‌های سازمانی، بهره‌گیری از ویژگی خودتطبیقی در این سامانه‌هاست. در این پژوهش یک مدل تصمیم‌گیری ارائه می‌شود که با استفاده از آن می‌توان از میان چارچوب‌های موجود، یک چارچوب را جهت توسعه‌ی یک سامانه‌ی سازمانی خودتطبیق انتخاب نمود.

در این راستا سؤالات زیر قبل طرح می‌باشند:

۱- هر کدام از این چارچوب‌ها دارای چه ویژگی‌ها و مزایایی نسبت به سایر چارچوب‌ها می‌باشند؟
۲- سازمان‌های توسعه‌دهنده چگونه می‌توانند از میان انواع چارچوب‌های معرفی شده، چارچوب مناسبی را جهت بهره‌برداری انتخاب نمایند؟

۱-۴- مراحل پژوهش

در انجام این پژوهش گام‌های اساسی زیر طی شده است که الزاماً ترتیبی نبوده است:

۱- شناسایی و بیان کامل مسئله، ویژگی خودتطبیقی در نرم‌افزار
۲- بررسی کاربردهای کنونی و موارد استفاده از سامانه‌های خودتطبیق
۳- شناسایی و بررسی اجمالی اجزا و مؤلفه‌های سامانه‌های سازمانی

۴- شناسایی و تشریح چارچوب‌های مورد استفاده در توسعه‌ی سامانه‌های خودتطبیق و ارائه صفت‌یابی[۱] (تعیین ویژگی‌ها و محدودیت‌ها) در خصوص آن‌ها

۵- شناسایی معیارهای مؤثر بر مدل تصمیم‌گیری و نحوه‌ی اثرگذاری آن‌ها بر این مدل

۶- شناسایی تصمیم‌گیرندگان کلیدی در خصوص توسعه‌ی سامانه‌های سازمانی و نحوه‌ی مشارکت ایشان در فرآیند تصمیم‌گیری برای انتخاب چارچوب توسعه

۷- ایجاد نگاشتی بین نیازهای سامانه‌ها، نیازهای سازمان‌ها و صفت‌یابی صورت گرفته و معرفی نحوه‌ی انتخاب چارچوب مورد نظر با استفاده از روش‌های تصمیم‌گیری

۸- انجام دو مطالعه‌ی موردی آزمایشگاهی

۱-۵- محدودیت‌های پژوهش

همانطور که پیشتر اشاره شد، هدف این پژوهش ارائه یک مدل تصمیم‌گیری می‌باشد. برای طراحی مدل‌های تصمیم‌گیری بایستی مجموعه‌ای از راه‌حل‌های مورد استفاده در مسئله تصمیم‌گیری شناسایی شوند. این راه‌حل‌ها در مسئله‌ی حاضر مجموعه چارچوب‌های توسعه‌ی سامانه‌های خودتطبیق می‌باشد. یکی از محدودیت‌های این پژوهش عدم امکان بررسی تمام چارچوب‌های موجود می‌باشد؛ چرا که تعداد چارچوب‌هایی که به طور متناوب در حال ارائه شدن می‌باشند به طور فزاینده‌ای در حال افزایش می‌باشد و امکان بررسی تمام آن‌ها در این پژوهش وجود ندارد. البته مدل ارائه شده امکان پشتیبانی از چارچوب‌های جدید را فراهم خواهد آورد.

محدودیت دوم مربوط به عدم وجود محیط واقعی برای ارزیابی مدل می‌باشد. بدین منظور برای ارزیابی از دو مطالعه‌ی موردی آزمایشگاهی استفاده شده است و از طریق آن‌ها، کاربردپذیری مدل بررسی گردیده است.

۱-۶- ساختار تحقیق

در ادامه این متن، فصول زیر ارائه خواهند شد:

---

[۱] Characterization

- فصل دوم: ادبیات پژوهش؛ در این فصل به بررسی ویژگی خودتطبیقی در سامانه‌های نرم‌افزاری پرداخته شده و این ویژگی در ابعاد مختلفی مورد بحث و بررسی قرار می‌گیرد.
- فصل سوم: چارچوب‌ها؛ در این فصل دسته‌بندی‌های چارچوب‌های توسعه‌ی سامانه‌های خودتطبیق معرفی شده و در ادامه چند چارچوب مورد بررسی قرار می‌گیرند. در انتها این چارچوب‌ها مورد ارزیابی قرار می‌گیرند.
- فصل چهارم: مدل پیشنهادی؛ در این فصل با توجه به بررسی‌ها و تحلیل‌های انجام گرفته در فصول قبل مدل تصمیم‌گیری مورد نظر طراحی شده و در پایان با استفاده از دو مورد کاوی، کاربردپذیری آن نشان داده خواهد شد.
- فصل پنجم: جمع‌بندی؛ در این فصل نتایج، خلاصه پژوهش و مراحل آن و همینطور کارهای پژوهشی آینده ارائه می‌شود.

۲- فصل دوم: ادبیات پژوهش

۲-۱- مقدمه

در این فصل مفاهیم بنیادی در خصوص چارچوب‌های نرم‌افزاری، سامانه‌های خودتطبیق و ویژگی‌های آن‌ها و همچنین سامانه‌های سازمانی معرفی خواهند شد.

۲-۲- تعاریف و مفاهیم اولیه‌ی موضوع

۲-۲-۱- چارچوب نرم‌افزاری

"چارچوب مجموعه‌ای از عناصر همکاری کننده‌ای است که یک طرح قابل استفاده‌ی مجدد برای دسته‌ای خاص از نرم‌افزارها فراهم می‌آورد" [۵]. یک چارچوب یک طراحی قابل استفاده‌ی مجدد می‌باشد که نیازمند تعدادی مؤلفه-ی نرم‌افزاری برای کارکردن می‌باشد. برای نمونه‌سازی از یک چارچوب، مهندس نرم‌افزار باید این مؤلفه‌های مورد نیاز را فراهم آورد. از این رو واسط مؤلفه‌ها نیز بایستی کاملاً تعریف شده باشد [۷]. در حقیقت استفاده‌کننده از یک چارچوب، یک مهندس نرم‌افزار می‌باشد که مؤلفه‌های آن را فراهم می‌آورد. این نحوه‌ی استفاده، چارچوب‌ها را از جعبه‌ابزار[۱]ها یا کتابخانه‌های نرم‌افزاری متفاوت می‌سازد.

۲-۲-۲- معماری نرم‌افزار

بر اساس تعریف ارائه شده از مؤسسه مهندسی نرم‌افزار، معماری نـرم‌افزار، یک برنامه یا سیستم محاسباتی، ساختار یا ساختارهای آن سیستم محـاسباتی است که خصوصیات قابل رؤیت از بیرون عناصر و ارتباطات بین آن‌ها را نشان می‌دهد. در این تعریف تمرکز بر رفتار و محاوره‌های مؤلفه‌ها به صورت جعبه سیاه می‌باشد.

۲-۲-۳- پردازش خودمختار

واژه‌ی پردازش خودمختار نخستین بار درباره‌ی سامانه زیستی بدن انسان بکار گرفته شده است. سامانه عصبی درون بدن انسان، بدون آگاهی شخص، بر ضربان قلب، قند خون و دمای بدن نظارت نموده و آن را تنظیم می‌نماید.

---

[۱] Toolkit

این سامانه قادر است در صورت لزوم از طریق راهکارهای خاصی، آینده را پیش‌بینی نموده و شرایط بدن را با آن تطبیق دهد (مانند افزایش توان ماهیچه‌ها و فشار خون در حالت اضطراب). به طور مشابه در حالت ایده‌آل در سامانه‌های فناوری اطلاعات مبتنی بر پردازش خودکار، خودِ سامانه می‌تواند بر اثر تغییرات محیطی یا داخلی، خود را با آن منطبق نموده و یا حتی نیازهای آتی خود را حدس‌زده و برای پوشش آن‌ها اقدامات لازم را با حداقل تلاش نیروی انسانی به انجام برساند. آزادسازی نیروی انسانی از انجام فعالیت‌های سطح پایین فنی از دو جهت برای سازمان‌ها فایده دارد [۸]:

۱- تعداد نیروی انسانی مورد نیاز برای نگهداشت سامانه کاهش می‌یابد.

۲- نیروی انسانی می‌تواند به فعالیت‌های سطح بالاتری که ارزش افزوده‌ی بیشتری برای سازمان به همراه دارند، بپردازد.

با این وجود تفاوت محسوسی بین این دو ساختار (سامانه عصبی بدن و سامانه‌ی خودمختار فناوری اطلاعات) وجود دارد. بسیاری از تصمیماتی که سامانه عصبی بدن انسان اتخاذ می‌کند غیرداوطلبانه هستند درحالیکه در سامانه‌های خودمختار فناوری اطلاعات، کارشناسان فناوری اطلاعات با آگاهی و اختیار می‌توانند بخشی از وظایف خود را به سامانه منتقل نمایند. این کار از طریق تصویب مجموعه‌ای از سیاست‌ها امکان‌پذیر می‌شود به‌گونه‌ای که سامانه با راهنمایی این سیاست‌ها و عدم نقض آن‌ها می‌تواند به تصمیم‌گیری بپردازد. می‌توان یک سامانه پردازشی خودمختار را یک سامانه با قابلیت مدیریت خود دانست که به تغییرات سیاست‌ها و اهداف حرفه و اتفاقات درون و بیرون خود حساس بوده و به آن‌ها واکنش نشان می‌دهد.

۲-۲-۴- سامانه خودمدیریت[۱]

در یک محیط پردازشی خودمدیریت، عناصر سامانه، خواه عناصر نرم‌افزاری مانند سیستم‌عامل، میان‌افزار یا کاربردهای حرفه، عناصر و مؤلفه‌های سخت‌افزاری مانند دستگاه‌های ذخیره‌سازی و سرورها، قادرند با استفاده از یک حلقه‌ی بازخوردی چهار ویژگی عمده‌ی زیر را از خود بروز دهند:

---

[۱] Self-management

- خودپیکربندی[۱]: قابلیت فراهم نمودن سرویس تحت هر شرایط محیطی و درونی؛ این بدان معناست که سامانه قادر است پیکربندی خود را بر اساس سیاست‌ها و اهداف سطح بالای حرفه که از سوی کاربر تعریف شده، تغییر دهد. این ویژگی بیشتر بر قابلیت خودتطبیقی سامانه ناظر است که از طریق آن سامانه می‌تواند برخی از مؤلفه‌های تحت اختیارش را حذف کند، تغییر دهد، جایگزین نمایند یا حتی مؤلفه‌ی جدیدی را استخدام نماید. این دست تغییرات مؤلفه‌های یک سامانه می‌تواند منجر به تغییر رفتار کل یا بخشی از سامانه شود.
- خوددالتیامی[۲]: جلوگیری از کاستی‌ها و کمی‌ها و تصحیح چنین مواردی در صورت وقوع از طریق تشخیص خودکار نقص‌های سامانه و برطرف سازی آن‌ها. برای این منظور مؤلفه‌های سامانه‌ی مربوطه، سؤِ عمل سامانه را تشخیص می‌دهند تا اقدامات لازم در قبال آن صورت پذیرد. بدین ترتیب سامانه از انعطاف و استحکام بیشتری برخوردار شده در عملیات روزانه‌ی خود، کمتر دچار شکست[۳] می‌شود.
- خودبهینه‌سازی[۴]: نگهداشت کیفیت خدمت بر اساس تغییرات محیطی. بر این اساس سامانه مرتباً منابع و الگوی استفاده از آن‌ها را خود را نظارت می‌کند تا از کارآمدی نحوه‌ی استفاده از آن‌ها اطمینان حاصل نماید. بر این اساس سامانه می‌تواند در صورت تشخیص تغییر در بارکاری[۵] یک حوزه‌ی خاص از حرفه یا تمایلات کاربر سیاست تخصیص منابع خود را بروزرسانی نماید.
- خودحفاظتی[۶]: مقابله با مزاحمت‌ها یا حملات خارجی. این ویژگی نشان‌دهنده‌ی توانایی سامانه در شناسایی و جلوگیری از حملات خارجی و یا خطاهای آبشاری[۷] – که توسط ویژگی خودالتیامی قابل ردیابی نیستند [۹] – می‌باشد. برای تشخیص و مقابله با مخاطرات خارجی از

---

۱ Self-configuration
۲ Self-healing
۳ Failure
۴ Self-optimizing
۵ Work load
۶ Self-protecting
۷ Cascade failures

بررسی رفتار متخاصم[۱] مانند دسترسی غیرمجاز، نفوذ و توزیع بدافزارها[۲] یا حملات رد سرویس[۳] استفاده می‌شود.

۲-۲-۵- حلقه‌ی کنترل[۴]

روش عمده‌ی ساخت یک سامانه خودمختار در جامعه‌ی مهندسی نرم‌افزار، اعطا نمودن یک حلقه‌ی کنترل به سامانه می‌باشد [۱۰]. به طور سنتی این حلقه‌های کنترل در خارج از سامانه قرار گرفته و یک نوع کنترل متمرکز را پدید می‌آورند. برای مثال یک ناظر[۵] خارجی بر سامانه مسلط شده و درباره‌ی تغییرات سامانه تصمیم‌گیری می‌نماید. چنین حلقه‌های کنترلی در دو سطح طراحی می‌شوند:

۱- سراسری[۶]: در این سطح یک حلقه برای کل سامانه وجود دارند که نشانگر یک کنترل مرکزی است که اگر دانش کامل درباره‌ی سامانه داشته باشد می‌تواند به صورت همنواساز[۷] عمل کند. اما چنین حلقه‌ی کنترلی از مشکل عدم انعطاف‌پذیری و عدم مقیاس‌پذیری رنج می‌برد [۱۱].

۲- محلی[۸]: در این سطح هر عنصر خودمختار (در یک سامانه مقیاس فوق وسیع) یا هر بخش مشخص از یک سامانه، صاحب یک حلقه‌ی کنترل است. از آن‌جا که معمولاً دانش تمام یک سامانه بزرگ قابل جمع‌آوری به صورت متمرکز نیست، هماهنگی این حلقه‌های کنترل محلی برای حفظ صحت سامانه لازم است. این حلقه‌ها از انعطاف‌پذیری بالاتری نسبت به مورد قبلی برخوردارند.

مباحث تکمیلی در خصوص اجزا و نحوه‌ی عملکرد یک حلقه‌ی کنترل در بخش مؤلفه خودتطبیق بیان خواهند شد.

---

۱ Hostile behavior
۲ Malware
۳ Denial of Service (DOS)
۴ Control loop
۵ Monitor
۶ Global
۷ Orchestrator
۸ Local

۲-۲-۶- مؤلفه خودتطبیق

یک مؤلفه (عنصر) خودتطبیق را می‌توان مغز اصلی یک سامانه خودتطبیق دانست. باید توجه داشت که عنوان مؤلفه در اینجا الزاماً به معنای وجود یک مؤلفه نرم‌افزاری مشخص برای پیاده‌سازی مؤلفه خودتطبیق نیست و این مؤلفه خود از اجزای گوناگونی تشکیل شده است. این مؤلفه‌ها را می‌توان به یک سامانه‌ی در حال فعالیت اضافه نمود تا برخی ویژگی‌های خودتطبیقی را به آن‌ها اضافه نمود. در حقیقت این مؤلفه مسئول پیاده‌سازی حلقه‌ی بازخوردی یک سامانه خودتطبیق می‌باشد. این مؤلفه چهار کارکرد اصلی زیر را پیاده‌سازی می‌نماید:

- پایش (نظارت): شامل جمع‌آوری، تصفیه و سایر پردازش‌های لازم بر روی داده‌های دریافتی از حسگرها می‌شود.
- تحلیل: این کارکرد در حقیقت نشان‌دهنده تفسیر اطلاعات ورودی است که با استفاده از انواع مدل‌های ریاضی، هوش مصنوعی، پیش‌بینی و ... قابل پیاده‌سازی می‌باشد. یکی از مهمترین مسائلی که در این کارکرد مطرح می‌شود، امکان یادگیری بر اساس داده‌های دریافتی (در حکم بازخورد) می‌باشد.
- برنامه‌ریزی: این کارکرد با توجه به مجموعه سیاست‌ها یا قوانین تعریف شده برای سامانه و جهت دست‌یابی به اهداف آن، مجموعه‌ای از راهکارهای از پیش طراحی شده را برای ایجاد تغییر در عنصر مدیریت شونده اتخاذ می‌نماید. این راهکارها برای ساخت اعمال مورد نیز برای طی فرآیند تطبیق سامانه استفاده می‌شوند.
- اجرا: این کارکرد شامل مجموعه‌ای از راهکارهایی (از جمله زمانبندی اعمال) برای کنترل فرآیند تطبیق سامانه در حین اجرای اعمالی که بالاتر اشاره شد را در اختیار دارد و تغییرات مورد نظر را به کمک عملگرها در عنصر مدیریت شونده اعمال می‌نماید.

با فعالیت این چهار کارکرد در کنار یکدیگر، یک حلقه‌ی کنترل شکل می‌گیرد به نحوی که اجزای آن با انتقال دانش مورد نیاز به یکیگر، با هم همکاری می‌نمایند.

۲-۲-۷- عنصر مدیریت شونده

عنصر مدیریت شونده همان سامانه‌ی فناوری اطلاعات است که ویژگی‌های خودتطبیقی به کمک یک حلقه‌ی کنترل به آن افزوده می‌شود. این عنصر به لحاظ می‌تواند یک سامانه معمولی یا یک سامانه یکپارچه سازمانی باشد.

۲-۲-۸- حسگر

یک حسگر که معمولاً به صورت یک مؤلفه نرم‌افزاری جداگانه پیاده‌سازی می‌شود وظیفه کسب اطلاع و گزارش‌دهی در خصوص وضعیت یک عنصر مدیریت شونده را بر عهده دارد. یک حسگر شامل دو بخش زیر می‌شود:

- مجموعه‌ای از ویژگی‌هایی که اطلاعاتی را درباره‌ی عنصر مدیریت شونده، ارائه می‌دهند
- مجموعه‌ای از رویدادهای مدیریتی که در قبال تغییر وضعیت ویژگی‌های عنصر مدیریت شونده فعال می‌شوند

۲-۲-۹- عملگر

یک عملگر وظیفه ایجاد تغییر در عنصر مدیریت شونده را بر عهده دارد و دو بخش زیر می‌شود:

- مجموعه‌ای از عملیات که امکان تغییر وضعیت عنصر مدیریت شونده را فراهم می‌آورند
- مجموعه‌ای از عملیات که امکان ارائه درخواست از سوی عنصر مدیریت شونده به مؤلفه خودتطبیق را فراهم می‌آورند

اگر حسگر و عملگر در یک سامانه خودتطبیق یکی باشند و یک عنصر مشخص قابلیت‌های هر دو را داشته باشد آنگاه کاوشگر نامیده خواهد شد.

۲-۳- قوانین و سیاست‌های تطبیق

سامانه‌های خودتطبیق و خودمدیریت رفتارشان را بر اساس اهداف و محدودیت‌های از مشخص شده بهینه می‌نمایند. یکی از راه‌های تبیین این اهداف و محدودیت‌ها، بیان آن‌ها در قالب مجموعه‌ای از سیاست‌ها و قوانین می‌باشد [۱۳]. "سیاست‌ها را می‌توان نوعی راهنمایی برای تشخیص اعمال و تصمیمات بهینه دانست" [۱۴].

۲-۴- طیف‌های خودتطبیقی

در این بخش قصد داریم سه طیف مختلف را برای سامانه‌های خودتطبیق با توجه به سه ویژگی نحوه‌ی کارکرد واحد تحلیل و برنامه‌ریزی، گرایش به نظریه‌ی کنترل یا مهندسی نرم‌افزار و همینطور استفاده از روش‌های داخلی یا خارجی معرفی نماییم.

۲-۴-۱- گرایش به مهندسی نرم‌افزار و نظریه کنترل

سامانه‌های خودتطبیق را می‌توان از دیدگاه روشی کلی برخورد با وقایع سامانه دسته‌بندی نمود. روش تطبیق می‌تواند با استفاده از نظریه‌ی کنترل[۱] و یا روش‌های صرفاً مبتنی بر مهندسی نرم‌افزار پیاده‌سازی شود [۱۳]. اگر این دو روش را در دو سر یک طیف در نظر بگیریم روش‌های ترکیبی جایگاهی بر روی این طیف پیدا می‌کنند. در روش-های مبتنی بر نظریه‌ی کنترل، اعمال کنترلی، یک دنباله‌ی ادامه‌دار از سیگنال‌ها هستند که پارامترهای رفتاری منبع مدیریت شونده را تحت تأثیر قرار می‌دهند. معمولاً ساختار منبع مدیریت شونده در این روش‌ها غیرقابل تغییر است در حالیکه می‌توان رفتار آن‌ها را مدل‌سازی ریاضی نمود [۱۶]. در مقابل روش‌های مبتنی بر مهندسی نرم‌افزار عموماً دارای اعمال کنترلی گسسته می‌باشند که می‌تواند ساختار یا معماری منبع مدیریت شونده را تغییر دهد. (شکل ۲-۱: طیف مهندسی نرم‌افزار و نظریه کنترل) این طیف را نشان می‌دهد.

شکل ۲-۱: طیف مهندسی نرم‌افزار و نظریه کنترل

[۱] Control theory

در سامانه‌های خودتطبیق موجود که از روش‌های ترکیبی استفاده می‌نمایند، معمولاً از اعمال کنترلی گسسته که زیرساخت اجراکننده‌ی سامانه مدیریت شونده را تحت تأثیر قرار می‌دهد و یا تغییری در مجموعه پردازش‌های سامانه مدیریت شده اعمال می‌کند، استفاده می‌شود [۱۳].

۲-۴-۲- روش‌های داخلی و خارجی

مجموعه تلاش‌های انجام شده در زمینه‌ی توسعه‌ی مدل‌ها، معماری‌ها، چارچوب‌ها یا به طور کلی روش‌های توسعه‌ی سامانه‌های خودتطبیق را می‌توان در دودسته‌ی بزرگ طبقه‌بندی نمود:

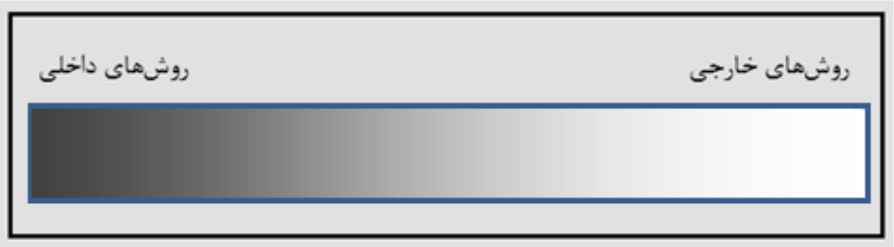
۱- روش‌های خارجی: در این دسته از روش‌ها ماژول‌هایی که ویژگی‌های خودتطبیقی را فراهم می‌کنند اساساً خارج از سامانه اصلی یا همان عنصر مدیریت شونده قرار می‌گیرند.
۲- روش‌های داخلی: در این روش‌ها عنصر مدیریت شونده عملاً حاوی کد لازم برای خودتطبیق‌دهندگی بوده و این دو قسمت به روشنی قابل تفکیک نیستند.

به نظر می‌رسد که روش‌های خارجی به دلیل جداسازی مناسب صفات خودتطبیقی از منطق برنامه و قرار دادن کد مربوط به آن‌ها در ماژول‌های دیگر روش‌های مؤثرتری باشند. در زیر به برخی مزایای این دسته از روش‌ها اشاره می‌نماییم:

- این روش‌ها این امکان را فراهم می‌کنند تا با جداسازی ماژول‌های کنترل‌کننده، بتوان از آن‌ها در سایر سامانه‌ها مجدداً استفاده نمود.
- در این حالت مدیریت و تغییر این مؤلفه‌ها یا سایر مؤلفه‌های سامانه با آثار جانبی کمتری امکان‌پذیر است. زیرا این دو دسته مؤلفه عملاً ارتباط سستی با هم دارند.
- روش‌های خارجی با توجه به ماهیت مستقلشان این قابلیت را دارند که از آن‌ها برای سامانه‌های موروثی فعلی نیز بهره‌برد زیرا می‌توان عناصر آن‌ها را جداگانه به یک سامانه در حال عملیات اضافه نمود که در این صورت سامانه در حال عملیات نیازمند تغییر می‌باشد.

البته در سامانه‌های خودتطبیق کنونی از روش‌های ترکیبی نیز استفاده می‌شود. در واقع همانطور که در (شکل ۲-۲: طیف روش‌های داخلی و خارجی) نشان داده شده دو دسته روش اشاره شده را باید دو سر یک طیف در نظر

گرفت. روند تکاملی سامانه‌ها به سوی استفاده از روش‌های خارجی می‌باشد. چارچوب‌های معرفی شده در این سمینار همگی مبتنی بر روش‌های خارجی می‌باشند.



شکل ۲-۲: طیف روش‌های داخلی و خارجی

۲-۵- مدل تصمیم‌گیری

یک مدل تصمیم‌گیری مجموعه‌ای از فرآیند، معیارها و روش‌های مورد استفاده تصمیم‌گیرندگان برای انتخاب زیرمجموعه‌ای از راه‌حل‌های موجود در خصوص یک مسئله‌ی مشخص می‌باشد.

۲-۵-۱- فرآیند تصمیم‌گیری

تصمیم‌گیری را می‌توان فرآیندی دانست که یک مسئله مشخص و اهداف و نیازمندی‌های تصمیم‌گیرندگان ورودی آن به شمار رفته و در انتها یک راه‌حل مشخص به عنوان خروجی تولید می‌شود. در زیر گام‌های اساسی برای اجرای یک فرآیند تصمیم‌گیری تشریح شده است [۴۲]:

- تعریف مسئله

هدف این گام ارائه یک تعریف مشخص و شفاف از مسئله، شرایط موجود و مطلوب می‌باشد. برای دست‌یابی به چنین تعریف شفافی نیاز است تا تمامی ذینفعان فرآیند تصمیم‌گیری بر روی تعریف مسئله اتفاق نظر داشته باشند. مهمترین اقداماتی که در این گام انجام می‌شود:

- شناسایی دلایل ریشه‌ای وجود مسئله (ریشه‌یابی)
- شناسایی محدودیت‌های گوناگون و فرضیات
- شناسایی مسائل مربوط به ذینفعان

- تعیین نیازمندی‌ها

نیازمندی‌ها شامل مجموعه شرایطی هستند که هر راه‌حل قابل تصور برای حل مسئله باید آن‌ها برطرف نماید. بسیار ضروری است که تا حد امکان مجموعه‌ی نیازمندی‌ها به صورت کمی تعریف شوند.

- شکل‌دهی به اهداف

اهداف مجموعه‌ای از ارزش‌ها هستند که حالات مطلوب را تعریف می‌نمایند. در حقیقت اهداف تصمیم‌گیری مجموعه‌های نیازمندی‌ها را جمع می‌نمایند و وضع مطلوب را به صورت کیفی نمایش می‌دهند.

- شناسایی راه‌حل‌ها

راه‌حل‌های مختلف راهکارهای گوناگونی را برای تبدیل وضع موجود مسئله به وضع مطلوب (اهداف به دست آمده) ارائه می‌نمایند. راه‌حل‌هایی که نیازمندی‌های تعریف شده را برطرف نمی‌کنند از ادامه فرآیند حذف می‌شوند.

- تعریف معیارها

معیار امکان جداسازی و مقایسه راه‌حل‌ها را فراهم می‌آورند. تعریف معیارها با کمک اهداف انجام می‌شود. این بدان معناست که معیارهای تعریف شده باید در جهت تحقق اهداف وضع شوند. راه‌حل‌هایی که معیارها را به شکل بهتری ارضا می‌نمایند، باید عملاً اهداف را به بیشتر محقق نمایند. به ازای هر هدف یک یا تعداد بیشتری معیار تعریف می‌گردد.

- انتخاب روش تصمیم‌گیری

در این گام، تصمیم‌گیرندگان باید روش تصمیم‌گیری را انتخاب نمایند. این روش‌ها در پیوست ۱ معرفی شده‌اند.

- ارزیابی راه‌حل‌ها

ارزیابی راه‌حل‌ها توسط روش تصمیم‌گیری و با استفاده از معیارها انجام می‌شود. ارزیابی ممکن است به صورت عینی یا ذهنی انجام شود که این مسئله به ماهیت ارزیابی بستگی دارد. ارزیابی مقیاس‌های قابل اندازه‌گیری رایج (مثل پول، زمان، طول و ...) به صورت عینی انجام می‌پذیرد. پس از انجام ارزیابی، روش تصمیم‌گیری انتخاب شده در مرحله‌ی قبل برای انتخاب راه‌حل برتر، اولویت‌بندی راه‌حل‌ها یا انتخاب مجموعه‌ای از راه‌حل‌ها به کار گرفته می‌شود.

- ارزیابی

پس از انتخاب یک راه‌حل باید آن را در برابر اهداف و نیازمندی‌های تعریف شده مورد ارزیابی قرار داد. در بعضی موارد راه‌حل انتخاب شده و نتیجه‌ی حاصل از برگزیدن آن باعث می‌شود که نظر تصمیم‌گیرندگان به اهداف یا نیازمندی‌هایی جلب شود که پیشتر مورد غفلت قرار گرفته بوده است.

۲-۵-۲- دسته‌بندی مدل‌های تصمیم‌گیری

مدل‌های تصمیم‌گیری مورد استفاده را می‌توان از دیدگاه‌های مختلف دسته‌بندی نمود:

- تعداد معیارها
  - تک معیاره
  - چند معیاره[۱]
- تعداد راه‌حل
  - متناهی
  - نامتناهی
- تعداد تصمیم‌گیرندگان
  - یک تصمیم‌گیرنده
  - چند تصمیم‌گیرنده[۲]

---

۱ Multi-Criteria Decision Making (MCDM)
۲ Muti-Person Decision Making (MPDM)

به توجه به مسئله مورد پژوهش، مدل تصمیم‌گیری طراحی شده در دسته‌ی چند معیاره، راه‌حل‌های متناهی و چند تصمیم‌گیرنده قرار می‌گیرد.

۲-۵-۳- تصمیم‌گیری گروهی

در تصمیم‌گیری گروهی تلاش می‌شود تا نظرات و ترجیحات افراد مختلف در ارزیابی و انتخاب معیارها و راه‌حل‌های گوناگون، به صورت یک مجموعه‌ی به هم پیوسته‌ی واحد در آید. معمولاً هر یک از تصمیم‌گیرندگان، مجموعه‌ای از تجارب و دانش مخصوص به خود را دارند که می‌تواند در ارزیابی بخشی از راه‌حل‌ها در قبال برخی از معیارها به کار گرفته شود. بنابراین لازم است تا افراد با توجه به زمینه‌ی حرفه‌ای صلاحیت شان در فرآیند تصمیم‌گیری مشارکت داده شوند. این مشارکت در دو زمینه انجام می‌شود:

- تعیین وزن به ازای هر معیار
- تعیین کارایی هر راه‌حل به ازای هر معیار

نحوه‌ی تأثیر صلاحیت هر یک از تصمیم‌گیرندگان نیز به صورت‌های مختلفی قابل اعمال می‌باشد:

- اعمال ضریب مستقیم به تحلیل‌های تصمیم‌گیرندگان در خصوص اوزان معیارها
- اعمال ضریب مستقیم به تحلیل‌های تصمیم‌گیرندگان در خصوص کارایی راه‌حل‌ها
- استفاده از هر تصمیم‌گیرنده برای نظردهی در خصوص معیار یا راه‌حلی که در حوزه‌ی صلاحیت وی می‌باشد

۲-۶- سامانه‌های سازمانی

سامانه‌های سازمانی به سازمان در جهت رسیدن به اهدافشان کمک می‌نمایند. این نرم‌افزارها مسئولیت اجرا و یا تسهیل انجام مراحل مختلف کارهایی که در یک سازمان انجام می‌شوند را بر عهده دارند و معمولاً بر بستر اینترنت و اینترانت (هر دو) اجرا می‌شوند. یکی از اهداف مهم این سامانه‌ها، یکپارچه‌سازی فرآیندهای عمده سازمانی (حسابداری، فروش، مالی، منابع انسانی، انبار، تولید و ...) می‌باشد. سامانه‌های سازمانی یک جز صحیح از یک سامانه اطلاعاتی مبتنی بر رایانه به حساب می‌آیند و برای حل مسائلی که در حوزه سازمان و نه یک یا چند واحد مشخص از

آن وجود دارد استفاده می‌شوند؛ بدین ترتیب بهره‌وری و بازدهی توأمان افزایش می‌یابد. این سامانه‌ها معمولاً به کمک یک واسط به سایر سامانه‌ها پیوند می‌خورند و مدیریت متمرکزی دارند.

مؤسسه گارتنر ویژگی‌های اصلی سامانه‌های سازمانی را به شرح زیر مطرح می‌نماید:

- استفاده از یک نگهدارنده برای اجرای مؤلفه‌های مختلف نرم‌افزار
  - نگهدارنده مجموعه‌ای از واسط‌های برنامه‌نویسی کاربردی[۱] را به مؤلفه‌ها ارائه می‌نماید
  - نگهدارنده معمولاً به صورت مقیم[۲] و به عنوان یک وظیفه[۳] طولانی مدت سیستم عامل اجرا می‌شود
  - می‌توان به کمک سرویس‌های فراهم شده نگهدارنده به صورت از راه دور[۴] به مؤلفه‌های دسترسی داشت
  - منابعی سیستم عامل در اختیار سامانه قرار می‌دهد در کنترل نگهدارنده بوده و نگهدارنده این منابع را در اختیار مؤلفه‌ها قرار می‌دهد
  - نگهدارنده امکان دسترسی به سیستم فایب را برای مؤلفه‌ها فراهم می‌آورد
  - نگهدارنده امکان دسترسی به پایگاه داده را فراهم می‌آورد
- پشتیبانی از محاسبات توزیع شده
- انجام تأیید هویت و دسترسی توسط نگهدارنده
- ارائه امکان پایش و مدیریت حداقلی وضعیت نمونه‌های[۵] نگهدارنده
- امکان فراخوانی[۶] مؤلفه‌ها وجود دارد

همچنین فاولر در [۲۰] مشخصات دیگری را برای سامانه‌های سازمانی مشخص می‌نماید:

- حجم زیاد داده‌های ماندگار

---

۱ Application programming Interface (API)
۲ Resident
۳ Task
۴ Remote
۵ Instance
۶ Invoke

- دسترسی همزمان و چندکاربره به داده‌ها
- تعداد زیاد و تنوع زیاد واسط‌های کاربری
- یکپارچگی با سایر سامانه‌ها که معمولاً تراکنش محور یا زمان محور صورت می‌گیرد
- تعداد زیاد ترجمه و تغییر قالب داده‌ها برای اموری چون یکپارچه‌سازی یا گزارش‌گیری
- نیاز به قابلیت توسعه‌ی بالا

با توجه به تعاریف بالا، سامانه‌های زیر نمونه‌هایی از سامانه‌های سازمانی به شمار می‌روند:

- سامانه‌ی حسابداری
- سامانه‌ی هوش تجاری
- سامانه‌ی مدیریت فرآیندهای حرفه[۱]
- سامانه‌ی مدیریت ارتباط با مشتری[۲]
- سامانه‌ی مدیریت زنجیره‌ی تأمین
- سامانه‌ی مدیریت دارایی‌های مشترک پروژه
- سامانه‌ی مدیریت محتوا
- سامانه‌ی پیام‌رسانی سازمانی[۳]

۲-۷- کارهای مشابه

همانطور که پیش از این نیز اشاره شد، تا کنون هیچ راهنمایی کلی یا مدل تصمیم‌گیری برای انتخاب چارچوب توسعه‌ی سامانه‌های خودتطبیق ارائه نشده است. اما مدل‌های تصمیم‌گیری ارائه شده پر شماری در حوزه‌ی انتخاب چارچوب‌ها در زمینه‌های دیگر وجود دارند. در زیر نمونه‌هایی از این مدل‌های تصمیم‌گیری ارائه شده‌اند:

فتحیان و محمدی در [۴۶] یک مدل تصمیم‌گیری برای انتخاب چارچوب معماری سازمانی ارائه نموده‌اند. در این مدل نخست با بهره‌گیری از نظرات خبرگان و به واسطه‌ی پیمایش دلفی، معیارهای ارزیابی و انتخاب چارچوب مذکور

---

۱ Business Process Managament Suits (BPMS)
۲ Customer Relationship Managament System (CRM)
۳ Enterprse Messaging System (EMS)

از بین معیارهای شناسایی شده، برگزیده و بر مبنای آن‌ها چارچوب‌های مورد توجه به عنوان گزینه‌های تصمیم‌گیری از طریق فرآیند تحلیل سلسله مراتبی ارزیابی شده و پس از تأیید اعتبار تصمیمات اخذ شده در خلال این فرآیند، چارچوب مناسب معماری سازمانی برای سازمان‌های بیمه‌گر مجازی معرفی شده است.

در این مدل از روش مطالعات کتابخانه‌ای برای شناسایی راه‌حل‌ها و معیارهای مدل تصمیم‌گیری استفاده شده است. سپس این مجموعه به عنوان ورودی یک روش تصمیم‌گیری چند معیاره مورد استفاده قرار گرفته است. این مدل با توجه به استفاده از فرآیند سلسله مراتبی تحلیلی امکان رتبه‌بندی و ارزشیابی نسبی راه‌حل‌ها نسبت به یکدیگر را فراهم می‌آورد.

یکی از مشکلات این مدل را می‌توان عدم تعریف دقیق فرآیند تصمیم‌گیری دانست. در این پژوهش گرچه فرآیند طراحی مدل به خوبی تشریح شده است اما راهکار مشخصی برای نحوه‌ی استفاده از مدل و فرآیند آن ارائه نشده است.

در مدلی دیگر، جلیلی و شاه محمدی یک مدل تصمیم‌گیری را برای انتخاب سبک معماری نرم‌افزار با کمک فرآیند تحلیلی سلسله مراتبی معرفی کرده‌اند [۴۷]. در این پژوهش روشی برای انتخاب سبک معماری نرم‌افزار از میان سبک‌های کاندید با توجه به تقدم صفات کیفی نرم‌افزار و با استفاده از فرایند تحلیل سلسله مراتبی ارائه است شده. علاوه بر آن در ارزیابی صفت کیفی قابلیت استفاده مجدد نرم افزار و صفت کیفی قابلیت اصلاح نرم افزار از معیار اتصال استفاده شده است که راهی است برای کمی‌سازی بعد اتصال در این دو صفت کیفی.

فرآیند تصمیم‌گیری مورد استفاده در این مدل شامل شش گام اساسی می‌شود که عبارتند از: ۱- توصیف خواسته‌های نرم‌افزار ۲- تعیین الویت صفات کیفی با استفاده از روش AHP ۳- توصیف سبک‌های کاندید معماری نرم‌افزار ۴- ارزیابی صفات کیفی سبک‌های کاندید ۵- تشکیل ماتریس مقایسه سبک‌های کاندید ۶- رتبه‌بندی سبک‌های معماری کاندید.

در این مدل پژوهشگر به ازای هر سبک معماری یک بار نرم‌افزار را طراحی می‌نماید و سپس با توجه به الگوی بدست آمده از مؤلفه‌های نرم‌افزار، صفات کیفی مختلف را به کمک فرآیند تحلیلی سلسله مراتبی ارزیابی می‌نماید.

در این مدل پژوهشگر نتوانسته دلایل دقیقی برای انتخاب معیارهای ارزیابی هر صفت کیفی ارائه نماید.

۲-۸- نتیجه‌گیری

در این فصل عناصر پایه‌ای مورد نیاز از نظریه‌ی تصمیم‌گیری، مهندسی نرم‌افزار و نیز سامانه‌های خودتطبیق و ویژگی خود تطبیقی معرفی شد. همچنین برخی از مهمترین سامانه‌های سازمانی پرکاربرد و مؤلفه‌های اساسی بکار رفته در ایشان نیز ارائه گردید. در انها نیز دو مدل تصمیم‌گیری در بخش کارهای مشابه ارائه و بررسی گردید.

۳- فصل سوم: چارچوب‌ها

۳-۱- مقدمه

در این فصل به بررسی چارچوب‌های توسعه‌ی سامانه‌های خودتطبیق می‌پردازیم. شایان ذکر است تعداد این چارچوب‌ها همواره در حال افزایش می‌باشد. ابتدا مجموعه چارچوب‌های طراحی شده یا سامانه‌های خودتطبیق را طبقه‌بندی خواهیم نمود و در ادامه به بررسی هفت چارچوب خواهیم پرداخت.

همچنین در این فصل اصول تصمیم‌گیری شامل فرآیند، معیارها، تصمیم‌گیرندگان و روش‌های معروف تصمیم‌گیری معرفی شده و در پایان شیوه‌ی تصمیم‌گیری به صورت گروهی تشریح می‌گردد.

تصمیم‌گیری شامل مطالعه و شناسایی راه‌های جایگزین بر اساس ارزش‌ها و ترجیحات تصمیم‌گیرنده می‌باشد. تصمیم‌گیری به طور ضمنی نشان‌دهنده‌ی وجود راه‌حل‌های جایگزین و گوناگون قابل اجرا برای حل یک مسئله مشخص است. پس در هنگام تصمیم‌گیری شناسایی هر چه بیشتر راه‌حل‌های جایگزین و انتخاب راه‌حلی که به بهترین شکل ممکن اهداف، تمایلات، ارزش‌ها و نیازهای تصمیم‌گیرنده یا تصمیم‌گیرندگان را ارضا نماید، امری مهم است.

۳-۲- طبقه‌بندی چارچوب‌ها

علاوه بر دسته‌بندی‌ها و طیف‌های ارائه شده در فصل اول ، خالد[۱] و سایرین در [۹] یک طبقه‌بندی فراگیر را برای چارچوب‌ها و معماری‌های سامانه‌های خودتطبیق ارائه داده اند که شامل هفت دسته‌ی زیر می‌شود. البته باید به این نکته توجه داشت که هرکدام از این چارچوب‌ها یا معماری‌ها ممکن است صرفاً به یک دسته متعلق نباشند. به عبارت دیگر این هفت دسته نشانگر هفت ویژگی خاص می‌باشد که یک چارچوب یا معماری می‌تواند مجموعه‌ای از آن‌ها را در آن واحد داشته باشد.

- الهام گرفته شده از زیست‌شناسی: استفاده از ویژگی‌های سامانه‌ی عصبی بدن انسان
- توزیع‌شده و مقیاس وسیع: قابل استفاده برای سامانه‌های توزیع‌شده

---

[۱] Amina Khalid

- مبتنی بر عامل: استفاده از چند حلقه‌ی کنترل هر عامل حلقه‌ی کنترل خود را دارد.
- چارچوب‌های مبتنی بر مؤلفه
- فن[۱] محور: استفاده از یک تکنیک خاص ریاضی یا هوش مصنوعی یا نظایر آن
- سرویس‌گرا
- افزودنی: ایجاد امکان افزودن رفتار خودتطبیق به سامانه‌های موروثی موجود

در متن حاضر سعی شده چارچوب‌هایی مورد بحث و بررسی قرار بگیرند که در دسته‌ی افزودنی قابل جایدهی باشند چراکه بسیاری از سازمان‌های کنونی سامانه‌هایی در اختیار دارند که نیازمند ارتباط بین آن‌ها می‌باشند. این نیاز و عدم تمایل سازمان‌ها به دورریزی سامانه‌های فعلی سبب گرایش سازمان‌ها به استفاده از چارچوب‌هایی می‌شود که بتوانند قابلیت خودتطبیقی را به سامانه‌های موجود (موروثی) اضافه نمایند.

۳-۳- چارچارچوب گارلان و دیگران (Ranibow)

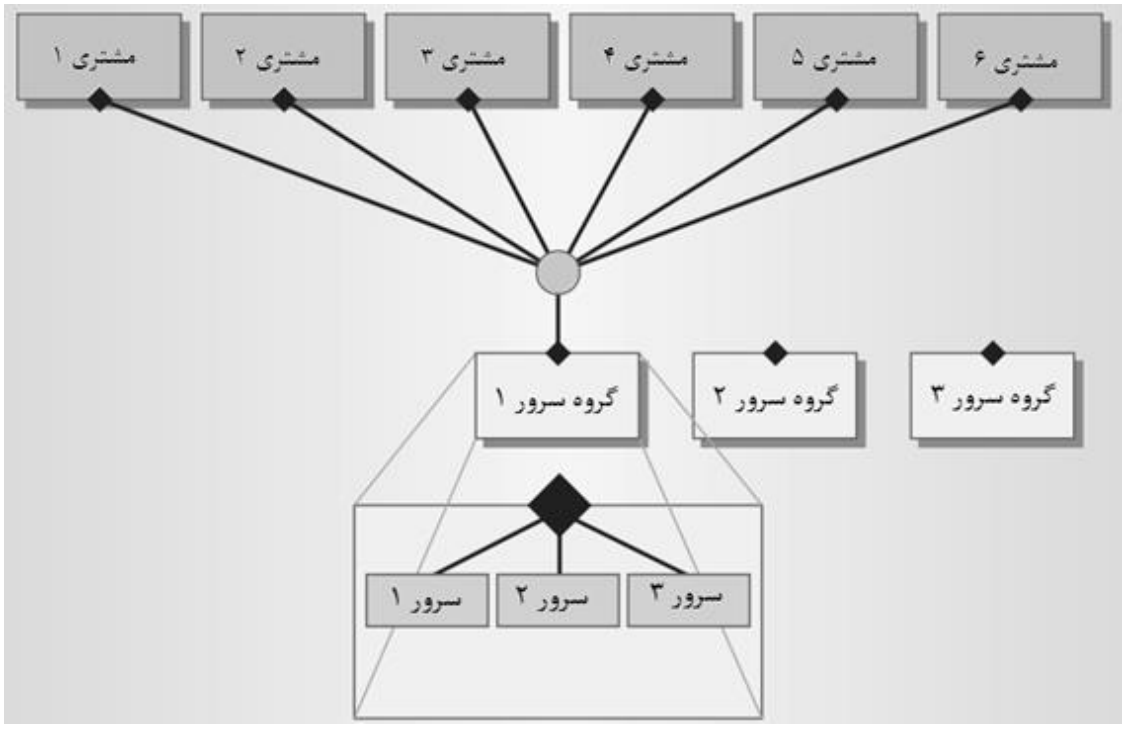
این چارچوب در سال ۲۰۰۴ میلادی توسط دیوید گارلن[۲]، شانگ ون چنگ[۳] و سایرین ارائه شد. این چارچوب یک ساختار قابل استفاده‌ی مجدد را به همراه راهکارهایی برای اختصاصی نمودن آن برای یک کاربرد مشخص ارائه می‌دهد. این راهکارها به توسعه‌دهنده‌ی قابلیت انطباق اجازه می‌دهد که جنبه‌های مختلف را برای مدل‌سازی و نظارت تعیین و سپس شرایطی که محرک انطباق هستند و نحوه‌ی تطبیق را انتخاب نماید. در واقع این چارچوب بستر و زمینه‌ای را برای توسعه‌ی سایر چارچوب‌ها فراهم نموده به نحوی که اکثر اجزای اصلی برای توسعه‌ی یک سامانه‌ی خودتطبیق را در خود جای داده است.

این چارچوب از یک مدل معماری برای نظارت بر ویژگی‌های زمان اجرای سامانه و همینطور ارزیابی مدل برای تشخیص تخطی از محدودیت‌ها و شروط سامانه استفاده می‌نماید. در صورت بروز تخطی، این چارچوب قابلیت انطباق در سطح سامانه و همینطور مؤلفه را فراهم می‌کند. در واقع برخلاف استفاده سنتی از معماری نرم‌افزار به عنوان یک فرآورده‌ی مختص زمان طراحی و توسعه، در این چارچوب مدل معماری سامانه در زمان اجرا نیز نگهداری می‌شود.

---

[۱] Technique
[۲] David Garlen
[۳] Shang-Wen Cheng

این مدل دربرگیرنده‌ی اجزای اصلی یک سامانه مثل واسطها، مؤلفه‌ها، سرورها و نحوه‌ی اتصال آن‌ها و همینطور ویژگی‌هایی از قبیل بار پردازشی سرورها، پهنای باند و زمان پاسخ و غیره می‌شود. به نمایش این مدل معماری در زمان اجرا از یک نمودار به صورت مجموعه‌ای از گره‌ها و اتصالات استفاده می‌گردد که گره‌ها مواردی مثل سرورها، پایگاه‌های داده، مشتری‌ها و واسطهای کاربر می‌باشند و اتصالات مسیر تعاملات بین گره‌ها را نشان می‌دهد. برای هر گره می‌توان از توضیحاتی برای بیان ویژگی‌های آن مثل توان عملیاتی مورد انتظار، میزان تأخیر و توافق‌نامه تعامل استفاده نمود. هر کدام از گره‌ها ممکن است نشان دهنده‌ی یک زیر نمودار دیگر باشد. بدین ترتیب امکان استفاده از چندین مدل معماری به صورت سلسله مراتبی فراهم می‌شود. (شکل ۳-۱: مثال مدل معماری مشتری – خدمتگذار) مثالی از یک مدل معماری استفاده شده در این چارچوب را نشان می‌دهد. در این شکل معماری مشتری - خدمت-گذار را نشان می‌دهد. گره "گروه سرور۱" به یک زیر نمودار دیگر توسعه یافته است. صفات سامانه در شکل زیر نشان داده شده است:

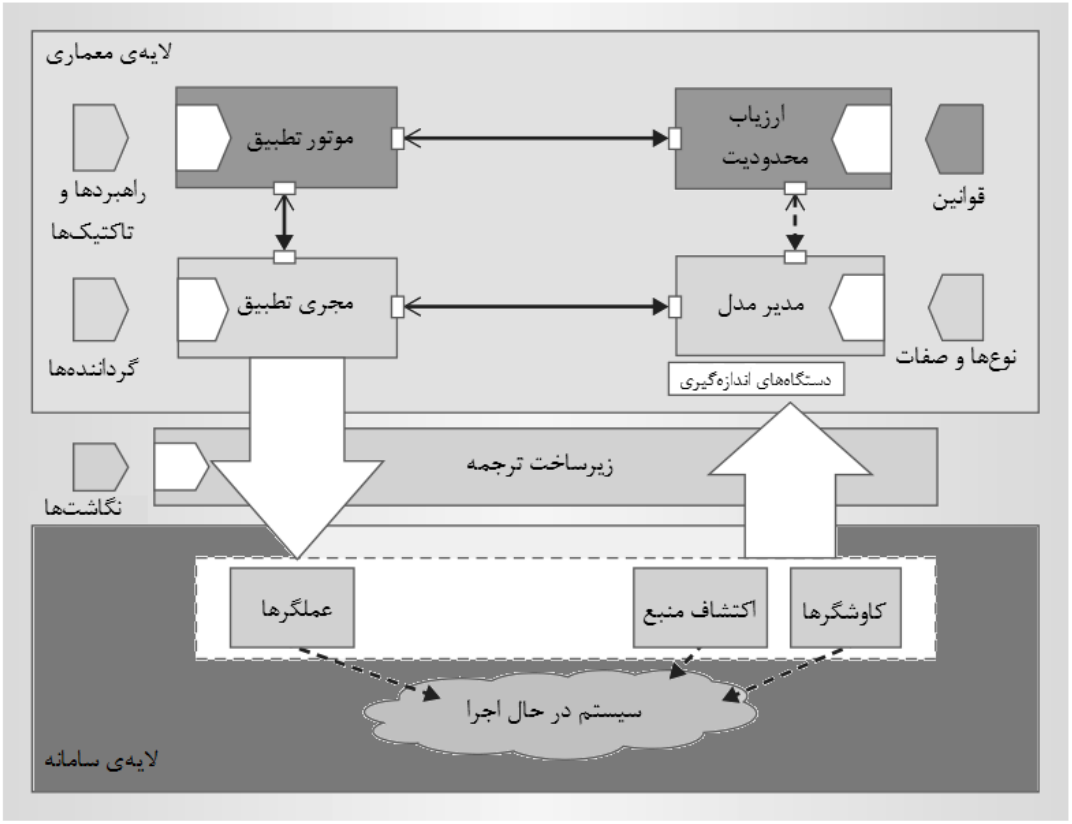


شکل ۳-۱: مثال مدل معماری مشتری – خدمتگذار [۳۷]

استفاده از چنین مدل انتزاعی یک دید کلی و سطح بالا از نحوه‌ی عملکرد و صفات سامانه ارائه می‌نماید. علاوه بر این می‌توان برخی از محدودیت‌های سامانه را نیز در این مدل نشان داد. برای مثال در شکل بالا در هر لحظه یک گروه سرور می‌تواند فعال باشد (شکل دایره). این محدودیت‌ها مشخص‌کننده‌ی تغییرات مجاز برای سامانه می‌باشند؛ به این معنا که یک تغییر معتبر است اگر پس از اجرای آن محدودیت‌ها کماکان ارضا شوند.

۳-۳-۱- طرح کلی چارچوب

بطور کلی این چارچوب شامل دو بخش زیرساخت تطبیق و لایه‌ی سامانه می‌شود. لایه‌ی سامانه نشانگر یک سامانه‌ی معمولی در حال اجراست که قرار است خواص خودتطبیقی به آن افزوده شود. این کار به کمک بخش زیرساخت تطبیق انجام می‌پذیرد. زیرساخت تطبیق در دو لایه پیاده سازی می‌شود: معماری و ترجمه. این لایه وظیفه‌مندی‌های عمومی یک سامانه خودتطبیق را فراهم می‌آورد. (شکل ۳-۲: طرح کلی چارچوب RAINBOW) ساختار کلی این چارچوب را نمایش می‌دهد:



شکل ۳-۲: طرح کلی چارچوب RAINBOW [۳۷]

۳-۳-۱-۱- لایه‌ی سامانه

این لایه شامل واسط دسترسی به سامانه است. کاوشگرها که راهکارهایی برای اندازه‌گیری سامانه هستند بر سامانه در حال اجرا نظارت می‌نمایند. این اطلاعات توسط کاوشگرها منتشر شده و یا هنگام پرس‌وجو آن‌ها را ارائه می‌کنند. راهکار کشف منبع در خصوص وجود منابع جدید مورد پرس‌وجو قرار می‌گیرد. عملگران نیز تغییرات را در سامانه اعمال می‌نمایند.

۳-۳-۱-۲- لایه‌ی معماری

دستگاه‌های اندازه‌گیری، اطلاعات را از کاوشگرها جمع‌آوری نموده و در صورت تغییر در سامانه‌ی در حال اجرا، مدل معماری سامانه را بروز می‌نمایند تا وضعیت جدید سامانه را نشان دهد. دستیابی به مدل معماری از طریق مدیر مدل فراهم می‌شود. ارزیاب، محدودیت‌های مدل معماری را به صورت دوره‌ای بررسی می‌نماید تا تخطی از آن‌ها را کشف نماید. در صورت بروز تخطی فرآیند تطبیق سامانه را آغاز می‌کند. سپس موتور تطبیق روال تطبیق را یافته و اجرا می‌نماید.

۳-۳-۱-۳- زیرساخت ترجمه

در نگاشت اطلاعات بین دو لایه‌ی بالایی از این لایه کمک گرفته می‌شود. برای این منظور یک مخزن ترجمه که حاوی نگاشت‌های گوناگون است برای این کار لازم می‌باشد.

۳-۴- چارچوب جورجس[۱] و تیلور[۲]

این چارچوب در سال ۲۰۰۴ میلادی ارائه شده است. "هدف اصلی این چارچوب معرفی یک روش مبتنی بر دانش و بر محور معماری، برای تعیین و ایجاد سیاست‌های تطبیق‌پذیری به عنوان محرک اصلی انطباق‌پذیریست" [۳۶]. استفاده از این چارچوب امکان تغییر و تحول پویا و مستقل سیاست‌های انطباق سامانه را فراهم می‌آورد. در حقیقت بنیان اصلی این چارچوب بر تغییر معماری سامانه در زمان اجراست که این ایده‌ی (تغییر معماری) در [۴] تشریح شده است.

---

[۱] John C. Georgas
[۲] Richard N. Taylor

۳-۴-۱- طرح کلی چارچوب

در این چارچوب استدلال و تصمیم‌گیری درباره‌ی زمان و طبیعت یک انطباق بخصوص بر مبنای سیاست‌های انطباقی است که در یک پایگاه‌دانش[۱] ذخیره می‌شوند. این پایگاه‌دانش علاوه بر این سیاست‌ها شامل دانش مرتبط با خود سامانه نیز می‌شود. این دانش برای مثال می‌تواند درباره‌ی میزان منابع سامانه و سیاست‌های استفاده از این منابع باشد. این چارچوب سیاست‌های انطباق و دانش مرتبط با سامانه را بعنوان عناصر سطح اول معماری در نظر می‌گیرد [۳۶]. با جدا در نظر گرفتن این اجزا از پیاده‌سازی سامانه، امکان توسعه‌ی مجزای آن و همینطور اعمال تغییرات در آن افزایش می‌یابد. مجموعه‌ی این پایگاه‌دانش توسط یک سامانه خبره[۲] بطور پویا مدیریت می‌گردد.

مؤلفه‌های اصلی این چارچوب و همینطور اتصالات میان آن‌ها توسط توضیحات معمارانه[۳] مدل‌سازی می‌شوند. سپس از این مدل‌ها برای تجزیه‌ی انطباق به سه عنصر مختلف استفاده می‌شود:

۱- انواع داده یا دانش‌هایی که نشان‌دهنده‌ی نیاز به انطباق هستند.

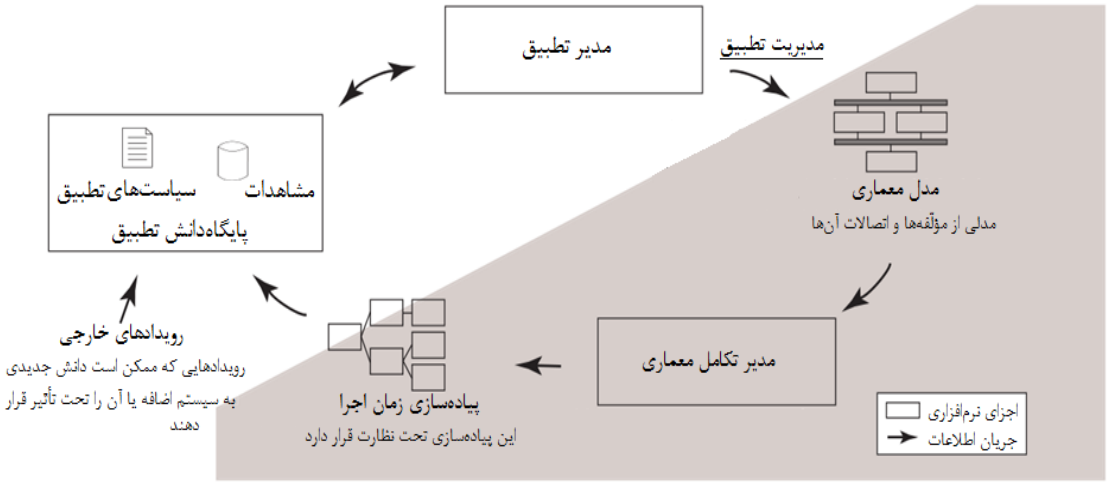
۲- اعمالی معمارانه که از طریق آن‌ها فرآیند انطباق عملی می‌شود.

۳- سیاست‌های انطباقی که ارتباط بین دو مورد قبل را فراهم می‌آورد.

با داشتن دانش در خصوص موارد بالا می‌توان با استفاده از فنون مربوط به پایگاه دانش، به استدلال پویا پرداخت. در (شکل ۳-۳: اجزا و جریان اطلاعات در چارچوب جرجس و تیلور) نمای کلی این چارچوب نشان داده شده است.

---

[۱] Knowledge base
[۲] Expert system
[۳] Architectural



شکل ۳-۳: اجزا و جریان اطلاعات در چارچوب جرجس و تیلور [۳۶]

در ادامه به تشریح عناصر تصویر بالا می‌پردازیم:

مدیر تکامل معماری: مسئول حفظ همخوانی مدل معماری و پیاده‌سازی آن در سامانه است.

پایگاه‌دانش تطبیق: این پایگاه، رویدادهای گرفته شده از سامانه در زمان اجرا و همینطور رویدادهای گرفته شده از ابزارهای نظارتی را جمع می‌نماید. این پایگاه دانش مجموعه‌ی این رویدادها را در یک منبع مشاهدات جمع‌آوری می‌نماید و با استدلال بر روی آن‌ها دانش مرتبط با سامانه‌ی خودتطبیق را کپسوله[۱] می‌نماید. این استدلال‌ها بر مبنای سیاست‌های تطبیق معماری انجام می‌پذیرد که از آن برای تشخیص زمان و ماهیت تطبیق سامانه استفاده می‌گردد. در انتها این پایگاه دانش واکنش‌هایی را که جهت تطبیق سامانه ضروری می‌داند بعنوان "اصلاحات" به مدیر تغییر معماری اعلام می‌نماید. این واکنش‌ها شامل دو نوع هستند:

۱- تغییر سیاست‌های تطبیق جاری
۲- تغییر ساختار در مدل معماری

---

[۱] Encapsulate

در پایان هر تغییر (تطبیق) مدیر تکامل معماری مطمئن می‌شود که تغییرات اعمال شده در عملیات و ساختار اجرایی سامانه نیز اعمال شده باشد.

مدیر تطبیق معماری: در این چارچوب پاسخ‌های انطباق تولید شده توسط پایگاه‌دانش تطبیق مستقیماً اعمال نمی‌شوند بلکه توسط مدیر تطبیق هماهنگ و اعمال می‌شوند.

مدل معماری: می‌توان این مدل را یک تجریدی دانست که ساختار و معماری یک نرم‌افزار را نشان می‌دهد.

۳-۴-۲- سیاست‌های انطباق در پایگاه‌دانش

اطلاعات مرتبط با تطبیق بصورت مجموعه‌ای از مشاهدات، جمع‌آوری می‌شوند که حقایق دانسته شده درباره‌ی سامانه را تشکیل می‌دهند [۳۶] درحالیکه سیاست‌های تطبیق تعریف‌کننده‌ی قوانینی هستند که ممکن است توسط هر یک از مشاهدات سامانه تحریک[۱] شده و در نتیجه یک یا چند پاسخ تطبیق[۲] را برای اجرا توسط مدیر تطبیق تعیین نمایند. هر یک از این سیاست‌های تطبیق دارای شماره‌ی شناسایی و توضیح متنی (اختیاری) است. به هر قانون تطبیق یک یا چند رویداد به عنوان محرک مرتبط می‌شوند تا به عنوان مشاهده‌ی تحریک‌کننده‌ی قانون عمل نمایند. هر یک از مشاهدات و پاسخ‌های تطبیق نیز دارای شماره‌ی شناسایی خاص خود هستند تا به طور جداگانه قابل دسترسی و تغییر باشند.

به طور کلی دو منبع برای دریافت مشاهدات وجود دارد: اول، خود سامانه و دوم، اطلاعات خارجی منتقل شده به سامانه. نوع اول ممکن است ناشی از نتیجه یک عملیات جاری سامانه یا از طرف زیرساخت نظارت باشد. نوع دوم نیز ناشی از رویدادهای صادر شده از طرف یک سامانه خارجی و یا کاربر می‌باشد. این مجموعه مشاهدات صرف نظر از منبعشان به لحاظ محتوایی نیز به دو دسته قابل تقسیم می‌باشند:

۱- اطلاعات ساختاری درباره‌ی معماری سامانه (مستقل از معنا[۳])
۲- دانش وابسته به معنا که با ویژگی‌های مورد نظر رفتاری ارتباط دارد.

---

[۱] Trigger
[۲] Adaption response
[۳] Semantic

نوع دوم وابستگی زیادی به هر سامانه مشخص دارد و بایستی توسط معمار نرم‌افزار تعیین گردد [۳۶] ولی نوع اول شامل اطلاعاتی عمومی مثل خطا در اجرای مؤلفه‌های نرم‌افزاری و موارد داخل سامانه‌ای از این دست می‌باشد. این اطلاعات زمینه‌ی لازم را برای اظهار و نمایش تغییرات مرتبط با عناصر سطح بالای معماری شامل مؤلفه‌ها و اتصالات بین آن‌ها را فراهم می‌آورد. از آن‌جا که این اطلاعات عمومی در بیشتر سامانه‌ها مشابه هستند امکان استفاده مجدد از آن‌ها بالاست.

۳-۴-۳- پاسخ‌های تطبیق

پاسخ‌های تطبیق تغییرات مشخصی هستند که در نتیجه‌ی نیاز به تطبیق سامانه بایستی اعمال شوند [۳۶] که مانند مشاهدات ممکن است وابسته یا مستقل از معنا باشند. پاسخ‌های وابسته به معنا ناظر بر تغییر در سیاست‌های تطبیق سامانه (افزودن، حذف و تغییر مشاهدات و سیاست‌های تطبیق از پایگاه دانش) می‌باشد و در مقابل پاسخ‌های مستقل از معنا ناظر بر تصحیح ساختاری معماری (افزودن، حذف و یا جایگزینی مؤلفه‌ها و اتصالات آن‌ها) است. هر دوی این نوع پاسخ‌ها از کارکدهای اصلی سامانه مستقل هستند [۳۶].

۳-۵- چارچوب AOA[۱]

این چارچوب در سال ۲۰۰۸ میلادی توسط تروین[۲] و جوسن[۳] ارائه شده است. در این چارچوب سعی شده الگویی بر اساس برنامه‌نویسی جنبه‌گرا و بر مبنای چارچوب‌های مطرحی مانند [۳۷] ارائه شود. مهمترین مشخصه‌ی این چارچوب تمرکز بالای آن بر دغدغه‌های سامانه به ویژه نیازمندی‌های کیفی سامانه مانند دسترس‌پذیری، کارایی و مواردی از این دست می‌باشد؛ چراکه عملاً هر نیاز کیفی را به عنوان یک دغدغه در نظر می‌گیرد به طوریکه یک کد جنبه مسئول نگهداری و مدیریت آن می‌باشد.

۳-۵-۱- طرح کلّی چارچوب

این چارچوب دارای سه بخش اصلی است:

---

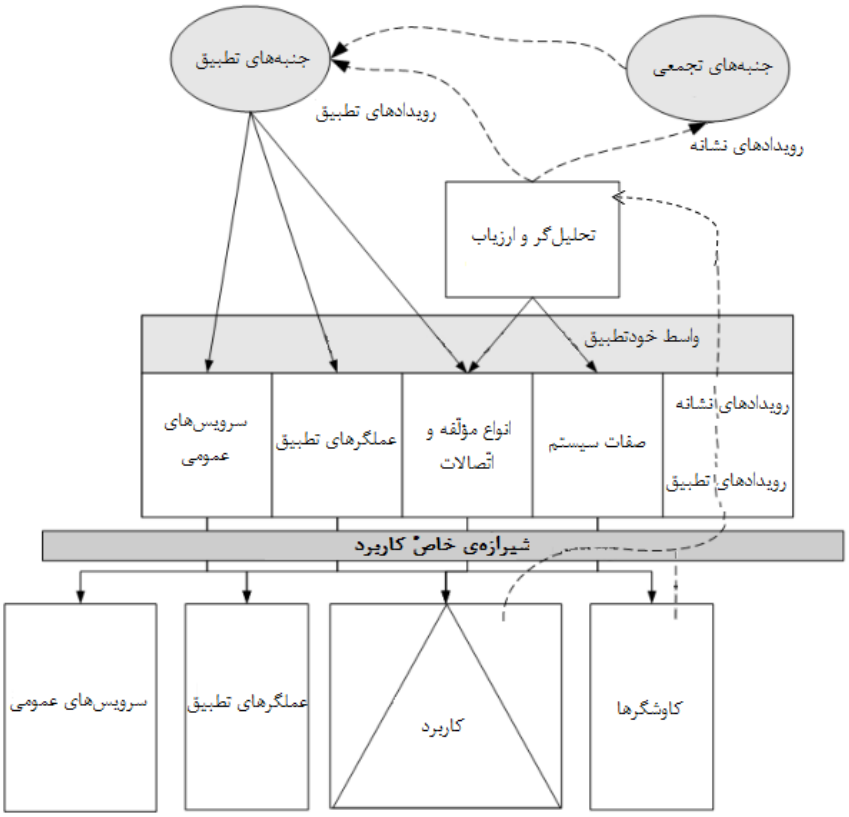
[۱] Aspect Oriented Architecture
[۲] Eddy Truyen
[۳] Wouter Joosen

۱- یک واسط خودتطبیق که تمام موارد مورد نیاز برای اجرای سیاست‌های تطبیق خاصِّ سامانه را فرموله و اجرا می‌نماید.

۲- یک شیرازه‌ی خاصِّ کاربرد که آن را به واسط خودتطبیق متّصل می‌نماید.

۳- یک تحلیل‌گر یا ارزیاب که بین انواع رویدادها تمایز قائل می‌شود و در عین حال می‌تواند نقض شدن سیاست‌های تطبیق مثل کاهش کارایی یک مؤلّفه تا یک حدِّ مشخّص را تشخیص دهد.

شکل زیر ساختار کلّی این چارچوب را نشان می‌دهد:



شکل ۳-۴: عناصر اصلی چارچوب AOA [۳۷]

۳-۵-۱-۱- واسط خودتطبیق

این واسط یک لایه‌ی انتزاعی که فقط اطلاعات مرتبط با چارچوب را منعکس می‌کند، ارائه می‌نماید. این اطلاعات شامل مجموعه‌ای از اتصالات و مؤلّفه‌ها را نشان می‌دهد که هرکدام دارای خواص (نوع اطلاعات مورد نیاز برای تشخیص زمان و ماهیّت انواع تطبیق) و روش‌هایی (تسهیل پیاده‌سازی انواع تطبیق) هستند. محتوای واسط به دو مورد بستگی دارد [۶]: اول، سبک معماری مورد استفاده در کاربرد و دوم، دغدغه‌های سامانه مثل کارایی، دسترس-پذیری و .... بر اساس این دو مورد، توسعه‌دهنده می‌تواند واسط خودتطبیق را مشخص نماید.

۳-۵-۱-۲- تحلیل‌گر یا ارزیاب

در این چارچوب از دو نوع رویداد متفاوت استفاده می‌شود:

۱- نشانه[۱]: این نوع رویداد به منزله‌ی نقض یک محدودیت یا بروز یک وضعیت مشکل‌ساز است که لزوماً نیازمند تطبیق فوری سامانه نیست.
۲- تطبیق: این نوع رویداد سامانه را وادار به اجرای فوری تطبیق می‌نماید.

وظیفه‌ی این عنصر در تمییز دادن این دو نوع رویداد و فرستادن هرکدام به قسمت مربوطه می‌باشد. همچنین سیاست‌های تطبیق در این عنصر قرار می‌گیرند تا با بررسی صفات سامانه یا رویدادهای دریافتی نقض این سیاست‌ها تشخیص داده شود. بنابراین این عنصر دو روش برخورد با ورودی‌هایش در پیش می‌گیرد:

۱- سیاست‌های تطبیق که با یک رویداد تطبیق فعّال می‌شوند: این سیاست‌ها را می‌توان به صورت جنبه پیاده‌سازی نمود. برای مثال در رابطه با دغدغه‌ی دسترس‌پذیری می‌توان یک کد جنبه نوشت به صورتی که با از کارافتادگی یک سرور یک سرور جدید در گروه سرورها جایگزین آن شود.
۲- سیاست‌های تجمّعی که با یک رویداد نشانه فعّال می‌شوند: گاهی با بروز یک رویداد نیازی به تطبیق رفتار سامانه وجود ندارد ولی این رویداد می‌تواند نشانه‌ای از نیاز به تطبیق احتمالی در

---

[۱] Indication

آینده باشد. بنـابراین لازم اسـت تـا رویـدادها جمـع‌آوری و نگهـداری شـده تـا امکـان برنامـه‌ریـزی بـرای آینده بر اساس الگوی ورودی این رویدادها فراهم گردد.

۳-۵-۱-۳- شیرازه‌ی خاصِّ کاربرد

واسط خودتطبیق از طریق این شیرازه به یک سامانه یا کاربرد مشخّص متّصل می‌گردد. این شیرازه دارای چهار بخش اصلی است که وظایف زیر را انجام می‌دهند:

۱- هرکدام از عناصر نرم‌افزاری سامانه را به نوع خاصّی از مؤلّفه یا اتّصال نگاشت[۱] می‌نماید.
۲- با استفاده از کاوشگرهایش به این عناصر دسترسی دارد و صفات سامانه را محاسبه می‌نماید.
۳- از طریق عملگرهای تطبیق، تغییرات را در سامانه اعمال می‌نماید.
۴- خدمات زیرساختی رایج مثل یافتن منبع را فراهم می‌آورد.

۳-۶- چارچوب DSCB[۲]

این چارچوب در سال ۲۰۰۸ میلادی توسط بارِسی[۳]، گینی[۴] و تامبورلی[۵] ارائه شده است. فرض اساسی در طراحی این چارچوب بکارگیری آن برای محیط‌های توزیع شده می‌باشد که البته امکان استفاده از آن به صورت متمرکز نیز وجود دارد.

در این چارچوب عدم تمرکز از طریق گروه‌بندی اجزای نرم‌افزاری در خوشه[۶]هایی همگن بدست می‌آید. این گروه‌بندی بر اساس معیارهای خاص سامانه انجام می‌شود. هر خوشه دارای یک مؤلفه‌ی مشخص به عنوان سرپرست آن خوشه است. مؤلفه‌ی سرپرست مسئول سرپرستی خوشه‌ی خود و همینطور جمع‌آوری اطلاعات از سایر قسمت-های سامانه می‌باشد که با ارتباط مؤلفه‌های سرپرست با یکدیگر از طریق یک شبکه‌ی ارتباطی امکان‌پذیر شده است. سایر مؤلفه‌ها که در خوشه‌های مختلف توزیع شده‌اند سه نقشِ پیاده‌سازی منطق حرفه، حسگر یا عملگر را ایفا می-نمایند.

---

۱ Map
۲ Decentralized Self-adaptive component-based
۳ Luciano Baresi
۴ Sam Guinea
۵ Giordano Tamburrelli
۶ Cluster

سه سطح مخلف تطبیق‌پذیری در این چارچوب قابل پیاده‌سازی است:

- سطح اول سطح کلی است که در آن با توجه به اطلاعات گردآوری شده از مؤلفه‌های سرپرست در خصوص اعمال تطبیق در سطح کلی سامانه تصمیم‌گیری می‌شود.
- سطح دوم سطح محلی است که در آن یک مؤلفه‌ی سرپرست با توجه به اطلاعات خوشه‌ی خود، تغییرات را در خوشه اعمال می‌نماید. در این حالت سرپرست می‌تواند رفتار یک یا چند مؤلفه را تغییر دهد.
- سطح سوم سطح مؤلفه می‌باشد. در این چارچوب با استفاده از تکنیک‌های جنبه‌گرایی امکان تغییر رفتار برای هر مؤلفه به طور مجزا فراهم آمده است که مؤلفه‌ی را قادر می‌سازد با توجه به اطلاعات بدست آمده از کاوشگرها درصورت نیاز خود را تطبیق دهد.

۳-۶-۱- طرح کلی چارچوب

این چارچوب قابلیت پشتیبانی از خودتطبیقی سامانه‌های مبتنی بر مؤلفه‌های توزیع شده را بر مبنای یک شبکه-ی نظیر به نظیر دارد. این شبکه برای اتصال و ارتباط بین مؤلفه‌ها بکارمی‌رود که دو دسته مؤلفه را در بر می‌گیرد:

- مؤلفه‌های تحت سرپرستی: این مؤلفه‌ها وظیفه‌ی پیاده‌سازی منطق حرفه را دارند. همچنین ممکن است با توجه به محتوای استقرار در نقش حسگر یا عمل‌گر نیز ظاهر شوند. قابلیت خودتطبیق‌دهندگی در این مؤلفه‌ها به صورت یک دغدغه‌ی مداخله‌ای در نظر گرفته می‌شود. برای ایجاد امکان تطبیق در این مؤلفه‌ها از کاوشگرهای جنبه‌گرا استفاده می‌شود که می‌توانند همزمان نقش حسگر و عملگر را ایفا نمایند. بدین ترتیب سطح سوم خودتطبیقی در این چارچوب عملی می‌شود.
- مؤلفه‌های سرپرست: این دست مؤلفه‌ها برای نظارت بر عملکرد و رفتار سایر مؤلفه‌ها به کار می-روند. همچنین تصمیم‌گیری در مورد تطبیق سطح اول و دوم نیز به عهده‌ی این مؤلفه‌ها می‌باشد.

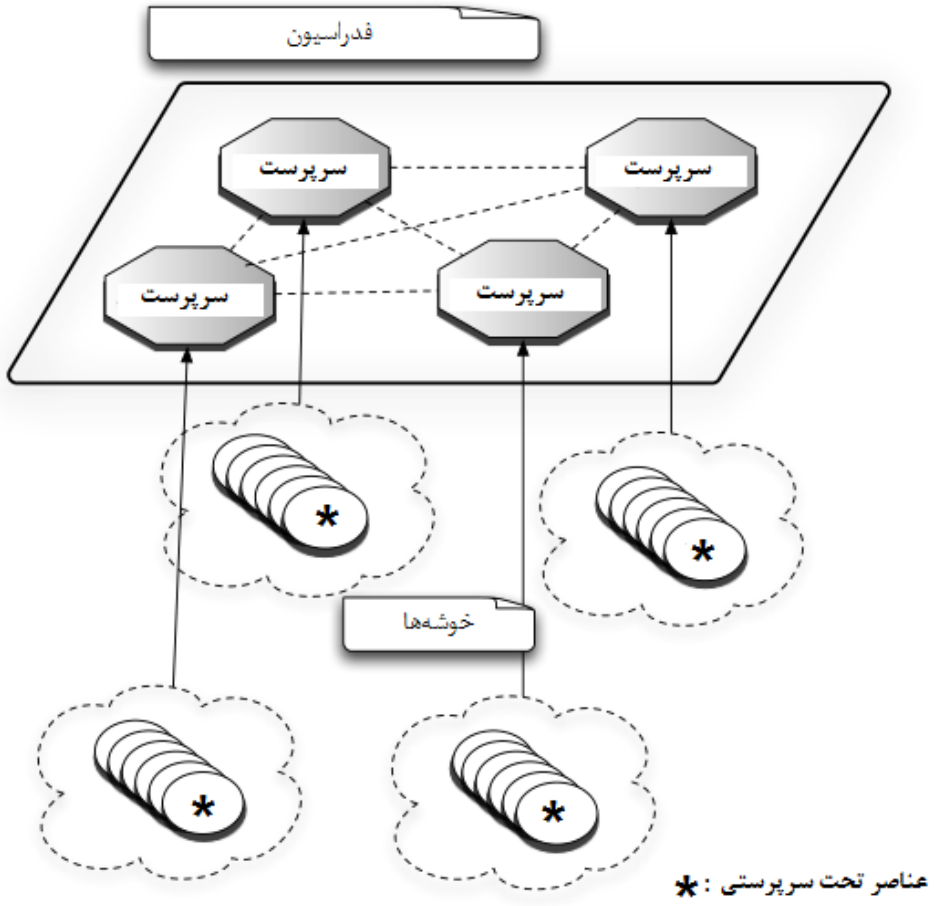
همانطور که اشاره شد در یک نگاه کلی این چارچوب مجموعه‌ای از خوشه‌هاست که هرکدامشان مجموعه‌ای از مؤلفه‌های تحت نظارت را شامل شده و توسط یک سرپرست، مدیریت می‌شوند. در هنگامیکه یک مؤلفه‌ی جدید قصد اضافه شدن به سامانه را دارد، کد آگاهی[۱] آن وظیفه دارد خوشه‌ای را بیابد که معیارهای خاص سامانه را بهینه نماید. این معیار می‌تواند مثل فاصله‌ی فیزیکی مؤلفه‌های همکار ساده و یا هر جزء اطلاعاتی دیگر در سامانه باشد. پس از پیوستن مؤلفه به خوشه، کد آگاهی آن یک جریان اطلاعاتی با سرپرست خوشه می‌گشاید. هرگونه پیوستن، خروج یا شکست در اجرای مؤلفه‌ها با الگوی رویداد محور به اطلاع سرپرست می‌رسد. رویداد مربوط به شکست، این امکان را به مؤلفه‌های تحت سرپرستی می‌دهد تا در صورت شکست در اجرای سرپرست، از آن آگاه شده و خوشه‌ی جدیدی برای خود بیابند.

تمامی مؤلفه‌های این چارچوب می‌توانند با استفاده از یک توافق‌نامه‌ی اشتراک و اطلاع‌رسانی [۳۹] خود را از وقوع رویدادهایی خاص که توسط کاوشگرها شناسایی می‌شوند، مطلع گردند. هر سرپرست با اشتراک در مجموعه‌ای از رویدادهای مرتبط با منطق حرفه‌ی تحت سرپرستیش از رویدادهای آن آگاه شده و می‌تواند با استفاده از این توافق‌نامه حجم اطلاعات دریافتی را به صورت پویا تنظیم نماید [۴۰].

برای انجام تطبیق سطح اول نیاز است تا مؤلفه‌های سرپرست با یکدیگر ارتباط برقرار نموده تا امکان داشتن تصویری از سامانه فراهم شود. بنابراین سرپرستان نیز در قالب یک اتحادیه[۲] گردهم می‌آیند و با همان توافق‌نامه‌ی اشتراک و اطلاع‌رسانی ارتباط برقرار می‌نمایند. برای ورود و خروج سرپرستان جدید نیز از همان شیوه‌ی رویداد محور معرفی شده استفاده می‌شود. (شکل ۳-۵: خوشه‌ها و فدراسیون) نمای کلی این چارچوب را نشان می‌دهد [۴۰].

---

[۱] Advice code
[۲] Feeration



شکل ۳-۵: خوشه‌ها و فدراسیون [۴۰]

۳-۶-۲- تطبیق در گستره‌ی خوشه[۱]

در قسمت قبلی در خصوص تطبیق در سطح سوم (مؤلفه) توضیح داده شد. در این قسمت به سطح دوم (خوشه) می‌پردازیم. سرپرست رویدادهای رخ داده در سامانه (آن‌هایی را که قبلاً مشترک شده) را برای مشاهده‌ی الگویی خاص بررسی می‌نماید. در صورتیکه سرپرست تصمیم به تغییر بگیرد، می‌تواند مؤلفه‌های درون خوشه را تغییر دهد.

[۱] Cluster-wide adaption

۳-۶-۳- تطبیق در گستره‌ی سامانه

در این نوع تطبیق سیاست‌هایی اعمال می‌شود که بر تمام سامانه اثرگذار است. برای تحلیل و استنتاج در خصوص لزوم تطبیق سامانه لازم است اطلاعاتی از سراسر آن جمع‌آوری گردد که این امر از طریق جریان رویدادهای معرفی شده بین خوشه‌ها امکان پذیر می‌شود. در حقیقت توزیع دانش در سامانه از طریق توافق‌نامه‌ی اشتراک و اطلاع‌رسانی صورت می‌پذیرد [۴۰].

۳-۶-۴- حلقه‌های کنترل

حلقه‌ی کنترل معرفی شده در این چارچوب شامل ۵ مرحله‌ی اصلی می‌باشد؛ جمع‌آوری[۱]، ارتباط‌دهی[۲]، متراکم-سازی[۳]، تحلیل و واکنش که هر یک از مؤلفه‌های این چارچوب امکان اجرای این حلقه را دارند.

- جمع‌آوری: شامل دریافت رویدادها و تولید توالی‌های محدودی از رویدادهایی است که در تمام حلقه‌ی کنترل مد نظر هستند. این توالی‌ها به گام بعدی می‌روند.
- ارتباط‌دهی: در این گام سامانه با استفاده از یک مشخصه‌ی ارتباط‌دهی[۴] رویدادهای مرتبط با هر توالی را تطابق می‌دهد. هر مشخصه‌ی ارتباط‌دهی یک توالی جدید از چندتایی‌های[۵] رویداد تعریف می‌نماید که یک مشخصه را ارضا می‌نماید.
- متراکم‌سازی: در این گام داده‌های اضافی حذف، و داده‌های باقیمانده بازآرایی شده تا برای تحلیل آماده شوند.
- تحلیل: در این گام مشخص می‌شود که آیا توالی‌ها، مجموعه مشخصه را ارضا می‌کنند یا خیر. اگر این‌طور نباشد توالی‌ها به گام بعدی ارسال می‌شوند.
- واکنش: واکنش مناسب در این گام انجام می‌شود تا رفتار مؤلفه تغییر یابد.

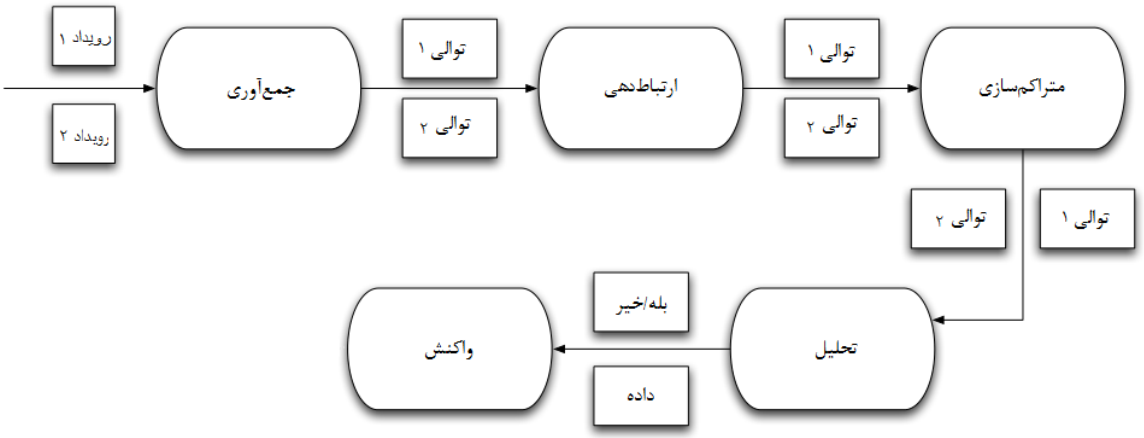
---

۱ Collection
۲ Correlation
۳ Aggregation
۴ Correlation property
۵ Tuple

با توجه به سطح تطبیق‌پذیری در سامانه می‌توان انتظار داشت که سه حلقه‌ی کنترل نیز در سامانه وجود داشته باشد. اولین حلقه‌ی کنترل در سطح یک مؤلفه اجرا می‌شود. هر مؤلفه با اشتراک برای مجموعه‌ی رویدادهای خاص مورد نظرش می‌تواند در صورت نیاز در رفتارش تغییر ایجاد کند. فرضاً با بروز خطا در اجرای سرپرست، مؤلفه‌های خوشه‌ی مربوطه این امکان را دارند تا خوشه‌ی جدیدی بیابند. در اینجا مؤلفه به کمک کاوشگرهای سامانه، رویدادهای مورد نظرش را دریافت و اطلاعات را جمع‌آوری می‌نماید. سایر گام‌های حلقه‌ی کنترل را خود مؤلفه و از طریق منطق پیاده سازی شده در کد جنبه‌گرایی آن اجرا می‌نماید.

سطوح دوم و سوم تطبیق‌پذیری در این چارچوب مشابه بوده و فقط در منبع دریافت رویداد متفاوت می‌باشند [۴۰]. در این سطوح حلقه‌ی کنترل با دریافت رویدادی خاص توسط یک سرپرست آغاز می‌شود که از جانب یک مؤلفه‌ی تحت سرپرستی یا یک سرپرست دیگر می‌باشد. در حالت اول انجام روال تطبیق ممکن همگام یا ناهمگام باشد. در تطبیق همگام مؤلفه‌ای که کاوشگرِ آن، رویداد را به سرپرست خوشه داده منتظر می‌ماند تا تطبیق انجام شود و سپس اجرای منطق حرفه را ادامه می‌دهد. در این حالت تطبیق باید بلافاصله انجام شود ولی در حالت ناهمگام مؤلفه‌ای که رویداد را فرستاده به اجرایش ادامه می‌دهد و روال تطبیق در زمان دیگری انجام خواهد شد. در این چارچوب ارتباطات بین سرپرستان تماماً از نوع ناهمگام در نظر گرفته شده است [۴۰].

در بین این تطبیق‌ها، سطح اول اولویت بالاتری داشته و اگر در میانه‌ی اجرای یک حلقه‌ی کنترل دیگر، حلقه‌ی کنترل در سطح فدراسیون (سطح اول) اجرا شود، بایستی حلقه اول متوقف شده و بعداً از ابتدا اجرا شود. (شکل ۳-۶ حلقه‌ی کنترل چارچوب DSCB) ساختار این حلقه‌های کنترل را نشان می‌دهد.



شکل ۳-۶: حلقه‌ی کنترل چارچوب DSCB [۴۰]

۳-۷- چارچوب MOCAS

این چارچوب در سال ۲۰۰۹ میلادی توسط باربیر[۱]، همورلین[۲] و بالاگنی[۳] ارائه شده که جزءِ معدود چارچوب‌هایی است که به تفصیل در خصوص جزئیات عناصرش توضیح داده شده است. این چارچوب در حقیقت یک مدل مؤلفه‌ی بر مبنای حالت[۴] بوده و متکی بر تطبیق‌پذیری رفتاری به جای تطبیق‌پذیری ساختاری می‌باشد. در این چارچوب هر مؤلفه حاوی یک ماشین حالت UML برای عینیت بخشی به رفتارش، هنگامیکه در سامانه اجرا می‌شود، می‌باشد. این ماشین حالت در یک نگهدارنده[۵] نصب می‌شود که فرآیند تطبیق را مدیریت می‌نماید و سازگاری[۶] سامانه را در طول این فرآیند تضمین می‌نماید [۱۱]. "در این چارچوب محرک اصلی آغاز فرآیند تطبیق سامانه، نقض شدن ثابت‌هایی مرتبط با صفات حرفه‌ی مؤلفه‌ها می‌باشد" [۱۱] بدین معنا که قوانین مشخصی در ازای هر مؤلفه تعریف شده و هر مؤلفه در برابر این قوانین مورد نظارت قرار می‌گیرد.

۳-۷-۱- طرح کلی چارچوب

در این چارچوب رفتار مؤلفه‌ها بوسیله‌ی ماشین حالت UML[۷] مشخص می‌شود. مؤلفه‌ها هرکدام در یک نگهدارنده قرار گرفته و این نگهدارنده‌ی حالت‌محور[۸] به عنوان واسطه‌ای برای تغییر رفتار مؤلفه که از محیط بیرون ناآگاه است عمل می‌نماید.

۳-۷-۱-۱- ماشین حالت UML

در این چارچوب رفتار هر مؤلفه توسط یک ماشین حالت UML – که رفتار MOCAS[۹] نام دارد - مشخص می‌شود. هر مؤلفه ماشین حالت مخصوص به خود را دارد. در طراحی این ماشین حالت شباهت زیادی به شبکه‌های پتری به چشم می‌خورد. هر ماشین حالت مجموعه‌ای از حالات متصل از طریق انتقالات است. هر انتقال دارای یک

---

۱ Franck Barrier
۲ Nabil Hameurlain
۳ Cyril Ballagny
۴ State-based
۵ Container
۶ Consistency
۷ UML state machine
۸ State-based
۹ MOCASBihavior

سیگنال ورودی، یک محافظ[۱] و یک تأثیر[۲] می‌باشد. انتقال وقتی صورت می‌پذیرد که سیگنال ورودی فعال باشد و محافظ که در حقیقت یک عبارت شرطی است به "درست" ارزیابی شود. هر اثر نیز یک عمل درونی و یک سیگنال برای ارسال است. هر حالت دارای عملیات درونی برای ورودی، انجام و خروجی است که مشابه نمودار حالت در UML عمل می‌نمایند. در نهایت هر حالت دارای ثابت‌هایی است که مربوط به صفات حرفه بوده و به وسیله‌ی زبان محدودیت شیء[۳] تشریح می‌شوند. این صفات نیز به صورت عبارات شرطی بیان شده و هنگامیکه یک حالت فعال است، صفات مربوط به آن نیز بایستی همیشه به "درست" ارزیابی شوند.

ارتباط بین مؤلفه‌ها به صورت نامتقارن[۴] و از طریق سیگنال برقرار می‌شود. هر سیگنال پس از رسیدن به مؤلفه در انتهای صف پردازش سیگنال باقی می‌ماند تا سایر سیگنال‌ها یکی یکی پردازش شوند.

۳-۷-۲- مدل مؤلفه‌ها

MOCAS یک مدل مؤلفه بر مبنای حالت است که این مدل سه نوع مؤلفه زیر را دربر می‌گیرد:

۳-۷-۲-۱- مؤلفه‌های حرفه

وظیفه‌ی اصلی این مؤلفه‌ها ارسال و دریافت سیگنال است. این مؤلفه‌ها دارای صفات حرفه هستند (صفاتMOCAS)[۵] و اعمال داخلی حرفه را (که در واسط "اعمالMOCAS"[۶] مشخص می‌شوند) به انجام می‌رسانند که این اعمال شامل محاسبات روی صفات حرفه و تغییر (شکل) آن‌هاست. کلاسی که این اعمال را درون مؤلفه انجام می‌دهد "زمینه‌ی‌اجراییMOCAS"[۷] نام دارد.

۳-۷-۲-۲- مؤلفه‌های تطبیق‌پذیر

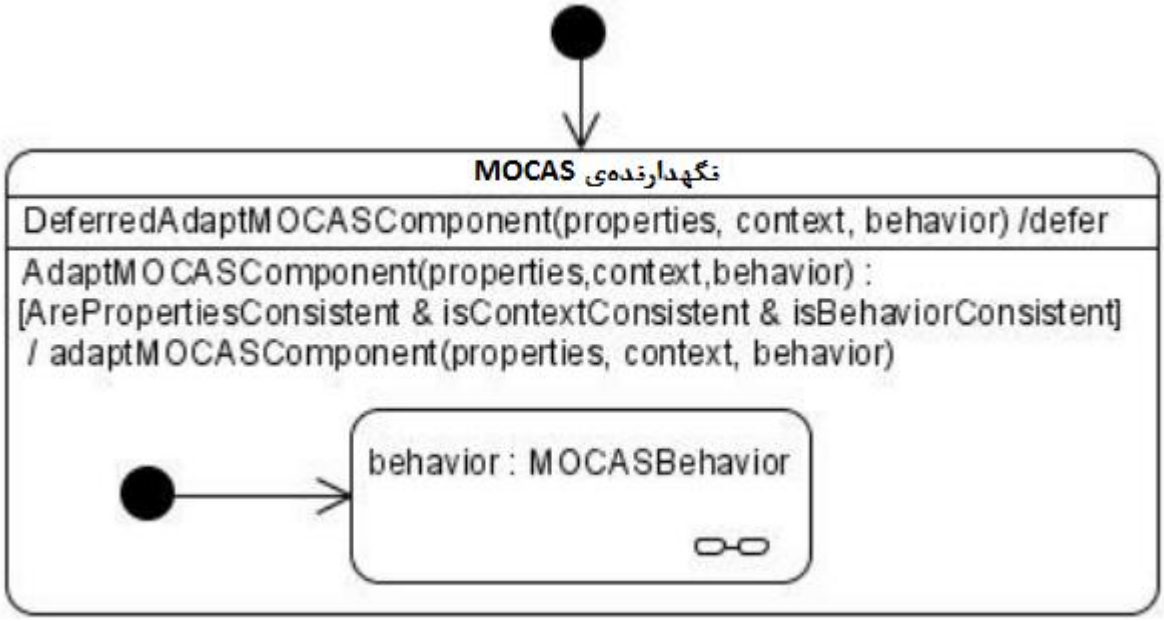
---

۱ Guard
۲ Effect
۳ Object Constraint Language (OCL)
۴ Asynchronous
۵ MOCASProperties
۶ MOCASAction
۷ MOCASFunctionalContext

در این چارچوب با قرار دادن مؤلفه‌های حرفه در داخل یک نگهدارنده، مؤلفه تطبیق‌پذیر بدست می‌آید. این نگهدارنده واسط بین مؤلفه و محیط خارجی آن بوده و مانند سایر مؤلفه‌های این چارچوب دارای ماشین حالت UML درونی بوده (شکل ۳-۷: نگهدارنده‌ی MOCAS) تا فرآیند تطبیق را مدیریت نماید. تطبیق مؤلفه توسط نگهدارنده‌ی آن انجام می‌شود و با دریافت سیگنال "تطبیق‌مؤلفهMOCAS"[۱] آغاز می‌گردد. از آن‌جا که این سیگنال مانند سایر سیگنال‌ها بایستی در صف پردازش سیگنال قرار بگیرد، سایر مؤلفه‌ها این امکان را دارند تا از تطبیق احتمالی آگاه شوند.

ماشین حالت نگهدارنده از موارد زیر استفاده می‌نماید:

- یک زیرماشین حالت که ماشین حالت رفتار مؤلفه‌ی حرفه‌ی درون نگهدارنده را نگهداری می‌کند.
- سیگنال "تطبیق‌مؤلفهMOCAS" و تخصیص‌های[۲] آن که انتقال داخلی بین حالات مختلف



شکل ۳-۷: نگهدارنده‌ی MOCAS [۱۱]

"نگهدارنده‌یMOCAS" را تحریک می‌نماید. این سیگنال دارای سه ویژگی است:

---

۱ AdaptMOCASComponent
۲ Specialization

- ارجاعی به یک مجموعه "رفتارMOCAS" و یا "تطبیق‌مؤلفه‌یMOCAS" و یا "صفاتMOCAS" جدید
- یک محافظ انتقال داخلی که سازگاری عناصر "مؤلفه‌ی MOCAS" جدید را با نمونه‌ی قبلی بررسی می‌نماید.
- رفتار "تطبیق‌مؤلفهMOCAS" که ارجاعات سیگنال "تطبیق‌مؤلفه‌یMOCAS" را عوض می‌نماید.

پس از اینکه سیگنال "تطبیق‌مؤلفهMOCAS" از صف پردازش سیگنال برداشته شد مراحل زیر دنبال می‌شود:

۱- ماشین حالت نگهدارنده با بررسی محافظ، امکان‌پذیری تطبیق را بررسی می‌کند. اگر محافظ "نادرست" ارزیابی شود و سیگنال "تعویق‌تطبیق‌مؤلفهMOCAS"[۱] یک تخصیص باشد، پس از تغییر حالت مؤلفه این بررسی مجدداً انجام می‌گیرد. اگر محافظ "درست" ارزیابی شود، عمل داخلی "تطبیق‌مؤلفهMOCAS" اجرا می‌گردد.

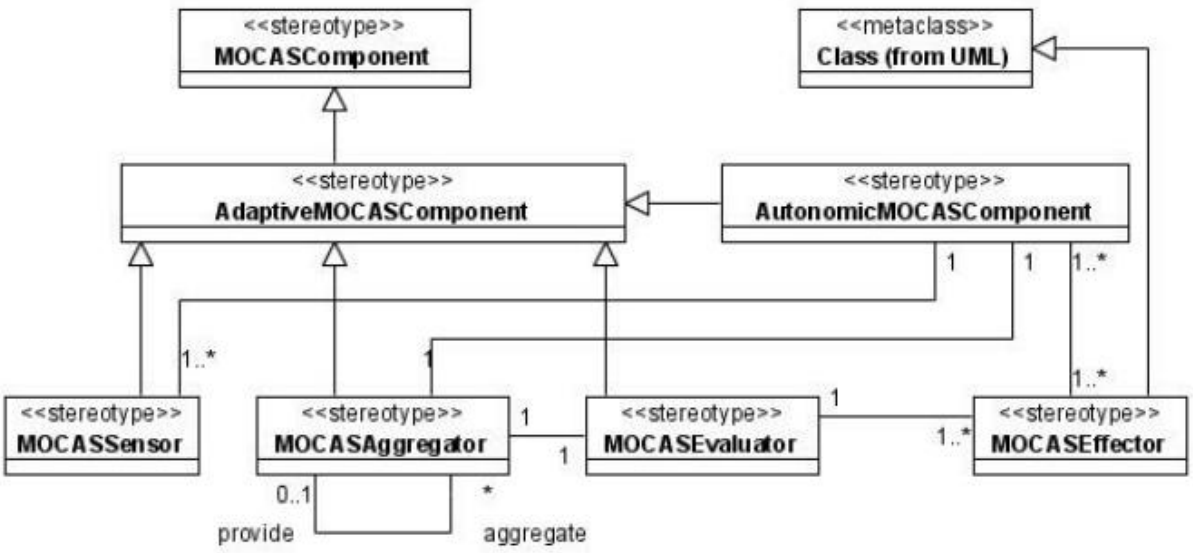
۲- عمل داخلی "تطبیق‌مؤلفهMOCAS" ابتدا مجموعه‌ی صفات را با مقادیر جدید درون سیگنال به‌روزرسانی کرده و ارجاع زمینه‌ی ارجاعی را با نمونه‌ی جدید آن در سیگنال تعویض می‌نماید. در نهایت نیز ارجاع ماشین حالتِ در اختیار نگهدارنده را با نمونه‌ی جدید درون سیگنال عوض می‌کند. البته با جدا بودن منطقی این موارد لزوماً هربار این سه تعویض با هم انجام نمی‌پذیرند [۱۱].

فرآیند تطبیق از حالت فعال مؤلفه نگهداری می‌کند تا در حال اجرای این فرآیند اعمال مربوط به حالت (ورود، اجرا، خروج) اجرا نشوند.

۳-۷-۲-۳- مؤلفه‌های خودمختار

[۱] DefferedAdaptMOCASComponent

هر "مؤلفه‌ی خودمختار MOCAS" دارای یک حلقه‌ی کنترل بوده که امکان خودتطبیقی را فراهم می‌آورد. این مؤلفه‌ها یک تخصیص از مؤلفه‌های تطبیق‌پذیر به حساب می‌آیند (شکل ۳-۸: مؤلفه‌ی خودمختار MOCAS).

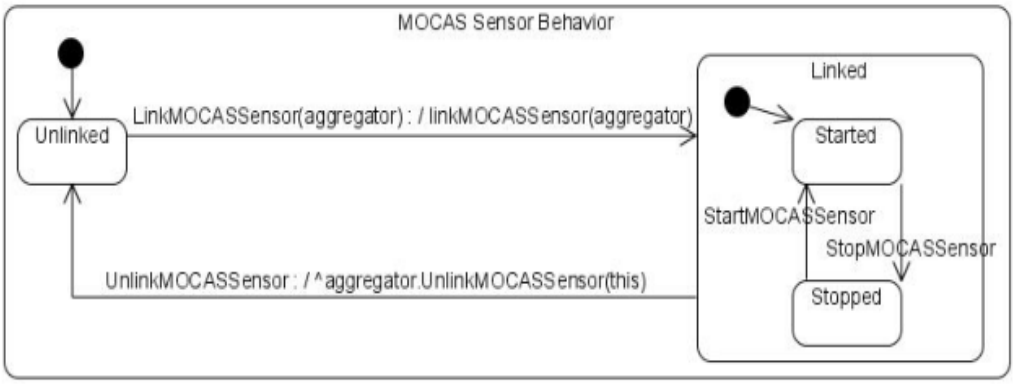


شکل ۳-۸: مؤلفه‌ی خودمختار MOCAS [۱۱]

۳-۷-۳- حلقه‌ی کنترل در **MOCAS**

حلقه‌ی کنترل در MOCAS با اتصال عناصر مختلفی به مؤلفه‌های خودمختار ایجاد شده و امکان نظارت، ارزیابی و تطبیق را برای آن فراهم می‌آورند.

"حسگر MOCAS" یک مؤلفه‌ی تطبیق‌پذیر است. هر حسگر در زمان اجرا تنها به یک مؤلفه‌ی خودمختار متصل می‌شود که ممکن است دارای چندین حسگر باشد. حسگرها می‌توانند موارد مختلفی از قبیل بار پردازنده، حافظه‌ی مصرفی، دریافت سیگنال، تغییر در صفات را نظارت نمایند. گرچه امکان خاص‌سازی رفتار حسگرها وجود دارد ولی طرح کلی آن در (شکل ۳-۹: رفتار یک حسگر MOCAS) آمده است.



شکل ۳-۹: رفتار یک حسگر MOCAS [۱۱]

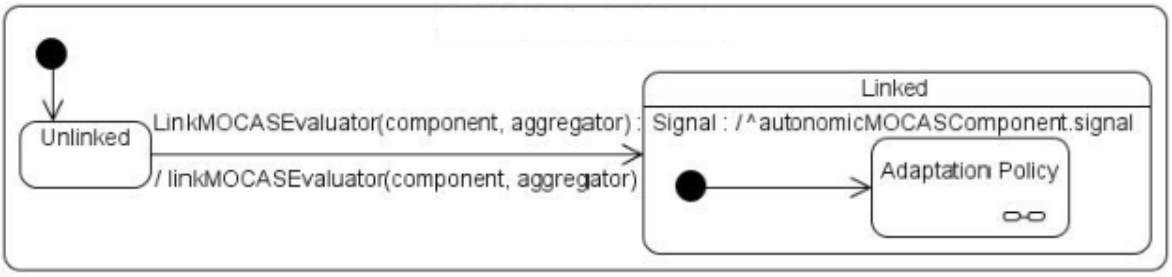
رفتار یک حسگر می‌تواند فعالانه یا منفعلانه باشد. اگر رفتار یک حسگر فعالانه باشد، حسگر به طور مستمر از سامانه سرکشی نموده که ممکن است سربار زیادی تحمیل نماید. ولی در رفتار منفعلانه سیگنال تنها زمانی فعال می‌شود که سیگنال به خصوصی رخ دهد. اگر در هنگام پردازش یک سیگنال توسط مؤلفه‌ی خود مختار یک استثنا رخ دهد، یک سیگنال شکست به حسگر متصل به آن ارسال می‌شود. هنگامیکه حسگر متوجه یک رویداد غیرطبیعی شود سیگنالی را به "مؤلفه‌ی‌مجتمعMOCAS"[۱] (جلوتر تشریح خواهد شد) و مؤلفه‌ی خودمختار متصل به آن ارسال می‌نماید. همچنین می‌توان با سیگنال "توقف‌حسگرMOCAS"[۲] حسگر را جهت آزادسازی منابع غیرفعال نمود.

"مؤلفه‌ی‌مجتمعMOCAS" اطلاعات دریافتی از حسگرهای متصل به آن را متمرکز نموده و می‌تواند این اطلاعات را در اختیار سایر "مؤلفه‌های مجتمعMOCAS" قرار دهد. از آن‌جا که خود "مؤلفه‌ی مجتمعMOCAS" یک "مؤلفه‌ی تطبیق‌پذیرMOCAS" است، امکان تکامل رفتارش برای پشتیبانی از سیگنال‌های جدید احتمالی در زمان آینده وجود دارد. "مؤلفه‌ی مجتمعMOCAS" به طور پیش‌فراض تمام سیگنال‌های دریافتی را به "ارزیابMOCAS"[۳] منتقل می‌نماید تا سایر مؤلفه‌های خودمختار بتوانند در صورت نیاز به آن دسترسی داشته باشند.

---

[۱] MOCASAggregator
[۲] StopMOCASSensor
[۳] MOCASAnalyser

"ارزیابMOCAS" یک "مؤلفه‌ی تطبیق‌پذیرMOCAS" است که سیاست‌های تطبیق مؤلفه‌ی خودمختار متصل به آن را نگهداری و بر آن‌ها نظارت می‌کند. در این چارچوب هر سیاست تطبیق به صورت یک ماشین حالت پیاده-سازی می‌شود که طرز[۱] عمل مؤلفه‌ی خودمختار را بر اساس شرایط محیطی تعریف می‌نماید. پیش از آن‌که یک سیگنال توسط مؤلفه‌ی خودمختار پردازش شود در اختیار "ارزیاب MOCAS" قرار می‌گیرد تا بررسی شود که آیا طرز عمل مؤلفه‌ی خودمختار کماکان برای محیط مناسب است یا خیر که اگر چنین نباشد تغییر رخ می‌دهد. از آن‌جا که ارزیاب یک "مؤلفه‌ی تطبیق‌پذیرMOCAS" است امکان افزودن سیاست‌های تطبیق جدید به آن وجود دارد. رفتار "ارزیاب MOCAS" در (شکل ۳-۱۰: رفتار ارزیاب MOCAS) نشان داده شده است.



شکل ۳-۱۰: رفتار ارزیاب MOCAS [۱۱]

"مجری MOCAS" شیئی است که اعمال تطبیق را انجام می‌دهد. "مجری MOCAS" به صفات مؤلفه‌ی خودمختاری که به آن متصل شده و همینطور صفاتی از سیگنال که باعث شروع فرآیند تطبیق می‌شوند دسترسی دارد. "مجری MOCAS" قابل وقفه[۲] نیست و اگر خطای در فرآیند تطبیق رخ دهد توسط حسگرها قابل شناسایی است و فرآیند تطبیق جدیدی می‌تواند آن را اصلاح نماید. طراح باید از پایان‌پذیر بودن مجری مثلاً در صورت رسیدن به سقف مجاز شکست یا امتحان کردن تمام راه‌حل‌های ممکن اطمینان حاصل نماید [۱۱].

---

[۱] Mode
[۲] Interruptible

۳-۸- چارچوب کریشنامورتی[۱] و بابو[۲]

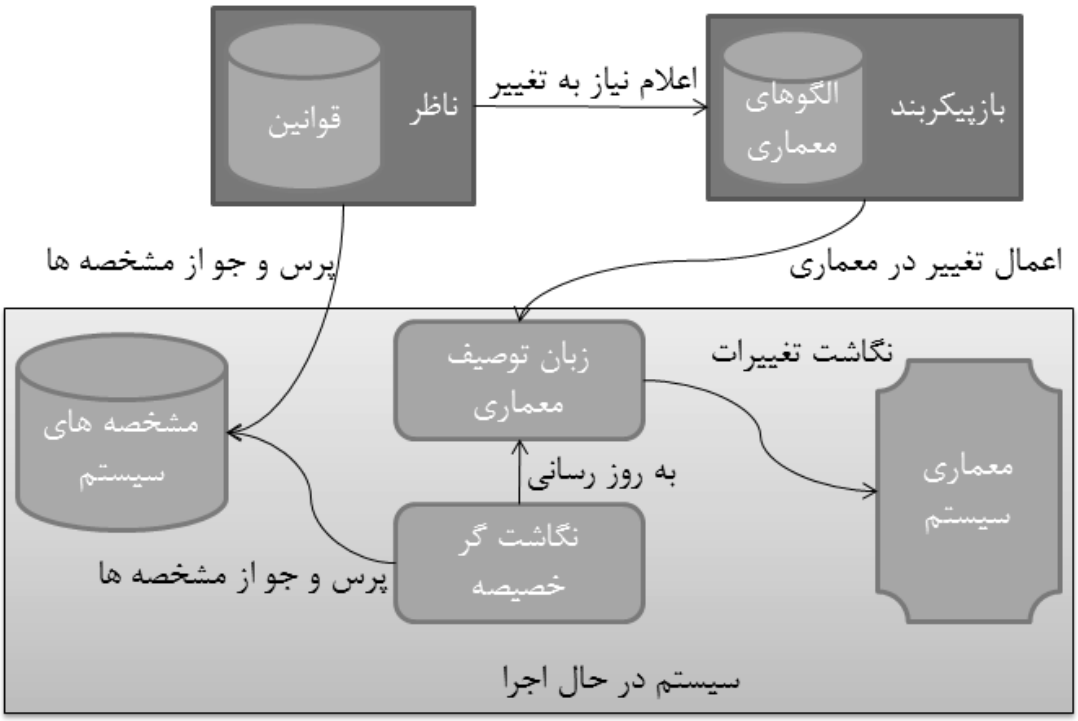
این چارچوب در سال ۲۰۱۰ میلادی ارائه شده است. در این چارچوب، الگوهای معماری سامانه‌های خودتطبیق با چارچوب اصلی سامانه‌های خودتطبیق ترکیب شده و به سامانه اجازه می‌دهد بر اساس الگوهای معماری از پیش تعریف شده که کارایی خود را اثبات نموده‌اند [۵]، معماری خود را تغییر دهد. این تغییرات سپس به سامانه در حال اجرا منعکس می‌شوند. تاکنون الگوهای معماری متفاوتی برای عناصر مختلف سامانه‌های خودتطبیق ارائه شده است که در [۱۱] دوازده مورد مختلف از این الگوها مورد بررسی قرار گرفته است.

۳-۸-۱- طرح کلی چارچوب

شیوه‌ی کار این چارچوب به این صورت است که ابتدا مشخصه[۳]های سامانه در حال اجرا جمع‌آوری شده و بوسیله‌ی نگاشت‌گر خصیصه[۴] در زبان توصیف معماری بروزرسانی می‌گردد. این خصیصه‌ها توسط ناظر خارجی و قوانین تعریف شده در آن مورد بررسی قرار می‌گیرند. هر وقت قانونی نقض شده باشد، واحد تصمیم‌گیری اقدام به تغییر در پیکربندی سامانه می‌نماید که این بازپیکربندی بر اساس الگوهای معماری انجام خواهد گرفت. بنابراین انتخاب الگوهای معماری مناسب در این سامانه حیاتی است. در پایان تغییرات انجام گرفته در زبان توصیف معماری به سامانه در حال اجرا نگاشت می‌شود. در (شکل ۳-۱۱: نمای کلی چارچوب کریشنامورتی و بابو) ساختار پیشنهادی این چارچوب نشان داده شده است.

---

۱ Vallidevi Krishnamurthy
۲ Chitra Babu
۳ Property
۴ Property mapper



شکل ۳-۱۱: نمای کلی چارچوب کریشنامورتی و بابو [۵]

یکی از مشخصه‌های این چارچوب عدم تعیین شیوه‌ای مشخص برای جمع‌آوری مشخصه‌های سامانه می‌باشد. در طرح کلی این چارچوب هیچ عنصری برای این منظور در نظر گرفته نشده و این مسئله در اختیار طراح است.

۳-۹- چارچوب یانگ[۱] و همکاران

این چارچوب در سال ۲۰۱۰ میلادی ارائه شده است. تفاوت اصلی این چارچوب با سایر چارچوب‌های ارائه شده در این متن استفاده‌ی آن از منطق فازی[۲] می‌باشد. در پیاده‌سازی این چارچوب یک کنترل‌کننده‌ی مرکزی فازی در نظر گرفته شده است که با توجه به اینکه از اطلاعات اکتشافی[۳] برای ساخت استراتژی‌های کنترل سامانه استفاده می‌نماید، می‌تواند از داده‌های غیر دقیق یا به عبارتی مبهم نیز پشتیبانی نماید. کنترل فازی بسیار شبیه استنتاج در

---

[۱] Qiliang Yang
[۲] Fuzzy logic
[۳] Heuristic

انسان عمل می‌نماید، بنابراین برای استفاده در سامانه‌های خودتطبیقی که با اطلاعات مبهم و نامطمئن کار می‌کنند بسیار مفید است [۱].

در [۱] برای این چارچوب – که به عنوان مدل عمومی مطرح شده - یک معماری پیاده‌سازی و همینطور یک متدولوژی پیشنهادی برای توسعه‌ی سامانه خودتطبیق با استفاده از این چارچوب پیشنهاد شده است که ما بر روی چارچوب تمرکز می‌نماییم.

۳-۹-۱- طرح کلی چارچوب

این چارچوب شامل دو لایه است: لایه‌ی منطق انطباق و لایه‌ی سامانه تطبیق‌پذیر. این دو لایه توسط گذرگاه نرم‌افزار در ارتباط هستند.

۳-۹-۱-۱- لایه منطق انطباق

این لایه شامل موتور انطباق و منطق مورد نیاز برای تغییر و تحول سامانه‌ی نرم‌افزاری هدف است. راهبردهای انطباق توسط قوانین فازی عینیت یافته و در یک پایگاه قواعد فازی ذخیره می‌شوند. موتور انطباق با بکارگیری راهکار استنتاج فازی بر روی این قوانین می‌تواند برای انطباق سامانه تصمیم‌گیری نماید.

این لایه دو دسته ورودی مجزا وجود دارد:

۱- اهداف سامانه که معمولاً نرخ تغییر پایینی دارند.
۲- ورودی‌های ماشین[۱] که از حسگرهای سامانه دریافت می‌شوند.

می‌توان گفت این دو دسته ورودی محیط عملیاتی را در این چارچوب تشکیل می‌دهند. اگر بین این دو دسته از ورودی‌ها تناقضی پیدا شود موتور انطباق فعال شده و داوری می‌نماید. نهایتاً خروجی این لایه بصورت غیرفازی-شده[۲]ی استنتاج‌های موتور انطباق می‌باشد.

---

[۱] Plant
[۲] Defuzzified

۳-۹-۱-۲- لایه‌ی سامانه‌ی تطبیق‌پذیر

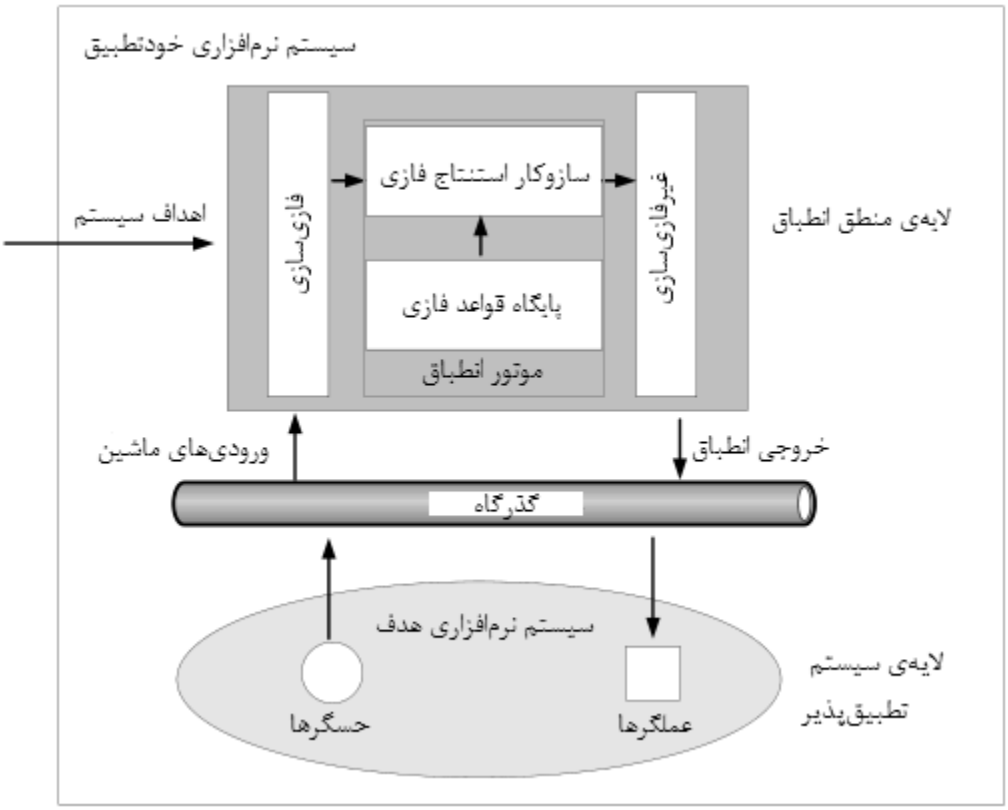
این لایه در حقیقت فراهم‌کننده‌ی کارکرد[۱]های اصلی سامانه‌ی نرم‌افزاری می‌باشد که برای مثال می‌تواند دارای قسمت‌های واسط کاربر، منطق حرفه و دسترسی به داده‌ها باشد. علاوه بر این موارد، این لایه شامل حسگر[۲]ها و عملگر[۳]ها نیز می‌شود. حسگرها وظیفه دارند تا هر آنچه که موتور استنتاج برای انطباق در خصوص وضعیت سامانه لازم دارد را در اختیار آن قرار دهند. عملگرها دستورات لایه‌ی منطق انطباق را اجرایی نموده و تغییرات را به لایه‌ی سامانه‌ی تطبیق‌پذیر اعمال می‌نمایند.

برای ارتباط اجزای مختلف این دو لایه از یک گذرگاه استفاده می‌شود. این گذرگاه به مانند واسطی عمل می‌نماید که بتوان در صورت نیاز این دو لایه را حتی بصورت جداگانه و در دو سامانه اجرا نمود.

در (شکل ۳-۱۲: نمای کلی چارچوب فازی سامانه خودتطبیق) نمای کلی این چارچوب نشان داده شده است.

---

۱ Functionality
۲ Sensor
۳ Actuator



شکل ۳-۱۲: نمای کلی چارچوب فازی سامانه خودتطبیق [۱]

یکی از نقاط ضعف این چارچوب تعریف نشدن سطح تطبیق‌پذیری سامانه و میزان توانایی عملگرها به اعمال تغییرات در سامانه می‌باشد.

۳-۱۰- نتیجه‌گیری

در این فصل ابتدا به دسته‌بندی انواع چارچوب‌های معرفی شده در حوزه سامانه‌های خودتطبیق پرداخته شد و هفت چارچوب از چارچوب‌های توسعه‌ی سامانه‌های خودتطبیق مورد بررسی قرار گرفت. می‌توان نشان داد تطبیق ساختاری (در مقابل تطبیق رفتاری) از پشتیبانی بیشتری در میان چارچوب‌های مورد بررسی برخوردار می‌باشد.

۴- فصل چهارم: مدل پیشنهادی

۴-۱- مقدمه

در این بخش مدل تصمیم‌گیری برای انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی خودتطبیق ارائه خواهد شد. بدین منظور نیاز است در ابتدا مجموعه‌ای از معیارها معرفی شده و چارچوب‌های مختلف بر اساس آن‌ها مورد ارزیابی قرار بگیرند. در ادامه‌ی این فصل بخش‌های اصلی مدل تصمیم‌گیری طراحی شده، معرفی شده و در پایان با استفاده از موردکاوی، کاربردپذیری این مدل بررسی می‌شود.

۴-۲- صفت‌یابی چارچوب‌ها

در طول دهه‌ی گذشته، قابلیت‌های پویای سامانه‌های خودتطبیق بهبود و گسترش زیادی داشته است. از همین روی نیاز زیادی به شیوه‌ها و ابزارهایی برای ارزیابی ویژگی‌های خودتطبیقی این سامانه‌ها در زمان طراحی و همینطور زمان اجرا وجود دارد. با این حال هنوز ابزارها، روش‌ها، معیارها و چارچوب‌های جامع و واسعی برای ارزیابی این سامانه‌ها توسعه نیافته است. در حقیقت گرچه راهکارها، چارچوب‌ها و تکنیک‌های خودتطبیقی در مهندسی سامانه‌های پویا مورد تحقیق و بررسی قرار گرفته است اما به کارگیری این موارد در مسائل واقعی به علت فقدان روش‌های اعتبارسنجی[۱] و صحت‌سنجی[۲] کماکان محدود است [۱۳].

واضح است که برای بهره‌گیری از قابلیت‌ها و مزایای سامانه‌های خودتطبیق لازم است راهکارهای تطبیق اعتبارسنجی شوند تا از صحت عملکرد آن‌ها اطمینان حاصل شود و کاربران سامانه به آن‌ها اطمینان نمایند. باید به این نکته توجه داشت که ارزیابی‌های رایج و مبتنی بر معیارها و شاخص‌ها برای سامانه‌های در حال استفاده قابل انجام می‌باشند. بنابراین در اینجا ما به روش مورد استفاده برای صفت‌یابی چارچوب‌های معرفی شده می‌پردازیم.

---

۱ Validation
۲ Verification

۴-۲-۱- روش صفت‌یابی

این روش که توسط ویلِگاس[۱]، تامورا[۲] و کاسالاس[۳] در [۱۳] ارائه شده است؛ توسط طراحان آن به عنوان یک چارچوب مطرح شده است که با توجه به عدم ارائه‌ی ابزار برای این روش به نظر نمی‌رسد که بتوان آن را یک چارچوب قلمداد نمود. این روش برای صفت‌یابی سامانه‌های خودتطبیق به کار می‌رود. در این روش ویژگی‌های خودتطبیقی به صراحت تعریف شده و بوسیله‌ی صفات کیفی معرفی شده توسط مؤسسه‌ی مهندسی نرم‌افزار هدایت می‌شوند. این روش موارد زیر را فراهم می‌آورد[۱۳]:

- ابعادی سودمند برای دسته‌بندی و طبقه‌بندی سامانه‌های خودتطبیق
- مختصری از صفات خودتطبیقی که توسط حلقه‌ی کنترل نظارت می‌شوند
- نگاشت صفات تطبیق به ویژگی‌های کیفی برای ارزیابی این صفات
- مجموعه‌ای از معیارهای کیفی برای ارزیابی صفات تطبیق و ویژگی‌های کیفی؛ برای تعریف این صفات تطبیق، مطالعه‌ای بر روی روش‌های خودتطبیقی موجود و همینطور ویژگی‌های مورد استفاده در نظریه‌ی کنترل صورت گرفته است

در این روش ارزیابی، دو بعد اساسی برای طبقه‌بندی و ارزیابی سامانه‌های خودتطبیق مورد استفاده قرار گرفته است و آن راهبرد کلی مورد استفاده برای فراهم‌سازی خودتطبیقی در منبع مدیریت شونده و همینطور خود مؤلفه خودتطبیق می‌باشد. این دو راهبرد کلی راهبردهای ساختاری و رفتاری نامیده می‌شوند که قبلاً نیز به آن‌ها اشاره شد. در نوع اول معماری سامانه، نحوه‌ی اتصال مؤلفه‌ها و مواردی از این دست تغییر می‌یابند تا سامانه خود را با شرایط جدید تطبیق دهد و در نوع دوم ساختار کلی سامانه دست‌نخورده باقی می‌ماند و تنها رفتار یا عملکرد اجزا یا عناصری از سامانه اصلاح می‌گردد.

جلوتر گزینه‌های مورد استفاده‌ی این روش را برای تحلیل ابعاد مختلف سامانه‌های خودتطبیق معرفی می‌نماییم. مجموعه‌ی این ابعاد امکان منش‌نمایی جامعی را خصوصیات کیفی نرم‌افزار فراهم می‌کند. در هر بعد تحلیل یک

---

۱ Villegas
۲ Tamura
۳ Casallas

سامانه خودتطبیق از مجموعه‌ای از گزینه‌های مرتبط به آن استفاده می‌شود. تمام این گزینه‌ها از نظریه‌ی کنترل استخراج شده‌اند [۴۱]:

- هدف تطبیق (AG): این هدف معمولاً دست‌یابی به یک یا چند ویژگی از ویژگی‌های سامانه‌های خودتطبیق، نگهداشت یک کیفیت خدمات مشخص یا به طور کلی، تنظیم نیازمندی‌های غیروظیفه‌مندی می‌باشد.
- ورودی‌های مرجع (RI): مجموعه‌ای مشخص از مقادیر و نوع‌های متناظرشان برای تشخیص حالت مطلوبی که منبع مدیریت شونده باید از طریق راهکارهای خودتطبیقی به آن دست یابد. این ورودی‌ها می‌توانند در قالب یکی از اشکال زیر حاضر باشند:
  - (RI۱) مقدار مرجع منفرد مثل یک صفات قابل اندازه‌گیری به صورت فیزیکی یا منطقی
  - (RI۲) نوعی قرارداد مانند کیفیت خدمات درخواستی
  - (RI۳) توافق در سطح سرویس
  - (RI۴) محدودیت‌هایی که وضعیت یا حالت پردازشی را مشخص می‌نمایند که مبتنی بر یک تعریف واضح از حالت ارائه می‌شوند
  - (RI۵) نیازهای وظیفه‌مندی که مثلاً به شکل عبارت‌های منطقی بیان می‌شوند
- خروجی‌های اندازه‌گیری‌شده (MO): مجموعه‌ی مقادیر خروجی و نوع‌های آن‌ها که در منبع مدیریت شونده اندازه‌گیری می‌شوند؛ این مقادیر با ورودی‌های مرجع برای بررسی دستیابی به حالت مطلوب مقایسه می‌شوند. در این زمینه، ذکر مشخصات خروجی‌هایی برای اندازه‌گیری و همینطور نحوه‌ی نظارت بر آن‌ها اهمیت دارد. مشخصات خروجی‌ها می‌تواند شامل موارد زیر باشد:
  - (MO۱) جریانی دنباله‌دار از یک داده یا سیگنال خاص
  - (MO۲) عبارات شرطی یا منطقی برای وضعیت‌های قرارداد
  - (MO۳) شروطی که بیان‌کننده‌ی سوءِ عمل سامانه می‌باشند

نحوه‌ی نظارت بر این خروجی‌ها شامل موارد زیر می‌شود:

- (MO۴) اندازه‌گیری صفات فیزیکی از سخت‌افزار مانند دمای پردازنده
- (MO۵) سنجش عناصر منطقی محاسباتی مانند بار پردازنده یا زمان پردازش یک درخواست
- (MO۶) سنجش شروط زمینه‌ی خارجی مانند مکان جغرافیایی کاربر یا وضع آب و هوا

- اعمال کنترلی محاسبه‌شده (CA): این گزینه در حلقه‌ی کنترل و به ویژه طبیعت و ماهیت خروجی واحد برنامه‌ریزی یا تحلیل‌گر مؤلفه خودتطبیق مشخص می‌شود و شامل موارد زیر است:
  - (CA۱) سیگنال دنباله‌دار که بر ویژگی‌های رفتاری منبع مدیریت شونده اثرگذار است
  - (CA۲) اعمالی گسته که بر زیرساخت محاسباتی مجری منبع مدیریت شونده اثر می‌گذارد مانند عملیات تغییر اندازه و اختصاص حافظه در میان‌گیر[۱] سامانه میزبان یا تغییر زمان‌بندی اجرای پردازه‌ها در پردازنده
  - (CA۳) اعمالی گسسته که مستقیماً بر پردازش‌های منبع مدیریت شونده اثرگذارند مانند احضار سرویس‌های سطح پردازه یا توقف، ازسرگیری، خواباندن، تکثیر، تغییر اولویت پردازه‌ها
  - (CA۴) اعمال گسسته‌ای که بر معماری یا ساختار منبع مدیریت شونده اثرگذارند مانند تغییر پیکربندی آن
- ساختار سامانه (SS): این گزینه اشاره به مدل‌سازی سامانه دارد که هم برای مؤلفه خودتطبیق که قابلیت خودتطبیقی را برای سامانه به ارمغان می‌آورد و هم برای منبع مدیریت شونده انجام می‌شود تا روشی که برای خودتطبیقی استفاده شده مشخص شود. این روش‌ها شامل دو دسته می‌شود:

---

[۱] Buffer

- (SS۱) مدل‌سازی منبع مدیریت شونده برای اثرگذاری بر رفتار آن از طریق تغییر ساختارش
- (SS۲) مدل‌سازی منبع مدیریت شونده برای اثرگذاری مستقیم بر رفتار آن

برای دستیابی به چنین هدفی نیاز به مدل‌سازی رفتار و ساختار منبع مدیریت شونده وجود دارد. در خصوص گزینه‌های موجود برای مدل‌سازی ساختار یک مؤلفه خودتطبیق می‌توان موارد زیر را نام برد:

- (SS۳) یک حلقه‌ی بازخورد شامل عناصر مؤلفه خودتطبیق
- (SS۴) کنترل قابل‌تطبیق به همراه ارجاع یا مدل‌های شناسایی[۱] رفتار منبع مدیریت شونده
- (SS۵) کنترل قابل بازپیکربندی شامل ساختار مؤلفه خودتطبیق با الگوریتم کنترل-کننده‌ی قابل‌تغییر مانند یک مؤلفه خودتطبیق بازپیکربندی معماری نرم‌افزار مبتنی بر قانون

برای مدل‌سازی ساختار منبع مدیریت شونده گزینه‌های زیر وجود دارند:

- (SS۶) غیرقابل تغییر
- (SS۷) قابل‌تغییر، همراه یا بدون توانایی استنتاج

- صفات تطبیق قابل مشاهده (AP): این صفات در حقیقت ویژگی‌هایی هستند که در فرآیند تطبیق سامانه قابل شناسایی و اندازه‌گیری می‌باشند. این ویژگی‌ها در مؤلفه خودتطبیق شامل موارد زیر می‌شود:
  - (AP۱) ثبات و پایداری: اشاره به درجه‌ی همگرایی و متمرکز شدن یک فرآیند تطبیق به یک هدف کنترلی دارد. یک فرآیند تطبیق ناپایدار ممکن است با اجرای بی‌پایان عمل کنترلی، وضعیت سامانه را با توجه به شرایط محیطی بهبود نبخشد یا حتی بدتر نماید.

---

[۱] Identification

- (AP۲) دقت: این ویژگی نشان‌دهنده‌ی دستیابی به هدف تطبیق با قبول توجه به حد قابل قبول خطاست و نشان‌دهنده‌ی نزدیکی سامانه به حالت مطلوبست.
- (AP۳) زمان سکون[۱]: بیان‌گر زمان مورد نیاز سامانه در رسیدن به حالت مطلوبست. افزایش این زمان ممکن است باعث ناپایداری سامانه و قرار گرفتن در وضعیت‌های بسیار نامطلوب شود.
- (AP۴) بالازدن یا از حد خارج شدن کوچک[۲]: این ویژگی نشان‌دهنده‌ی کارایی استفاده استفاده از منابع پردازشی سامانه در زمان اجرای فرآیند تطبیق می‌باشد. چنانچه بالازدگی وجود داشته باشد، در هنگام اجرای فرآیند تطبیق سامانه، بهروری استفاده از منابع کاهش می‌یابد.
- (AP۵) نیرومندی[۳]: یک فرآیند تطبیق نیرومند است اگر حتی تحت شرایط پیش‌بینی نشده عمل نماید.
- (AP۶) پایان یا فسخ فرآیند تطبیق: این ویژگی نشان‌دهنده‌ی تضمین‌گر پایان‌پذیر بودن فرآیند تطبیق است. همچنین قوانین مورد استفاده برای تطبیق سامانه نباید منجر به بن‌بست آن شوند مثلاً نباید تناقض بین دو قانون منجر به توقف اجرای فرآیند تطبیق شود.
- (AP۷) همخوانی: نشان‌گر جامعیت ساختار و رفتار سامانه، پس از تطبیق می‌باشد. برای این منظور در صورت شکست فرآیند تطبیق سامانه باید به حالت پیشین بازگردد، این فرآیند باید در انزوا انجام شود و نتایج آن پیش از پایان کار برای سایر اجزای سامانه قابل مشاهده نباشد و در صورت موفقیت، نتایج آن در سامانه پایدار بماند.
- (AP۸) قابلیت توسعه: این ویژگی توانایی مؤلفه خودتطبیق در پشتیبانی از تقاضاهای نو را با بکارگیری منابع جدید نشان می‌دهد.

---

۱ Settling
۲ Small overshoot
۳ Robustness

- (AP۹) امنیت: این ویژگی زمانی قابل دستیابی است که تمام داده‌ها، مؤلفه‌های مشترک و همینطور منبع مدیریت شونده از شنود، تغییر غیرمجاز یا تخریب در امان باشند.

۴-۲-۲- نتایج صفت‌یابی چارچوب‌ها

در این بخش به صفت‌یابی چارچوب‌های معرفی شده می‌پردازیم. در رابطه با هدف تطبیق ذکر این نکته مهم است که تمام چارچوب‌های مطرح شده برای پشتیبانی از ویژگی‌های چهارگانه خودتطبیقی قابل بهره‌برداری می‌باشند و مواردی که به عنوان هدف تطبیق در زیر بیان می‌شوند نشان‌دهنده تمرکز اصلی چارچوب و گرایش آن به یکی از این ویژگی‌هاست.

- گارلان و همکاران (Rainbow)

همانطورکه پیشتر اشاره شد، این چارچوب حالتی عمومی داشته و امکان نمونه‌گیری و ساخت چارچوب‌های جدید از آن وجود دارد. بنابراین این چارچوب از اکثر ویژگی‌های مورد استفاده برای صفت‌یابی پشتیبانی می‌نماید.

AG: خودتطبیقی

RI: (RI1 تا RI5) بدون محدودیت قابل پشتیبانی هستند.

MO: (MO1 تا MO6) بدون محدودیت قابل پشتیبانی هستند.

CA: (CA1 تا CA4) بدون محدودیت قابل پشتیبانی هستند.

SS: (SS1, SS2, SS5, SS7) همگی به عنوان ویژگی‌های برتر هر رده قابل پشتیبانی هستند.

AP: ویژگی‌های این رده به پیاده‌سازی سامانه بستگی دارد.

- کریشنامورتی و بابو

AG: خودپیکربندی بر اساس الگوهای معماری از پیش تعریف شده

RI: محدودیت‌های تعریف شده بر روی معماری سامانه امکان پشتیبانی از (RI4) را فراهم می‌آورد.

MO: "مشخصه‌های سامانه‌ی در حال اجرا" و شروط بیان‌کننده‌ی سوءِ عمل سامانه امکان پشتیبانی از (MO3) را فراهم می‌کنند.

CA: اعمال گسسته تغییردهنده‌ی معماری سامانه به کمک واحد بازپیکربندی (CA4)

SS: حفظ مدل معماری سامانه در حال اجرا برای تغییر آن امکان پشتیبانی از (SS1) را فراهم می‌کند. عدم وجود مدل‌های شناسایی رفتار منبع مدیریت شونده و در عین حال استفاده از قوانین، امکان استنتاج پویا در خصوص وضعیت سامانه را پدید می‌آورد (SS5)، امکان تغییر معماری با تغییر زبان توصیف معماری زمان اجرا و مدلسازی برای ایجاد تغییرات ساختاری در منبع مدیریت شونده زمینه‌ساز پشتیبانی از (SS7) می‌گردد.

AP: استفاده از زبان توصیف معماری تضمین‌کننده‌ی تطبیق دقیق خواهد بود (AP2). همچنین امکان تعریف قوانین خاصی برای حالات پیش‌بینی نشده وجود دارد (AP5). سایر ویژگی‌ها به پیاده‌سازی سامانه بستگی دارد.

- یانگ و همکاران

AG: خودتطبیقی، با توجه به عدم محدودیت بر روی عملگرها

RI: با توجه به تعریف ورودی‌های "لایه منطق انطباق" در قالب "اهداف سامانه" امکان پشتیبانی از (RI2, RI3) وجود دارد. امکان پشتیبانی از (RI1) نیز از طرق حسگرها و ماهیت فازی سامانه وجود دارد. سامانه‌های فازی برای نظارت بر روی انواع مقادیر و تکی یا سیگنال‌ها کاربرد دارند.

MO: با توجه به امکان پردازش‌های فازی در این چارچوب می‌تواند از (MO1) برای ورودی "سازوکار استنتاج فازی" بهره جست. محدودیتی جهت پشتیبانی از (MO2) وجود ندارد. چون نوع عمل مشخصی برای سامانه تعریف نمی‌شود، (MO3) قابل پشتیبانی نیست. (MO4 تا MO6) با توجه به ماهیت فازی سامانه قابل پشتیبانی هستند.

CA: با توجه به عدم تعریف دقیق عملگرها و استفاده از یک واحد "غیرفازی‌سازی"، امکان اعمال CA1 تا CA4 وجود دارد.

SS: محدودیتی برای مدل‌سازی "سامانه نرم‌افزاری هدف" تعیین نشده (SS1, SS2)، با توجه به قابلیت بازپیکربندی یک موتور استنتاج فازی و امکان تأثیر آن بر ساختار "سامانه نرم‌افزاری هدف" از (SS5, SS7) نیز پشتیبانی می‌شود.

AP: با توجه به بکارگیری یک پایگاه قواعد فازی، امکان پایداری سامانه در برابر حالات تعریف نشده بسیار افزایش می‌یابد (AP5). سایر ویژگی‌ها به پیاده‌سازی سامانه بستگی دارد.

- جُرجِس و تیلور

AG: با توجه به ذخیره دانش در خصوص مصرف منابع و همینطور امکان تغییر در مدل معماری، این چارچوب از خودبهینه‌سازی و خودپیکربندی پشتیبانی می‌نماید.

RI: با توجه به پشتیبانی از "رویدادهای خارجی" از (RI1) پشتیبانی می‌شود. ویژگی‌های (RI2 تا RI4) از طریق "سیاست‌های تطبیقِ" "پایگاه دانش" قابل پشتیبانی می‌باشند.

MO: با توجه به عدم تعریف دقیق حسگرها امکان پشتیبانی از (MO1 تا MO6) وجود دارد.

CA: با توجه به وظیفه "مدیر تطبیق" در تغییر مدل معماری مؤلفه مدیریت شونده، از (CA4) در این چارچوب پشتیبانی می‌شود.

SS: امکان تغییر در مدل معماری بر اساس قوانین پویا دیده شده است (SS۱, SS5, SS7).

AP: مدیر تکامل معماری همخوانی مدل معماری و پیاده‌سازی آن را تضمین می‌نماید (AP2, AP7). با توجه به امکان جستوجوهای کارآمد در پایگاه دانش از (AP5) پشتیبانی می‌شود. سایر ویژگی‌ها به پیاده‌سازی سامانه بستگی دارد.

- AOA

AG: در این چارچوب تمرکز اصلی بر روی حفظ کیفیت سرویس‌های سامانه و توافقات سطح سرویس می‌باشد (خود بهینه‌سازی).

RI: برای حفظ کیفیت سرویس‌ها از (RI2, RI3) پشتیبانی می‌شود. پشتیبانی از (RI5) از طریق اعمال کدهای جنبه بر "سرویس‌های عمومی" سامانه امکان‌پذیر است.

MO: از (MO2, MO3) برای حفظ کیفیت سرویس و توافقات سطح سرویس پشتیبانی می‌شود.

CA: حفظ کیفیت سرویس و توافقات سطح سرویس از طریق اعمال تغییراتی در سامانه به کمک (CA2, CA3) امکان‌پذیر است. امکان تغییرات "انواع مؤلفه‌ها و اتصالاتِ" "کاربرد" نیز در چارچوب در نظر گرفته شده است (CA4).

SS: با توجه به اعمال تغییرات در "سطح سرویس‌های عمومی" و همینطور ساختار "کاربرد" از (SS1, SS2, SS7) پشتیبانی می‌شود. وجود "جنبه‌های تجمعی"، امکان بررسی مدل‌های رفتاری کاربرد را پدید می‌آورد (SS4).

AP: افزایش کارایی محاسبات و امکان بررسی الگوها و پیش‌بینی رفتار کاربرد با جداسازی "رویدادهای نشانه" و "رویدادهای تطبیق" که حجم پردازش متفاوتی دارند (AP4, AP5). جداسازی محل‌های اعمال تطبیق (سرویس‌های عمومی و مؤلفه‌ها و اتصالات) از طریق جنبه‌گرایی (جداسازی دغدغه‌ها)، قابلیت توسعه سامانه افزایش یافته است (AP8). سایر ویژگی‌ها به پیاده‌سازی سامانه بستگی دارد.

- MOCAS

AG: خودبهینه‌سازی، خودالتیامی و خودحفاظتی از طریق پیام‌رسانی رویدادهای سامانه بین مؤلفه‌های آن.

RI: از آن‌جاکه تطبیق مؤلفه‌ها بر اثر رویدادهای مختلفی رخ می‌دهد که توسط یک "ماشین حالت UML" قابل پشتیبانی است، (RI1 تا RI5) قابل پشتیبانی هستند.

MO: (MO1) از طریق رویدادهای غیرهمزمان متعدد و (MO2 تا MO5) از طریق قوانین مربوط به هر مؤلفه قابل پشتیبانی هستند.

CA: (CA1) از طریق تغییر رفتار هر مؤلفه به کمک "زمینه‌ی اجرایی MOCAS" پشتیبانی می‌شود. تغییر در زمینه‌ی اجرایی MOCAS" امکان پشتیبانی از (CA2) را فراهم می‌آورد.

SS: رفتار مؤلفه‌ها به کمک "نگهدارنده" قابل تغییر است (SS2)، ولی ساختار منبع مدیریت شونده ثابت می‌ماند (SS6). با تعیین یک سری مدل‌های رفتاری برای هر مؤلفه امکان پشتیبانی از (SS4) وجود دارد.

AP: با توجه به اطلاع‌رسانی در خصوص تطبیق، بین مؤلفه‌ها و یا استفاده از یک روش سلسله مراتبی برای اولویت‌دهی به تطبیق‌ها امکان پشتیبانی از (AP1, AP6) فراهم شده است. استفاده از ماشین حالت UML امکان قرار گرفتن مؤلفه در حالت‌های غیر تعریف شده را از بین می‌برد (AP2, AP7). سایر ویژگی‌ها به پیاده‌سازی سامانه بستگی دارد.

- DSCB

AG: خودتطبیقی؛ این چارچوب به ویژگی‌های مختلف خودتطبیقی نگاه یکسانی دارد.

RI: محدودیتی بر روی حسگرها و نحوه‌ی جمع‌آوری اطلاعات در سطح سامانه وجود ندارد (RI1 تا RI4). همچنین با توجه به خوشه‌بندی و توزیع‌شدگی وظیفه‌مندی‌ها امکان پشتیبانی از (RI5) نیز وجود دارد.

MO: با توجه به وابستگی به رویدادها (اشتراک "سرپرست" برای رویدادهای خاص) امکان پشتیبانی از (MO1, MO2) وجود دارد. (MO3) از طریق بررسی الگوی رویدادها در "سرپرست" و "فدراسیون" امکان‌پذیر است. محدودیتی برای پشتیبانی از (MO4, MO5) در این چارچوب وجود ندارد. (MO6) با توجه به عدم تمرکز مؤلفه خودتطبیق و پشتیبانی از یک "توافق‌نامه‌ی اشتراک و اطلاع‌رسانی" قابل پشتیبانی است.

CA: (CA2, CA3) از طریق "تطبیق در گستره‌ی مؤلفه" به کمک جنبه‌گرایی قابل پشتیبانی است. با جابه‌جایی مؤلفه‌ها بین خوشه‌ها، امکان تغییر در مدل استقرار سامانه وجود دارد (CA4).

SS: (SS2) با اجرای کدتطبیق و جابه‌جایی مؤلفه‌ها بین خوشه‌ها و (SS1, SS7) از طریق تطبیق در گستره‌ی خوشه امکان‌پذیر است. (SS3) نیز با پیاده‌سازی چرخه‌ی خودتطبیقی در هر مؤلفه قابل پشتیبانی است.

AP: با توجه به اولویت‌دهی به انواع تطبیق، بن‌بستی در اجرای آن رخ نمی‌دهد (AP6). ماهیت توزیع‌شده و امکان رسیدگی به وقایع مهم در سطح فدراسیون، قابلیت توسعه را افزایش می‌دهد (AP8). سایر ویژگی‌ها به پیاده‌سازی سامانه بستگی دارد.

۴-۲-۳- خلاصه‌ی نتایج صفت‌یابی

جدول زیر حاوی خلاصه نتایج صفت‌یابی صورت گرفته بر روی چارچوب‌های هفت‌گانه مورد بررسی می‌باشد. کدهای استفاده شده در خانه‌های جدول بر اساس موارد تعریف شده در (۴-۲-۱- معرفی روش) مورد استفاده قرار گرفته‌اند.

جدول۴-۱: خلاصه‌ی نتایج صفت‌یابی چارچوب‌های سامانه‌های خودتطبیق

چارچوب	هدف تطبیق (AG)	ورودی‌های مرجع (RI)	خروجی‌های اندازه‌گیری شده (MO)	اعمال کنترلی محاسبه شده (CA)	ساختار سامانه (SS)	صفات تطبیق قابل مشاهده (AP)
کریشنامورتی و بابو [۵]	خودپیکربندی	RI4	MO3	CA4	SS1,SS5, SS7	AP2,AP5
یانگ و همکاران [۱]	همه موارد	RI1,RI2, RI3	MO1,MO2,MO 4,MO5,MO6	همه موارد	SS1,SS2, SS5,SS7	AP5
جرجس و تیلور [۳۶]	خودبهینه‌سازی، خودپیکربندی	RI1,RI2, RI3,RI4	همه موارد	CA4	SS1,SS5, SS7	AP2,AP5,A P7
AOA [۳۷]	خودبهینه‌سازی	RI2,RI3, RI5	MO2,MO3	CA2,CA3, CA4	SS1,SS2, SS4,SS7	AP4,AP5,A P8
MOCAS [۱۱]	خودبهینه‌سازی،خود حفاظتی،خودالتیامی	همه موارد	MO1,MO2,MO 3,MO4,MO5	CA1,CA2	SS2,SS4, SS6	AP1,AP2,A P6,AP7
DSCB [۴۰]	همه موارد	همه موارد	همه موارد	CA2,CA3, CA4	SS1,SS2, SS3,SS7	AP6,AP8

۴-۳- تصمیم‌گیرندگان

همانطور که در ۲- فصل دوم: ، اشاره شد، سامانه‌های سازمانی از تعداد مؤلفه‌های پرشماری تشکیل شده‌اند، تعداد کاربران زیادی دارند، استقرار و نگهداری آن‌ها ممکن است فرآیند دشواری باشد و لایه‌ی منطق حرفه‌ی آن‌ها پیچیده می‌باشد. در طول فرآیند توسعه‌ی این سامانه‌ها، حضور نمایندگان کارفرما در محیط توسعه بسیار به چشم می‌آید. ذینفعان متعددی در خصوص این سامانه‌ها اظهار نظر می‌نمایند که هر کدام در حوزه‌ای مشخص دارای صلاحیت بوده و مقاصدی را دنبال می‌نمایند. با توجه به این مطالب تصمیم‌گیرندگان این مسئله به دو بخش تقسیم می‌شوند:

- تصمیم‌گیرندگان حوزه‌ی کارفرما: شامل افرادی می‌شود که در توسعه و استقرار سامانه‌های سازمانی نقش مهمی دارند:
  - (CIO) بالاترین مقام فناوری اطلاعات[۱]
  - (CTO) بالاترین مقام فنی فناوری اطلاعات[۲]
  - (CSO) بالاترین مقام امنیت فناوری اطلاعات[۳]
- تصمیم‌گیرندگان حوزه‌ی توسعه‌دهنده: شامل افرادی می‌شود که تأثیر بسزایی در زمان‌بندی و کیفیت نهایی محصول نرم‌افزاری دارند:
  - (PM) مدیر پروژه
  - (SA) معمار نرم‌افزار

در بخش ۲-۵-۳- تصمیم‌گیری گروهی، راهکارهای مشارکت افراد در تصمیم‌گیری‌های گروهی تشریح شد. از آنجا که ارزیابی اولیه‌ی چارچوب‌ها به صورت فنی و دقیق قابل انجام بوده و در حوزه‌ی مسئله‌ی حاضر هم نمی‌گنجد، نوع نظردهی تصمیم‌گیرندگان در قالب تعیین وزن معیارها خواهد بود. همچنین با توجه به یکسان بودن ماهیت راه‌حل‌ها (همگی چارچوب سامانه‌های خودتطبیق هستند) و گوناگونی تخصص افراد تصمیم‌گیرنده (با توجه به جایگاهشان)، نحوه‌ی تأثیر صلاحیت تصمیم‌گیرندگان با مشخص نمودن معیارهای قابل نظردهی برای هر شخص انجام می‌شود. در بخش بعدی معیارهای تصمیم‌گیری و تصمیم‌گیرندگان دارای صلاحیت در خصوص هر معیار معرفی خواهند شد.

۴-۴- معیارها

معیارهای مورد استفاده در مدل تصمیم‌گیری با استفاده از ویژگی‌های مورد استفاده برای صفت‌یابی چارچوب‌ها در فصل سوم، استخراج می‌شوند. هر چارچوب می‌تواند مجموعه‌ای از این صفات را داشته باشد و یا به عبارت دیگر مجموعه‌ای از ویژگی‌ها را ارائه نماید که در اثر آن نیازمندی‌های خاصی ارضا می‌شوند. چنانچه چارچوبی برای توسعه‌ی یک سامانه‌ی خودتطبیق انتخاب شود و یک یا تعداد بیشتری از صفات مورد نیاز را فراهم نکرده باشد، معمار

---

[۱] Chief Information Officer (CIO)
[۲] Chief Technical Officeer (CTO)
[۳] Chief Information Security Officer (CISO)

نرم‌افزار مجبور به طراحی راهکاری برای جبران این نقطه ضعف خواهد بود. این راهکار باعث تغییر در چارچوب خواهد شد. جهت مرور جزئیات معیارها به بخش ۴-۲- صفت‌یابی چارچوب‌ها، رجوع شود.

جدول صفحه‌ی بعد، معیارها و افراد تصمیم‌گیرنده در خصوص وزن آن‌ها و دلیل انتساب هر معیار به افراد مختلف را نشان می‌دهد:

جدول ۴-۲: معیارها و تصمیم‌گیرندگان مدل تصمیم‌گیری

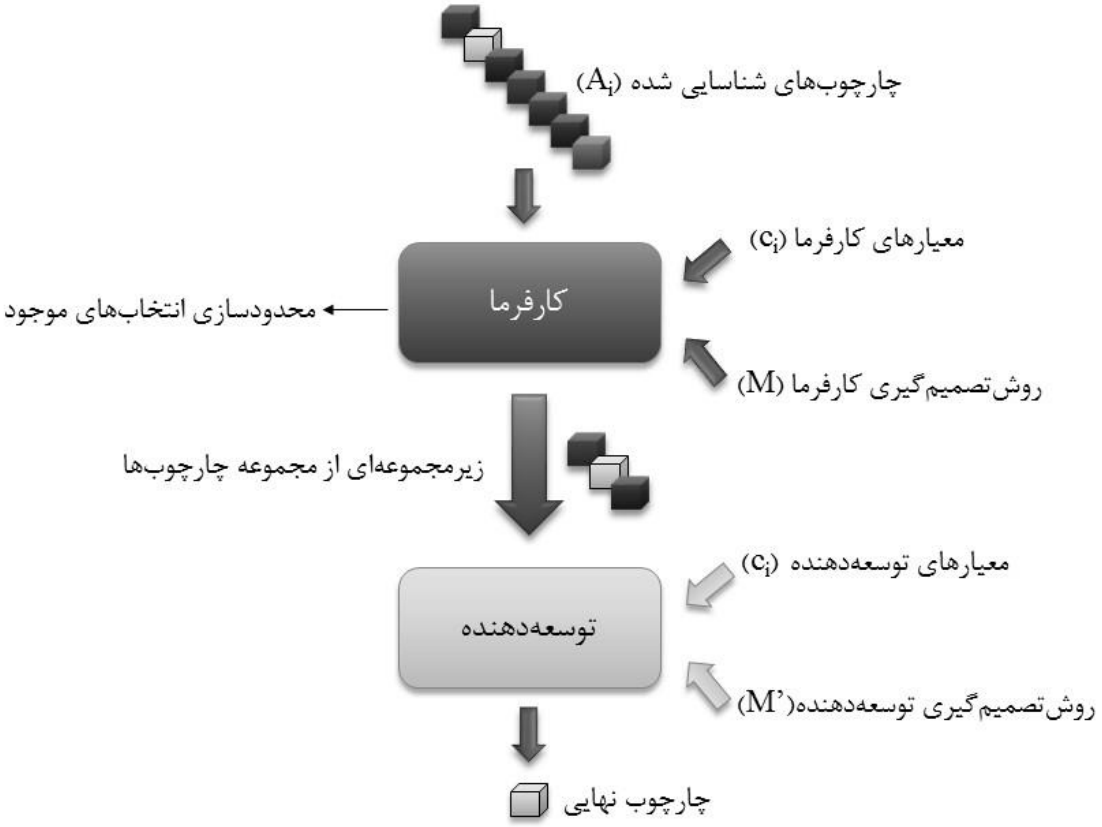
معیار	تصمیم‌گیرندگان در خصوص وزن معیار	دلیل
AG	CIO, CTO, CSO	این معیار بسیار کلیدی بوده و تقریباً امکان توسعه چارچوب بدون پوشش آن امکان ناپذیر است
RI1	SA	کاملاً فنی و در سطح پیاده‌سازی
RI2	CTO, PM, SA	کیفیت خدمات درخواستی بر کل پروژه به لحاظ فنی تأثیرگذار است
RI3	CTO, PM, SA	توافق سطح سرویس بر کل پروژه به لحاظ فنی تأثیرگذار است
RI4	SA	کاملاً فنی و در سطح پیاده‌سازی
RI5	CTO, PM, SA	نیازهای وظیفه‌مندی توسط سازمان قابل شناسایی هستند و بر ماهیت پروژه نیز اثرگذارند
MO1	SA	کاملاً فنی و در سطح پیاده‌سازی
MO2	CTO, SA	کاملاً فنی و در سطح پیاده‌سازی که توسط کارفرما قابل ردیابی است
MO3	CTO, CSO	نیاز امنیتی کارفرما
MO4	SA	کاملاً فنی و در سطح پیاده‌سازی
MO5	SA	کاملاً فنی و در سطح پیاده‌سازی
MO6	CIO, CTO	زمینه‌ی اجرایی سامانه توسط ذینفعان کلیدی آن قابل شناسایی است
CA1	SA	کاملاً فنی و در سطح پیاده‌سازی
CA2	SA	کاملاً فنی و در سطح پیاده‌سازی
CA3	CTO, SA	کاملاً فنی و در سطح پیاده‌سازی که توسط کارفرما قابل ردیابی است
CA4	SA	کاملاً فنی و در سطح پیاده‌سازی
SS1	SA	کاملاً فنی و در سطح پیاده‌سازی
SS2	SA	کاملاً فنی و در سطح پیاده‌سازی
SS3	SA	کاملاً فنی و در سطح پیاده‌سازی
SS4	SA	کاملاً فنی و در سطح پیاده‌سازی
SS5	CTO, SA, PM	موتورهای قانون تأثیر زیادی بر ماهیت پروژه دارند و نیازمند همکاری کارفرما در طراحی هستند (وابستگی به دامنه مسئله در محیط کارفرما)
SS6		این معیار عملاً نقیض SS7 بوده و حذف می‌شود
SS7	SA	کاملاً فنی و در سطح پیاده‌سازی
AP1	CSO, SA	معیار فنی و مؤثر بر امنیت می‌باشد
AP2	CTO, SA	معیار فنی بوده و جزئیات آن (حد قابل قبول خطا) توسط کارفرما قابل شناسایی است
AP3	CTO, SA	معیار فنی بوده و جزئیات آن (زمان مورد نیاز برای تطبیق) توسط کارفرما قابل شناسایی است
AP4	SA	کاملاً فنی و در سطح پیاده‌سازی
AP5	CTO, SA	معیار فنی و مؤثر بر ایمنی می‌باشد
AP6	SA	کاملاً فنی و در سطح پیاده‌سازی
AP7	CTO, CSO	معیار فنی مبتنی بر قابلیت اعتماد سامانه
AP8	CTO, SA	معیار فنی قابل ارزیابی توسط کارفرما
AP9	CSO, SA	معیار فنی و مؤثر بر امنیت می‌باشد

معیار	تصمیم‌گیرندگان در خصوص وزن معیار	دلیل
AP10		در خصوص "سامانه‌های" خودتطبیق در حال اجرا قابل اندازه‌گیری است (حذف می‌شود).
AP11		در خصوص "سامانه‌های" خودتطبیق در حال اجرا قابل اندازه‌گیری است (حذف می‌شود).

۴-۵- فرآیند

در توسعه‌ی سامانه‌های سازمانی، حضور کارفرما در مراحل مختلف توسعه بسیار جدی است؛ چراکه این سامانه‌ها مطابق آنچه که در فصل سوم نیز مورد اشاره قرار گرفت، عمدتاً در سرتاسر سازمان و یا حداقل در چندین واحد سازمانی مورد بهره‌برداری قرار می‌گیرند، طول عمر نسبتاً زیادی دارند، احتمالاً حجم بالایی از داده‌های حیاتی کسب و کار را تولید، استفاده و یا نگهداری می‌نمایند و باید نیازمندی ذینفعان پرشماری در توسعه‌ی آن‌ها مورد توجه قرار گیرد. این مجموعه عوامل سبب همکاری بالای متخصصین حرفه با متخصصین فناوری اطلاعات در امر توسعه‌ی این سامانه‌ها می‌شود.

با توجه به مطالب بالا فرآیند انتخاب چارچوب در دو گام تصمیم‌گیری انجام خواهد شد. نخست تصمیم‌گیرندگان حوزه‌ی کارفرما (شامل CIO، CTO و CSO) تعداد معدودی از چارچوب‌ها را با استفاده از معیارهایی که در بخش قبل تعریف شده است برمی‌گزینند. این تعداد شامل ۸۰٪ کل راه‌حل‌های موجود برای مسئله حاضر یعنی چهار راه‌حل می‌باشد. در گام دوم تصمیم‌گیرندگان حوزه‌ی توسعه‌ی سامانه (شامل SA و PM)، از میان این چارچوب‌های باقیمانده، چارچوب نهایی را شناسایی خواهد نمود (شکل زیر).



شکل ۴-۱: فرآیند تصمیم‌گیری انتخاب چارچوب

جدول زیر معیارهای کارفرما و توسعه‌دهنده را نشان می‌دهد:

جدول ۴-۳: معیارهای حوزه‌ی کارفرما و توسعه‌دهنده

حوزه	تصمیم‌گیرندگان	معیارها (c_i)
کارفرما	CIO	AG, MO6
	CTO	AG, RI2, RI3, RI5, MO2, MO3, MO6, SS5, AP2, AP3, AP5, AP7, AP8
	CSO	AG, MO3, AP1, AP7, AP9
توسعه‌دهنده	SA	RI1, RI2, RI3, RI4, RI5, MO1, MO2, MO4, MO5, CA1, CA2, CA3, CA4, SS1, SS2, SS3, SS4, SS5, SS7, AP1, AP2, AP3, AP4, AP5, AP6, AP8, AP9
	PM	RI2, RI3, RI5, CA3, SS5

۴-۶- روش تصمیم‌گیری

همانطور که در (شکل ۴-۱: فرآیند تصمیم‌گیری انتخاب چارچوب) نشان داده شده، از دو روش تصمیم‌گیری به صورت سِری در این مدل استفاده می‌شود که با ترکیب این دو روش، یک روش عطفی (رجوع شود به: پیوست ۱) بدست می‌آید. تصمیم‌گیرندگان حوزه‌ی کارفرما – که در بخش ۴-۲- تصمیم‌گیرندگان – با استفاده از روش تصمیم‌گیری کارفرما (M) و همینطور تصمیم‌گیرندگان حوزه‌ی توسعه‌دهنده با استفاده از روش تصمیم‌گیری توسعه‌دهنده (M') چارچوب نهایی را انتخاب می‌نمایند.

سیمپسون[۱] در [۴۳] به صراحت اعلام می‌نماید که "تفاوت‌های واضحی بین روش‌های مختلف تصمیم‌گیری وجود دارد ولی نمی‌توان به وضوح برتری خاصی میان روش‌ها قائل شد". همچنین گیتونی[۲] و همکاران در [۴۴] چارچوبی را برای انتخاب روش تصمیم‌گیری معرفی کرده‌اند که با طبقه‌بندی نوع ورودی‌های ماتریس تصمیم‌گیری و نوع خروجی مورد نظر در فرآیند تصمیم‌گیری، روش‌های مطلوب تصمیم‌گیری شناسایی می‌شوند. این چارچوب که در قالب یک جدول ارائه شده است، دارای خانه‌های خالی نیز می‌باشد. این خانه‌ها نشان‌دهنده‌ی عدم تفاوت چشمگیر بین روش‌های مختلف تصمیم‌گیری برای آن حالت (خانه) خاص می‌باشد. با بررسی این چارچوب مشخص شد که مسئله تصمیم‌گیری حاضر در یکی از خانه‌های خالی این چارچوب قرار می‌گیرد. بنابراین با توجه به تحلیل زیر روش‌های تصمیم‌گیری M و M' را انتخاب می‌نماییم.

جدول صفحه‌ی بعد نقاط ضعف و قوت و کاربرد کلی روش‌های تصمیم‌گیری را نمایش می‌دهد.

---

۱ Simpson

۲ Guitouni

جدول ۴-۴: مزایا و معایب روش‌های تصمیم‌گیری

روش تصمیم‌گیری	توضیح
"تحلیل هزینه، فایده" و "مزایا، معایب"	این دو روش متکی بر قدرت تحلیل و شهود شخصی بوده و عمدتاً در مسائلی مانند سیاست‌گذاری‌های کلان اجتماعی مورد استفاده قرار می‌گیرند.
"بیشینه‌ی کمینه‌ها و بیشینه‌ی بیشینه‌ها" و "رتبه‌نگاری"	این دو روش متکی بر سنجش و مقایسه یک معیار در هر لحظه بوده و روش سخت‌گیرانه‌ای برای رسیدن سریع به بهترین حالت یا اجتناب از بدترین حالت دارند.
"عطفی و فصلی"	این دو روش امکان اعمال تصمیم‌گیری دو یا چند مرحله‌ای را فراهم می‌آورند که برای تأمین حداقل خواسته‌ها در شرایط خاص مناسب است و برای مسئله‌ی حاضر مناسب می‌باشد.
روش سنجش چند متغیره ساده	مزیت این روش در سهولت آن می‌باشد. این روش تأثیر متقابل معیارها را در نظر نمی‌گیرد.
روش میانه‌های عمومی	از این روش بیشتر در مسائل بهینه‌سازی استفاده می‌شود. این روش تأثیر متقابل معیارها را در نظر نمی‌گیرد.
فرآیند سلسله مراتبی تحلیلی	تحلیلی: این روش امکان مقایسه‌های تک به تک راه‌حل‌ها یا معیارها را فراهم می‌آورد که برای مسائلی با تعداد معیارهای زیاد مناسب است (مانند مسئله‌ی حاضر). همچنین دیگر مزیت این روش ماهیت فازی آن است که برای مشارکت افراد مناسب است.
ELECTRE	این روش این امکان را فراهم می‌آورد تا تصمیم‌گیرندگان زیرمجموعه‌ای از راه‌حل‌ها را که از سایر مجموعه برتر هستند شناسایی نمایند به طوریکه این اندازه این زیر مجموعه قابل تغییر می‌باشد؛ با این وجود روش دقیقی برای تعیین اندازه‌ی این زیرمجموعه وجود ندارد. مشکل دیگر استفاده از این روش، محاسبه‌ی نمایه‌ی ناهمخوانی برای مسئله حاضر می‌باشد که همواره برابر صفر شده و امکان رتبه‌بندی راه‌حل‌های مختلف را از بین می‌برد.
PROMETHEE	این روش علاوه بر سادگی محاسبات، امکان مقایسه تک به تک معیارهای مختلف هر دو راه‌حل مختلف را فراهم می‌آورد که از این نظر برای مسائلی مانند مسئله‌ی حاضر که تعداد معیارها در آن زیاد است، مناسب می‌باشد.
TOPSIS	مزیت این روش در کاربرد گسترده‌ی آن در انواع صنایع و بی‌مقیاس‌سازی اعداد پیش از انجام محاسبات می‌باشد. در عین حال با توجه به محاسبه‌ی v_j^+ و v_j^- در این روش (رجوع شود به TOPSIS ۰) و همچنین ماهیت صفر یا یک کارایی راه‌حل‌ها در مسئله‌ی حاضر، عملاً مزیت استفاده از این روش خنثی می‌شود.

برای اولویت‌بندی و انتخاب روش‌های تصمیم‌گیری مناسب برای مسئله‌ی حاضر از روش نظرسنجی خبرگان استفاده شده است. در این روش یک گروه ده نفره از متخصصین علوم ریاضی کاربردی و فناوری اطلاعات کمک گرفته شده است.

در ابتدا طی دو جلسه یک ساعته روش‌های تصمیم‌گیری و جزئیات مربوط به آن‌ها برای مشارکت‌کنندگان تشریح گردید. سپس مشارکت‌کنندگان روش‌های مختلف تصمیم‌گیری را با توجه به معیارهای تعریف شده (در ادامه تشریح می‌شود) روش‌های مختلف تصمیم‌گیری با یکدیگر مقایسه نمودند. برای انجام این نظرسنجی از روش پرسشنامه استفاده شده است؛ به نحوی که هر یک از افراد مشارکت‌کننده در نظرسنجی پرسشنامه‌ای را مشابه با (جدول ۴-۶) جدول بعدی (جدول ۴-۶: مقایسه‌ی روش‌های تصمیم‌گیری) مجموعه‌ای از معیارها برای مقایسه روش‌های مختلف تصمیم‌گیری و همچنین نتایج حاصل از انجام نظرسنجی را نشان می‌دهد. این معیارها با توجه به نیازهای مسئله‌ی حاضر (ارائه یک مدل تصمیم‌گیری جهت انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی) انتخاب شده‌اند.تکمیل نموده‌اند و روش‌های تصمیم‌گیری را مقایسه نموده‌اند. در انتها برای جمع‌بندی نظرات، میانگین امتیاز داده شده به هر روش به ازای هر معیار محاسبه گردید.

جزئیات این نظرسنجی در زیر آمده است:

جدول ۴-۵: مشارکت‌کنندگان در نظرسنجی انتخاب روش تصمیم‌گیری

رشته	مشارکت‌کنندگان (نفر)	سطح تحصیلات	بازه تصمیم‌گیری
ریاضی کاربردی	۵	کارشناسی ارشد	کم – متوسط - زیاد
فناوری اطلاعات	۵	کارشناسی ارشد	

جدول بعدی (جدول ۴-۶: مقایسه‌ی روش‌های تصمیم‌گیری) مجموعه‌ای از معیارها برای مقایسه روش‌های مختلف تصمیم‌گیری و همچنین نتایج حاصل از انجام نظرسنجی را نشان می‌دهد. این معیارها با توجه به نیازهای مسئله‌ی حاضر (ارائه یک مدل تصمیم‌گیری جهت انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی) انتخاب شده‌اند.

- ساده و شهودی: این معیار میزان سهولت محاسبات و امکان تجسم ذهنی از آن را نشان می‌دهد.
- امکان ارائه‌ی رتبه‌بندی: یکی از اهداف این پژوهش، ارائه مدلی است که چارچوب‌ها را با توجه به معیارهای مختلف نسبت به هم رتبه‌بندی نماید.

- چند تصمیم‌گیرنده: توسعه‌ی سامانه‌های سازمانی معمولاً توسط چند تیم و با حضور نمایندگانی از مالک نهایی آن صورت می‌گیرد. همچنین این سامانه‌ها دارای ذینفعان متعددی می‌باشند.
- پشتیبانی از افرد غیر فنی: ممکن است برخی از ذینفعان یک سامانه‌ی سازمانی با نگاه غیر فنی (مدیریتی و ...) به پروژه توسعه‌ی یک سامانه بنگرند. بنابراین لازم است تا روش تصمیم‌گیری امکان پشتیبانی از این افراد را نیز داشته باشد.
- عینی به جای ذهنی: این معیار نشان‌دهنده‌ی دقیق و قطعی بودن روش تصمیم‌گیری است. روش‌هایی که محاسباتشان با اتکا بر تحلیل‌های شخصی انجام می‌گیرد معمولاً ذهنی به شمار می‌روند.
- پشتیبانی از معیارهای متعدد: با توجه به مطالب فصل دوم، انتظار می‌رود معیارهای متعددی برای مقایسه‌های چارچوب‌های مختلف مورد استفاده قرار گیرند. در برخی از روش‌های تصمیم‌گری با توجه به نوع محاسباتشان از تعداد اندکی معیار جهت مقایسه می‌توان استفاده نمود.
- جامع بودن مقایسه‌ها: این معیار نشان‌دهنده‌ی استفاده از تمام معیارها و روابطشان در فرآیند محاسبات می‌باشد.
- استفاده جهت وزن‌دهی: این معیار توانایی روش را برای استفاده به عنوان ابزاری جهت تعیین وزن معیارها در مسئله‌ای دیگر نشان می‌دهد.
- مقایسه‌ی تک به تک: این معیار امکان مقایسه تک به تک راه‌حل‌ها را توسط روش نشان می‌دهد. این ویژگی برای نمایش میزان برتری هر راه‌حل نسبت به سایر راه‌حل‌ها بکار می‌رود.

جدول ۴-۶: نتایج مقایسه‌ی روش‌های تصمیم‌گیری توسط خبرگان

روش	ساده و شهودی	امکان ارائه رتبه‌بندی	چند تصمیم گیرنده	پشتیبانی از افراد غیر فنی	عینی به جای ذهنی	پشتیبانی معیارهای متعدد	جامع بودن مقایسه‌ها	استفاده جهت وزن‌دهی	مقایسه‌ها ی تک به تک
تحلیل هزینه فایده	***	*	*	***	***	*	***	*	*
مزایا، معایب	***	*	*	***	***	*	**	*	**
بیشینه کمینه‌ها و ...	***	***	*	***	**	***	*	*	***
عطفی و فصلی	***	**	**	***	***	***	***	*	**
رتبه‌نگاری	***	***	*	***	**	***	*	*	**
روش سنجش چند متغیره‌ی ساده	***	***	***	**	***	***	**	*	*
میانه‌های عمومی	***	***	***	**	***	***	***	*	*
فرآیند سلسله مراتبی تحلیلی	***	***	***	***	**	**	***	***	***
ELECTRE	*	***	***	*	*	***	***	*	**
PROMETHEE	***	***	***	*	**	***	***	**	***
TOPSIS	**	***	***	*	*	***	***	*	**

همانطور که در جدول بالا نشان داده شد، روش‌های تصمیم‌گیری فرآیند سلسله مراتبی تحلیلی، PROMETHEE، عطفی و فصلی، روش سنجش چند متغیره‌ی ساده و میانه‌های عمومی به ترتیب بیشترین ارزش را با توجه به معیارهای انتخاب شده کسب نموده‌اند. از آن‌جا که ماهیت فرآیند تصمیم‌گیری معرفی شده (بخش قبل) به صورت دو مرحله‌ای – حذفی می‌باشد، می‌توان از روش عطفی برای پیاده‌سازی آن استفاده نمود؛ بدین ترتیب که برای هر مرحله‌ی آن از یک روش تصمیم‌گیری مناسب بهره برد:

- مرحله‌ی اول: این مرحله که توسط کارفرما (کارکنان رده‌ی میانی و بالاتر) پیگیری می‌شود نیازمند روشی نسبتاً شهودی و با قابلیت پشتیبانی از نظرات مدیریتی (غیر فنی) می‌باشد. در این مرحله تعیین وزن معیارها با توجه به عدم آشنایی عمیق تصمیم‌گیرندگان نسبت به چارچوب‌ها، می‌تواند به کمک روش‌های تحلیل و یا فازی انجام پذیرد.

- مرحله‌ی دوم: این مرحله که توسط پیمان‌کار (کارکنان رده عملیاتی) پیگیری می‌شود، نیازمند روشی عینی با پشتیبانی از معیارهای متعدد و همچنین بررسی و برتری‌یابی چارچوب‌ها نسبت به یکدیگر (مقایسه تک به تک) می‌باشد.

با توجه به مطالب بالا، از روش "فرآیند سلسله مراتبی تحلیلی" که امکان مقایسه‌های تک به تک معیارها را به شکلی فازی فراهم می‌کند، برای تعیین وزن معیارهای حوزه‌ی کارفرما و از "PROMETHEE" به عنوان روش تصمیم‌گیری کارفرما (M) و از روش "PROMETHEE" به تنهایی به عنوان روش تصمیم‌گیری توسعه‌دهنده (M') استفاده می‌شود. جدول زیر جزئیات مدل تصمیم‌گیری مورد استفاده در این مسئله را نشان می‌دهد:

جدول ۴-۷: جزئیات مدل تصمیم‌گیری

شماره‌ی گام	روش تصمیم‌گیری	تصمیم‌گیرندگان در خصوص وزن هر معیار	معیارها
۱	فرآیند سلسله مراتبی تحلیلی و PROMETHEE	CIO	AG, MO6
		CTO	AG, RI2, RI3, RI5, MO2, MO3, MO6, SS5, AP2, AP3, AP5, AP7, AP8
		CSO	AG, MO3, AP1, AP7, AP9
۲	PROMETHEE	SA	RI1, RI2, RI3, RI4, RI5, MO1, MO2, MO4, MO5, CA1, CA2, CA3, CA4, SS1, SS2, SS3, SS4, SS5, SS7, AP1, AP2, AP3, AP4, AP5, AP6, AP8, AP9
		PM	RI2, RI3, RI5, CA3, SS5

برای تعیین وزن هر معیار در گام دوم از روش متداول مقیاس دو قطبی فاصله‌ای[۱] استفاده می‌شود [۴۵]. بر این اساس معمار نرم‌افزار و مدیر پروژه، اهمیت معیارهای مربوط به خود را در بازه‌ی صفر تا ده تعیین می‌نمایند. وزن نهایی هر معیار نیز از میانگین نظرات تصمیم‌گیرندگان استخراج خواهد شد.

[۱] Interval bipolar – scale

شکل ۴-۲: بازه‌ی اهمیت معیارها

۴-۷- مطالعه‌ی موردی آزمایشگاهی ۱

❖ توضیح: تمام اعدادی که به عنوان نتایج ارزیابی تصمیم‌گیرندگان در خصوص وزن هر معیار بکار رفته، به کمک یک ابزار نرم‌افزاری و تصادفی تولید شده است.

مسئله: توسعه‌ی یک سامانه‌ی سازمانی خودبهینه‌ساز

- در گام نخست CIO، CTO و CSO باید در خصوص وزن معیارهای مختلف نظردهی کنند. به کمک نظردهی این افراد وزن معیارها مشخص شده و سپس امکان ارزیابی چارچوب‌ها فراهم می‌گردد.

جداول صفحه بعد ارزیابی تصمیم‌گیرندگان حوزه‌ی کارفرما را در خصوص وزن هر معیار به روش "فرآیند سلسله مراتبی تحلیلی" نشان می‌دهد:

CIO:

معیار	AG	MO6
AG	۱	۱/۹۱
MO6	۰/۵۲	۱

CTO:

معیار	AG	RI2	RI3	RI5	MO2	MO3	MO6	SS5	AP2	AP3	AP5	AP7	AP8
AG	۱	۰/۸۵	۱/۳۱	۱/۷۱	۰/۰۰۵	۰/۱۶	۱/۴۲	۰/۳۷	۰/۶۲	۰/۸	۰/۶۹	۱/۵۲	۱/۶۲
RI2	۱/۱۶	۱	۰/۵۱	۰/۳۹	۰/۷۹	۰/۸۸	۰/۳۸	۱/۷۵	۱/۰۷	۱/۹۷	۱/۱۸	۰/۰۸	۱/۲۷
RI3	۰/۷۶	۱/۹۲	۱	۰/۱۸	۰/۰۴	۰/۴۵	۰/۴۴	۱/۶۲	۰/۱۶	۰/۶	۰/۱۷	۱/۵۱	۰/۴۴
RI5	۰/۵۸	۲/۵۴	۵/۴۴	۱	۱/۵۱	۰/۴۴	۱/۷	۱/۴۲	۱/۱۲	۰/۳۸	۰/۵۷	۱/۰۶	۱/۰۷
MO2	۱۷۴/۸۴	۱/۲۶	۲۳/۲۲	۰/۶۶	۱	۰/۸۵	۱/۱	۱/۷۸	۱/۱۴	۰/۸۴	۰/۵۵	۱/۹۳	۰/۲۲
MO3	۵/۹۶	۱/۱۲	۲/۱۸	۲/۲۴	۱/۱۷	۱	۱/۷۳	۰/۸۷	۱/۹۶	۱/۶۱	۰/۸۳	۰/۷۷	۰/۰۳
MO6	۰/۷	۲/۵۸	۲/۲۳	۰/۵۸	۰/۹	۱/۳۶	۱	۱/۵۳	۱/۹۶	۱/۷۱	۰/۹۷	۰/۷۵	۰/۴۶
SS5	۲/۶۷	۰/۵۶	۰/۶۱	۰/۷	۰/۵۵	۱/۱۴	۰/۶۵	۱	۰/۲	۰/۵۱	۰/۰۱	۱/۱۷	۱/۰۰۴
AP2	۱/۵۸	۰/۹۲	۶/۲۱	۰/۸۸	۰/۸۷	۰/۵	۰/۵	۴/۹۴	۱	۱/۹۴	۱/۰۸	۱/۴۶	۱/۲۶
AP3	۱/۲۴	۰/۵	۱/۶۵	۲/۶	۱/۱۸	۱/۶۲	۰/۵۸	۱/۹۳	۰/۵۱	۱	۱/۳۷	۰/۵	۰/۹۱
AP5	۱/۴۴	۰/۸۴	۵/۶	۱/۷۳	۱/۸۱	۱/۲	۱/۰۲	۶۶/۷۷	۰/۹۲	۰/۷۲	۱	۰/۱۱	۱/۲۳
AP7	۰/۶۵	۱۱/۳۶	۰/۶۶	۰/۹۳	۰/۵۱	۱/۲۹	۱/۳۳	۰/۸۵	۰/۶۸	۰/۹۷	۸/۴۸	۱	۰/۱۶
AP8	۰/۶۱	۰/۷۸	۲/۲۲	۰/۹۳	۴/۴۲	۲۶/۵۵	۲/۱۳	۰/۹۹	۰/۷۹	۱/۰۹	۰۱/۸	۵/۹۶	۱

CSO:

معیار	AG	MO3	AP1	AP7	AP9
AG	۱	۰/۶۴	۰/۴۶	۱/۴۱	۰/۶۵
MO3	۱/۵۴	۱	۰/۶۲	۱/۶۷	۰/۹۷
AP1	۲/۱۵	۱/۵۸	۱	۱/۱۱	۰/۹
AP7	۰/۷	۰/۵۹	۰/۹	۱	۱/۴۶
AP9	۱/۵۳	۱/۰۲	۱/۱	۰/۶۸	۱

- حال می‌توان وزن هر معیار را بر اساس فرمول زیر شناسایی نمود. با استفاده از این فرمول وزن هر معیار به ازای هر جدول محاسبه می‌شود. برای محاسبه وزن نهایی معیارها، از میانگین‌گیری استفاده شده است:
  - P_i: وزن معیار
  - R_{ij}: مقادیر وارد شده در جداول بالا

$$P_i = \frac{\sum_{j=1}^{n} R_{ij}}{\sum_{i=1}^{n}\sum_{j=1}^{n} R_{ij}}$$

فرمول ۴-۱ : محاسبه اولویت‌ها در فرآیند سلسله مراتبی تحلیلی [۴۲]

جدول زیر وزن نهایی هر معیار را با توجه به ارزیابی انجام شده توسط تصمیم‌گیرندگان نشان می‌دهد:

معیار	AG	RI2	RI3	RI5	MO2	MO3	MO6	SS5	AP1	AP2	AP3	AP5	AP7	AP8	AP9
وزن	۰/۳۷	۰/۰۴	۰/۱	۰/۰۲	۰/۰۲	۰/۰۷	۰/۰۲	۰/۱۶	۰/۰۰۶	۰/۰۲	/۰۲	۰/۰۳	۰/۰۳	۰/۰۳	۰/۰۰۷

- تا این مرحله وزن معیارها در گام نخست شناسایی شده است. حال با استفاده از روش PROMETHEE چارچوب‌های مختلف با یکدیگر مقایسه می‌شوند. فرمول‌های این روش در زیر آمده است:

$$P_i(A_j, A_k) = 0, \quad if \quad a_{ij} - a_{ik} < 0$$
$$otherwise:$$
$$P_i(A_j, A_k) = a_{ij} - a_{ik}$$

فرمول ۴-۲: محاسبه‌ی تابع ترجیح [۴۲]

برای محاسبه‌ی ترجیح کلی یک راه‌حل بر دیگری (در جمیع معیارها) از نمایه‌ی ترجیح (π) استفاده می‌شود:

$$0 < \Pi(A_j, A_k) = \sum_{i=1}^{m} w_i P_i(A_j, A_k) < 1$$

فرمول ۴-۳: نمایه‌ی ترجیح [۴۲]

برای رتبه‌بندی راه‌حل‌های مختلف از جمع توابع جریان رتبه‌بندی مثبت و منفی و ساده استفاده می‌شود که فرمولشان در زیر آمده است:

$$\Phi^{+}(A_j) = \frac{1}{n-1} \sum_{k=1}^{n} \Pi(A_j, A_k)$$

$$\Phi^{-}(A_j) = \frac{1}{n-1} \sum_{k=1}^{n} \Pi(A_k, A_j)$$

$$\Phi(A_j) = \Phi^{+}(A_j) - \Phi^{-}(A_j)$$

فرمول ۴-۴: توابع جریان [۴۲]

نتایج حاصل از گام نخست با اجرای روش "PROMETHEE":

نمایه‌ی ترجیح:

راه‌حل	کریشنامورتی و بابو	یانگ و همکاران	جرجس و تیلور	AOA	MOCAS	DSCB
کریشنامورتی و بابو	-	۰/۰۹	۰	۰/۱۸	۰/۱۶	۰/۱۸
یانگ و همکاران	۰/۴۵	-	۰	۰/۲۳	۰/۲۱	۰/۲۶
جرجس و تیلور	۰/۴۸	۰/۱۲	-	۰/۰۷	۰/۲۱	۰/۲۴
AOA	۰/۵۶	۰/۱۱	۰/۰۵	-	۰/۰۶	۰/۰۳
MOCAS	۰/۵۶۶	۰/۱۴۶	۰/۱۱۶	۰/۰۵۶	-	۰/۰۵۶
DSCB	۰/۵۶	۰/۱۱	۰/۰۵	۰	۰/۰۵	-

تابع جریان (اولویت‌های ۱ تا ۴ راه‌حل‌های گام دوم خواهند بود):

راه‌حل	تابع جریان مثبت	تابع جریان منفی	تابع ترجیح	اولویت
کریشنامورتی و بابو	۰/۱۲۲	۰/۵۲۳۲	-۰/۴۰۱۲	۶
یانگ و همکاران	۰/۲۳	۰/۱۱۵۲	۰/۱۱۴۸	۲
جرجس و تیلور	۰/۲۲۴	۰/۰۴۳۲	۰/۱۸۰۸	۱
AOA	۰/۱۶۲	۰/۱۰۷۲	۰/۰۵۴۸	۳
MOCAS	۰/۱۸۸۰	۰/۱۳۸	۰/۰۵	۴
DSCB	۰/۱۵۸	۰/۱۵۵۲	۰/۰۳۲	۵

- گام نخست فرآیند دو مرحله‌ای تصمیم‌گیری در اینجا به پایان می‌رسد. در گام دوم تصمیم‌گیرندگان حوزه پیمانکار وزن معیارهای مربوط به خود را با استفاده از روش مقیاس دوقطبی تعیین می‌نمایند.

جدول صفحه‌ی بعد وزن هر معیار را با توجه به نظرات تصمیم‌گیرندگان گام دوم نشان می‌دهد:

معیار	وزن انتساب داده شده SA	وزن انتساب داده شده PM	میانگین (وزن نهایی)
RI1	۴	–	۴
RI2	۴	۸	۶
RI3	۴	۶	۵
RI4	۵	–	۵
RI5	۶	۲	۴
MO1	۴	–	۴
MO2	۲	–	۲
MO4	۱	–	۱
MO5	۲	–	۲
CA1	۴	–	۴
CA2	۲	–	۲
CA3	۶	۶	۶
CA4	۸	–	۸
SS1	۸	–	۸
SS2	۹	–	۹
SS3	۴	–	۴
SS4	۷	–	۷
SS5	۸	۲	۵
SS7	۳	–	۳
AP1	۲	–	۲
AP2	۳	–	۳
AP3	۲	–	۲
AP4	۲	–	۲
AP5	۵	–	۵
AP6	۶	–	۶
AP8	۶	–	۶
AP9	۱	–	۱

- با توجه به بدست آمدن وزن هر معیار، امکان مقایسه چارچوب‌ها در گام دوم وجود دارد که در ادامه نشان داده شده است:

نتایج حاصل از گام دوم با اجرای روش "PROMETHEE":

نمایه‌ی ترجیح:

راه‌حل	یانگ و همکاران	جرجس و تیلور	AOA	MOCAS
یانگ و همکاران	-	۲۱	۲۰	۳۵
جرجس و تیلور	۸	-	۱۲	۲۹
AOA	۱۹	۲۹	-	۳۸
MOCAS	۲۷	۲۰	۲۴	-

تابع جریان:

راه‌حل	تابع جریان مثبت	تابع جریان منفی	تابع ترجیح	اولویت
یانگ و همکاران	۲۵/۳۳	۱۸	۷/۳۳	۲
جرجس و تیلور	۱۶/۳۳	۲۳/۳۳	-۷	۳
AOA	۲۸/۶۶	۱۸/۶۶	۱۰	۱
MOCAS	۲۳/۶۶	۶۴	-۱۰/۳۴	۴

با توجه به جدول بالا، برای مسئله‌ی تعریف شده و با توجه به نظر تصمیم‌گیرندگان، چارچوب AOA اولویت تیم توسعه‌دهنده خواهد بود.

۴-۸- مطالعه‌ی موردی آزمایشگاهی ۲

مسئله: توسعه‌ی یک سامانه‌ی سازمانی خودپیکربند و خودحفاظ

جداول زیر ارزیابی تصمیم‌گیرندگان حوزه‌ی کارفرما را در خصوص وزن هر معیار به روش "فرآیند سلسله مراتبی تحلیلی" نشان می‌دهد:

CIO:

معیار	AG	MO6
AG	۱	۰/۸۵
MO6	۱/۱۷	۱

CTO:

معیار	AG	RI2	RI3	RI5	MO2	MO3	MO6	SS5	AP2	AP3	AP5	AP7	AP8
AG	۱	۱/۲۷	۱/۶۹	۱/۰۸	۱/۸۱	۰/۵	۱/۱۳	۰/۹۷	۰/۷۷	۱/۸۸	۰/۷۱	۱/۳۳	۱/۹۶
RI2	۰/۷۸	۱	۱/۹۷	۱/۹۸	۱/۹۱	۱/۹۲	۱/۴۳	۰/۴۲	۱/۸۸	۰/۶۷	۰/۸۷	۱/۵۳	۰/۲
RI3	۰/۵۸	۰/۵	۱	۱/۷۷	۱/۷۱	۱/۵۱	۰/۳۷	۱/۲۷	۰/۹۵	۱/۹۹	۱/۷	۰/۵۳	۰/۰۱
RI5	۰/۹۲	۰/۵	۱/۲۹	۱	۱/۳۸	۰/۲۴	۰/۷۹	۱/۹۳	۱/۸۶	۱/۳۶	۰/۵۴	۱/۱۴	۰/۰۸
MO2	۰/۵۴	۰/۵۲	۰/۵۸	۰/۷۲	۱	۱/۷۳	۰/۹۸	۱/۵۸	۰/۹	۱/۷	۱/۵۶	۰/۶۳	۰/۲۲
MO3	۱/۹۶	۰/۵۲	۰/۶۵	۴/۱۱	۰/۵۷	۱	۰/۳۷	۰/۰۵	۱/۱۲	۱/۵۲	۱/۴۷	۱/۷۳	۱/۴۹
MO6	۰/۸۷	۰/۶۹	۲/۶۶	۱/۲۶	۱/۰۱	۲/۶۷	۱	۰/۳۱	۰/۱	۰/۹۹	۰/۷۶	۱/۶۹	۱/۸۱
SS5	۱/۰۲	۲/۳۸	۰/۷۸	۰/۵۱	۰/۶۳	۱۷/۰۹	۳/۲	۱	۰/۸۹	۱/۰۱	۰/۸	۱/۹۵	۰/۵۹
AP2	۱/۲۹	۰/۵۹	۱/۰۴	۰/۵۳	۱/۱	۰/۸۸	۹/۳۱	۱/۱۱	۱	۱/۷۴	۱/۱۲	۱/۹۸	۱/۵۷
AP3	۰/۵۲	۱/۴۷	۱/۰۰۶	۰/۷۳	۰/۵۸	۰/۶۵	۱/۰۰۸	۰/۹۸	۰/۵۷	۱	۱/۰۵	۱/۳۴	۱/۵۲
AP5	۱/۳۸	۱/۱۳	۰/۵۸	۱/۸۲	۰/۶۳	۰/۶۷	۱/۳۱	۱/۲۴	۰/۸۹	۰/۹۴	۱	۰/۷۹	۱/۸
AP7	۰/۷۴	۰/۶۴	۱/۸۷	۰/۸۷	۱/۵۷	۰/۵۷	۰/۵۸	۰/۵۱	۰/۵	۰/۷۴	۱/۲۵	۱	۱/۲۱
AP8	۰/۵	۴/۸۱	۶۵/۸۱	۱۱/۶۸	۴/۵	۰/۶۷	۰/۵۵	۱/۶۹	۰/۶۳	۰/۶۵	۰/۵۵	۰/۸۲	۱

CSO:

معیار	AG	MO3	AP1	AP7	AP9
AG	۱	۰/۴	۰/۶۱	۰/۷۷	۱/۹۸
MO3	۲/۴۸	۱	۰/۷	۱/۱۴	۱/۶۳
AP1	۱/۶۱	۱/۴۲	۱	۱/۲۸	۱/۳۳
AP7	۱/۲۹	۰/۸۷	۰/۷۸	۱	۰/۴
AP9	۰/۵	۰/۶۱	۰/۷۴	۲/۴۷	۱

جدول زیر وزن نهایی هر معیار را نشان می‌دهد:

معیار	AG	RI2	RI3	RI5	MO2	MO3	MO6	SS5	AP1	AP2	AP3	AP5	AP7	AP8	AP9
وزن	۰/۰۴	۰/۰۵	۰/۲۷	۰/۰۸	۰/۰۵	۰/۱	۰/۰۷	۰/۰۴	۰/۰۰۹	۰/۰۳	۰/۰۴	۰/۰۴	۰/۰۶	۰/۰۴	۰/۰۱

نتایج حاصل از گام نخست با اجرای روش "PROMETHEE":

نمایه‌ی ترجیح:

راه‌حل	کریشنامورتی و بابو	یانگ و همکاران	جرجس و تیلور	AOA	MOCAS	DSCB
کریشنامورتی و بابو	–	۰/۱۳	۰	۰/۱۱	۰/۱۲	۰/۱۱
یانگ و همکاران	۰/۴۸	–	۰/۴۸	۰/۱۹	۰/۲۲	۰/۰۸
جرجس و تیلور	۰/۲۳	۰/۲۳	–	۰/۲۴	۰/۱۹	۰/۱۷
AOA	۰/۴۴	۰/۱۲	۰/۰۸	–	۰/۰۸	۰/۰۴
MOCAS	۰/۶۱۹	۰/۱۸۹	۰/۲۷۹	۰/۱۳۹	–	۰/۰۹۹
DSCB	۰/۶	۰/۲۹	۰/۱۶	۰/۴۱	۰/۱۵	–

تابع جریان (اولویت‌های ۱ تا ۴ راه‌حل‌های گام دوم خواهند بود):

راه‌حل	تابع جریان مثبت	تابع جریان منفی	تابع ترجیح	اولویت
کریشنامورتی و بابو	۰/۰۹۴	۰/۴۷۳۸	-۰/۳۷۹۸	۶
یانگ و همکاران	۰/۲۰۲	۰/۱۹۱۸	۰/۰۱۰۲	۴
جرجس و تیلور	۰/۲۱۲	۰/۱۱۱۸	۰/۱۰۰۲	۳
AOA	۰/۱۵۲۲	۰/۲۱۷۸	۰/۰۶۵۶	۵
MOCAS	۰/۲۶۵	۰/۱۵۲	۰/۱۱۳	۲

راه‌حل	تابع جریان مثبت	تابع جریان منفی	تابع ترجیح	اولویت
DSCB	۰/۳۲۲	۰/۰۹۹۸	۰/۲۲۲۲	۱

جدول زیر وزن هر معیار را با توجه به نظرات تصمیم‌گیرندگان گام دوم نشان می‌دهد:

معیار	وزن انتساب داده شده SA	وزن انتساب داده شده PM	میانگین (وزن نهایی)
RI1	۲	–	۲
RI2	۳	۵	۴
RI3	۶	۶	۶
RI4	۵	–	۵
RI5	۹	۷	۸
MO1	۴	–	۴
MO2	۱	–	۱
MO4	۱	–	۱
MO5	۲	–	۲
CA1	۲	–	۲
CA2	۲	–	۲
CA3	۵	۷	۶
CA4	۷	–	۷
SS1	۸	–	۸
SS2	۷	–	۷
SS3	۴	–	۴
SS4	۲	–	۲
SS5	۱	۳	۲
SS7	۳	–	۳
AP1	۶	–	۶
AP2	۸	–	۸
AP3	۸	–	۸
AP4	۲	–	۲
AP5	۲	–	۲
AP6	۹	–	۹
AP8	۲	–	۲

معیار	وزن انتساب داده شده SA	وزن انتساب داده شده PM	میانگین (وزن نهایی)
AP9	۱	-	۱

نتایج حاصل از گام دوم با اجرای روش "PROMETHEE":

نمایه‌ی ترجیح:

راه‌حل	یانگ و همکاران	جرجس و تیلور	MOCAS	DSCB
یانگ و همکاران	-	۱۳	۳۲	۶
جرجس و تیلور	۱۳	-	۲۲	۱۲
MOCAS	۳۰	۲۷	-	۲۷
DSCB	۲۸	۳۷	۳۰	-

تابع جریان:

راه‌حل	تابع ترجیح مثبت	تابع ترجیح منفی	تابع ترجیح	اولویت
یانگ و همکاران	۱۸/۳۳	۲۴/۶۶	-۵/۳۳	۳
جرجس و تیلور	۱۵/۶۶	۲۷	-۱۱/۳۴	۴
MOCAS	۲۸	۲۸	۰	۲
DSCB	۳۱/۶۶	۶/۳۳	۲۵/۳۳	۱

با توجه به جدول بالا، برای مسئله‌ی تعریف شده و با توجه به نظر تصمیم‌گیرندگان، چارچوب DSCB اولویت تیم توسعه‌دهنده خواهد بود.

۵- فصل پنجم: جمع‌بندی

۵-۱- مقدمه

در این فصل به جمع‌بندی مطالب این پژوهش پرداخته می‌شود. در ابتدا خلاصه‌ای از مجموعه مطالب این پژوهش ارائه می‌شود و نتایج حاصل از آن در بخش نتیجه‌گیری ارائه خواهد شد. قسمت انتهایی این فصل به کارهای آتی خواهد پرداخت.

۵-۲- خلاصه‌ی پژوهش

آنچه‌که در طول این پژوهش مورد بررسی قرار گرفته است، شامل سه بند زیر می‌شود:

- نگاهی بر نحوه‌ی کارکرد، ویژگی‌ها و قابلیت‌های سامانه‌های خودتطبیق نرم‌افزاری و سامانه‌های سازمانی

در این قسمت با معرفی چرخه‌ی خودتطبیقی و مؤلفه‌های اساسی مورد استفاده در آن زمینه‌ای برای پرداختن به ادبیات موضوع سامانه‌های خودتطبیق فراهم گردید و در انتها موارد موفقی از پیاده‌سازی‌های سامانه‌های خودتطبیق معرفی شد.

همچنین در این قسمت مجموعه‌ای از مهمترین سامانه‌های سازمانی و مؤلفه‌های اصلی تشکیل‌دهنده‌ی آن‌ها نیز معرفی گردید.

- ارائه، بررسی و صفت‌یابی چارچوب‌های مورد استفاده برای توسعه‌ی سامانه‌های خودتطبیق نرم‌افزاری

در این قسمت انواع چارچوب‌ها و دسته‌بندی‌هایی که برای طبقه‌بندی این چارچوب‌ها مورد استفاده قرار می‌گیرن معرفی گردید و در ادامه شش چارچوب مطرح و متمایز مورد بحث و تحلیل قرار گرفت. در ادامه، چارچوب‌های معرفی شده به کمک یک روش صفت‌یابی معرفی شده در [۱۳] مورد ارزیابی قرار گرفتند.

- طراحی یک مدل تصمیم‌گیری برای انتخاب چارچوب توسعه‌ی سامانه‌های سازمانی خودتطبیق

قسمت پایانی این پژوهش به طراحی یک مدل تصمیم‌گیری با ویژگی‌های زیر پرداخته است. برای این منظور، دوازده روش عمده‌ی تصمیم‌گیری معرفی شده و در انتها مجموعه معیارها، تصمیم‌گیرندگان و نحوه‌ی مشارکت ایشان، فرآیند تصمیم‌گیری و روش تصمیم‌گیری مرد استفاده در مدل تصمیم‌گیری معرفی شده‌اند:

- چندمعیاره: برای ارزیابی چند بعدی و جامع
- چند تصمیم‌گیرنده: با توجه به ماهیت مسئله
- دو مرحله‌ای: با توجه به نیاز مسئله

در انتها کاربرد مدل طراحی شده به کمک دو مورد کاوی مورد بررسی قرار گرفته است.

۵-۳- نتیجه‌گیری

روند پژوهشی در حوزه‌ی سامانه‌های خودتطبیق نشان می‌دهد که گرایش به استفاده از روش‌های خارجی (استفاده از یک مؤلفه‌ی تطبیق‌دهنده در نرم‌افزار) رو به رشد است. در این راستا چارچوب‌های مختلفی برای اضافه نمودن قابلیت خودتطبیقی به سامانه‌ها در حال معرفی شدن می‌باشند که در آزمایشگاه‌های مختلفی در سراسر جهان و با اهداف مشخصی توسعه یافته‌اند.

با توجه به صفت‌یابی صورت گرفته در فصل دوم می‌توان چنین برداشت نمود که هر یک چارچوب‌های معرفی شده دارای توانایی‌های مشخصی برای برآورده‌سازی نیازهای مختلف سامانه‌های خودتطبیق می‌باشند. برخی از این چارچوب‌ها گرایش زیادی به ایجاد خواص خودبهینه‌سازی برای عنصر مدیریت شونده داشته و در مقابل برخی دیگر بر تغییرات ساختاری و تغییر پیکربندی مؤلفه مدیریت شونده در زمان اجرا متکی هستند. برخی از چارچوب‌ها امکان دسترسی به ویژگی‌های مختلفی از عنصر مدیریت شونده را فراهم می‌آورند و برخی امکان ایجاد تغییرات خاصی را ممکن می‌سازند. در ادامه نتایج صفت‌یابی صورت گرفته به طور خلاصه ارائه می‌شود.

با استفاده از مجموعه‌ی این ویژگی‌ها در فصل پنجم یک مدل تصمیم‌گیری شامل فرآیند، معیارها، تصمیم‌گیرندگان و روش تصمیم‌گیری برای انتخاب یک چارچوب با توجه به نیازمندی‌های فنی مختلف ارائه گردید. این مدل از ویژگی‌های زیر برخوردار می‌باشد:

- استفاده از روش تصمیم‌گیری چندمعیاره و چندشاخصه
- توجه به نیازهای مدیریتی کارفرما
- توجه به ویژگی‌های فنی سامانه‌های خودتطبیق
- توجه به جنبه‌های توسعه‌ی سامانه و دخالت مدیر پروژه در فرآیند تصمیم‌گیری
- قابلیت استفاده‌ی مجدد از مدل با ظهور چارچوب‌های جدید

ارزیابی این مدل نشان داد که می‌توان از آن برای رتبه‌بندی چارچوب‌های مختلف استفاده نمود. این مدل نیازهایی که برای آن در نظر گرفته شده بودند را پوشش داده و امکان پشتیبانی از تعداد بیشتری چارچوب را نیز فراهم می‌آورد. برای استفاده از چارچوب‌های جدید در مدل بایستی این چارچوب‌ها ابتدا به کمک روش صفت‌یابی معرفی شده ارزیابی شوند و سپس به عنوان راه‌حل جدید در مدل اضافه گردند.

۵-۴- مخاطبان و موارد استفاده

مخاطبان و موارد استفاده از این مدل تصمیم‌گیری شامل موارد زیر می‌شود:

۱- سازمان‌های توسعه دهنده‌ی سامانه‌های بزرگ سازمانی: معماران و طراحان ارشد این سازمان‌ها می‌توانند از پیشران‌های معماری به عنوان ورودی این مدل تصمیم‌گیری استفاده نموده و از آن در مرحله طراحی معماری استفاده نمایند.

۲- سازمان‌های بهره‌برداری کننده: از آن‌جایی‌که چارچوب‌های مورد بحث را می‌توان به سامانه‌های موروثی در حال اجرا اضافه نمود تا امکان خودکارسازی نسبی آن‌ها فراهم شود، سازمان‌های بهره‌برداری‌کننده از سامانه‌های بزرگ می‌توانند با استفاده از این مدل تصمیم‌گیری، چارچوب مناسبی را با توجه به نیازهای خود برای این منظور انتخاب نمایند.

۵-۵- کارهای آتی

با توجه به تمرکز این مدل تصمیم‌گیری به ویژگی‌های فنی سامانه‌های سازمانی، موارد زیر به عنوان حوزه‌هایی که امکان توسعه‌ی این مدل در آن‌ها وجود دارد معرفی می‌گردد:

- استفاده از ویژگی‌های کیفی نرم‌افزار به عنوان عامل تأثیرگذار بر مدل
- عمومیت بخشی به مدل جهت بهره‌برداری در سایر زمینه‌های توسعه‌ی نرم‌افزارهای خودتطبیق مانند سامانه‌های بلادرنگ و نظایر آن
- توسعه‌ی ابزارهای نرم‌افزاری مورد نیاز برای پیاده‌سازی مدل طراحی شده
- ایجاد راهکاری برای افزودن تأثیر نوع سامانه‌ی سازمانی در حال توسعه بر فرآیند، معیارها یا سایر اجزای مدل تصمیم‌گیری

پیوست ۱ : روش‌های تصمیم‌گیری چندشاخصه

روش‌های تصمیم‌گیری، زمانی استفاده می‌شوند که راه‌حل‌ها و معیارها شناسایی و دقیقاً تعریف شده باشند. شناسایی این موارد امکان تشکیل ماتریس تصمیم‌گیری را برای تصمیم‌گیرندگان فراهم می‌آرود. یک ماتریس تصمیم‌گیری به سادگی مجموعه‌ی راه‌حل‌ها، معیارها و اوزان مربوط به آن‌ها را در کنار هم نمایش می‌دهد. جدول زیر کلیت یک ماتریس تصمیم‌گیری را نمایش می‌دهد:

ماتریس تصمیم‌گیری

-	-	A_1	.	.	A_n
w_1	C_1	a_{11}	.	.	a_{1n}
.	.	.	.	.	.
.	.	.	.	.	.
w_m	C_m	a_{m1}	.	.	a_{mn}

در جدول بالا، w_iها وزن هر معیار، C_iها معیارها، A_iها راه‌حل‌ها و a_{ij}ها ارزش یا امتیاز هر راه‌حل را در قبال هر معیار توصیف می‌نمایند. از این ماتریس به عنوان ورودی روش تصمیم‌گیری مورد نظر استفاده می‌شود. در ادامه مهترین روش‌های تصمیم‌گیری معرفی شده در [۴۲] ارائه می‌شوند.

۱- روش‌های ابتدایی

منظور از روش‌های ابتدایی، روش‌هایی است که در مسائلی با تعداد راه‌حل‌های اندک، تک تصمیم‌گیرنده و با تعداد معیارهای اندک مورد استفاده قرار می‌گیرند.

تحلیل هزینه فایده

در این روش تصمیم‌گیری مهترین مسئله هزینه‌ی راه‌حل انتخاب شده و فواید ناشی از اجرای آن است. اخیراً تلاش‌هایی شده تا تأثیرات محیطی و جانبی یک تصمیم نیز در هزینه‌ها یا فایده‌ها گنجانده شوند. مهمترین مورد استفاده از این روش در حوزه سیاست می‌باشد. همچنین می‌توان آن را در ترکیب با سایر روش‌ها بکار گرفت.

مزایا، معایب[۱]

مزایا – معایب، یک روش تصمیم‌گیری تحلیلی و غیر محاسباتی به شمار می‌رود. در این روش تمام جنبه‌های مثبت و منفی راه‌حل‌های گوناگون شناسایی شده و راه‌حلی که بیشترین مزایا و کمترین معایب را داشته باشد انتخاب می‌شود.

بیشینه‌ی کمینه‌ها[۲] و بیشینه‌ی بیشینه‌ها[۳]

در روش اول، هدف دوری از بدترین نتایج ممکن می‌باشد. بدین منظور ضعیف‌ترین معیار هر راه‌حل شناسایی شده و راه‌حلی که بیشترین مقدار را در معیار برگزیده‌اش داشته باشد، برگزیده می‌شود. روش دوم راهکاری کاملاً برعکس ارائه می‌نماید و در آن بیشترین مقدار معیارهای مختلف راه‌حل‌ها شناسایی شده و راه‌حلی که بیشترین مقدار را داشته باشد برگزیده می‌شود. بدیهی است که از این روش‌ها زمانی استفاده می‌شود که معیارها – حداقل به کمک یک مقیاس مشخص – قابل مقایسه باشند.

روش‌های عطفی و فصلی

در این روش‌ها به جای تمرکز بر بهترین مقادیر معیارها، تمرکز بر رضایت‌بخش بودن آن‌هاست. در روش عطفی، راه‌حل منتخب باید به ازای هر معیار، حداقل قابل قبولی را ارائه نماید. در روش فصلی، راه‌حل منتخب باید علاوه بر محدودیت قبلی، در یک معیار فراتر از حداقل‌ها باشد. معمولاً از این روش برای انتخاب زیر مجموعه‌ای از راه‌حل‌ها به عنوان ورودی یک روش تصمیم‌گیری دیگر استفاده می‌شود.

رتبه‌نگاری[۴]

در این روش معیارهای مورد نظر به لحاظ اهمیت رتبه‌بندی شده و راه‌حلی که بیشترین مقدار را در مهمترین معیار داشته باشد انتخاب می‌شود. چنانچه دو راه‌حل مقدار برابری به ازای مهمترین معیار داشته باشند، دومین معیار پر اهمیت بررسی می‌شود.

---

۱ Pros and cons
۲ Maximin
۳ Maximax
۴ Lexticographic

۲- نظریه‌ی کارایی چند معیاره[۱]

در این دسته از روش‌ها، ماتریس تصمیم‌گیری به عنوان ورودی یک تابع مطلوبیت[۲] مورد استفاده قرار می‌گیرد. این توابع مطلوبیت مقادیر خام معیارهای هر راه‌حل را با توجه به اوزان تعریف شده به ازای هر معیار با شیوه‌ای مشخص ترکیب کرده و در نهایت عددی بین صفر تا یک و یا صفر تا صد را به عنوان کارایی راه‌حل (x_j) ارائه می‌نمایند. هر چه این عدد ببشتر باشد، کارایی راه‌حل نیز بیشتر خواهد بود.

روش سنجش چندمتغیره‌ی ساده[۳]

در این روش از تابع مطلوبیت زیر استفاده می‌شود:

$$x_j = \frac{\sum_{i=1}^{m} w_i a_{ij}}{\sum_{i=1}^{m} w_i}, j = 1..n.$$

تابع مطلوبیت سنجش چند متغیره‌ی ساده [۴۲]

میانه‌های عمومی[۴]

از این روش بیشتر در مسائل بهینه‌سازی استفاده می‌شود. تابع مطلوبیت آن عبارت است از:

---

۱ Multiattribute Utility Theory (MAUT)
۲ Utility function
۳ Simple Multiattribute Rating Technique (SMART)
۴ Generalized Means

$$w = \sum_{i=1}^{m} w_i$$

$$x_i = \prod_{i=1}^{m} a_{ij}^{w_i / w}, i = 1,\ldots, n.$$

تابع مطلوبیت روش میانه‌های عمومی [۴۲]

فرآیند سلسله مراتبی تحلیلی[۱]

ایده‌ی اصلی در این روش تبدیل ارزیابی‌های ذهنی در خصوص اولویت‌بندی مسائل به مجموعه‌ای از وزن‌ها و امتیازات می‌باشد. برای انجام این اولویت‌بندی که می‌تواند برای راه‌حل‌ها و یا تعیین وزن معیارها بکارگرفته شود، راه‌حل‌ها یا معیارها دو به دو با هم مقایسه شده و میزان اهمیت هر کدام نسبت به دیگری مشخص می‌شود. برای این منظور عددی بین یک تا نه که نشان‌دهنده‌ی ارزش یک معیار نسبت به دیگری است انتخاب می‌شود. برای مثال در مقایسه معیار c_1 با c_2 عدد یک نشان‌دهنده‌ی اهمیت برابر، عدد ۵ نشان‌دهده‌ی اهمیت بالای c_1 به c_2 و عدد نه نشان‌دهنده‌ی اختلاف بسیار زیاد در مهم بودن c_1 به c_2 می‌باشد. بدین ترتیب با انجام تمام مقایسه‌های لازم، ماتریسی مشابه جدول زیر تشکیل خواهد شد:

مقایسه‌های دو به دو در فرآیند سلسله مراتبی تحلیلی

	c_1	c_2	…	c_{n-1}	c_n
c_1	1	α	…	β	θ
c_2	$1/\alpha$	1	…	μ	π
…	…	…	1	…	…
c_{n-1}	$1/\beta$	$1/\mu$	…	1	ω
c_n	$1/\theta$	$1/\pi$	…	$1/\omega$	1

[۱] The Analytic Hierarchy Process (AHP)

پس از تشکیل این ماتریس می‌توان به اولیت‌بندی معیارها یا راه‌حل‌ها مورد مقایسه پرداخت. متدوال‌ترین روش در این مرحله استفاده از ارزش مجموعه امتیازات حاصل از مقایسه‌های دو به دو می‌باشد. در فرمول زیر P_i ضریب اولویت معیار یا راه‌حل c_i را نشان می‌دهد (R_{ij}: مقادیر حاصل از مقایسه‌ی c_i به c_j).

$$P_i = \frac{\sum_{j=1}^{n} R_{ij}}{\sum_{i=1}^{n}\sum_{j=1}^{n} R_{ij}}$$

محاسبه اولویت‌ها در فرآیند سلسله مراتبی تحلیلی [۴۲]

۳- روش‌های برتری یابی[۱]

ELECTRE[۲]

مبنای این روش بر نمایه‌های همخوانی یا ناهمخوانی تعریف شده در زیر می‌باشد. به ازای هر دو راه‌حل یا معیار c_j و c_k، نمایه‌ی همخوانی (c_{jk}) برابر است با:

$$c_{jk} = \sum_{i:a_{ij} \geq a_{ik}} w_i \qquad j,k = 1,..,n,\ j \neq k.$$

نمایه‌ی همخوانی [۴۲]

---

۱ Outranking
۲ Elimination and Choice Expressing Reality

واضح است که این نمایه عددی بین صفر و یک خواهد بود. نمایه‌ی ناهمخوانی (d_{jk}) به صورت زیر تعریف می‌شود:

$$d_{jk} = 0 \quad if \quad a_{ij} > a_{ik} \quad j,k = 1,..,n,\ j \neq k.$$

$$otherwise:$$

$$d_{jk} = \max_{i=1,..,m} \frac{a_{ik} - a_{ij}}{\max a_{ij} - \min a_{ij}} \quad j,k = 1,..,n, \quad j \neq k.$$

نمایه‌ی ناهمخوانی [۴۲]

با توجه به فرمول بالا، مقدار نمایه‌ی ناهمخوانی (d_{jk})، بین اختلاف مقدار دو راه‌حل در یک معیار مشخص و حداکثر امتیاز یک راه‌حل در همان معیار، محاسبه خواهد شد. سپس یک مقدار آستانه‌ی همخوانی (c*) و آستانه‌ی ناهمخوانی (d*) به صورت $0<d^*<c^*<1$ تعریف می‌گردد. در این مرحله چنانچه $c_{jk}>c^*$ و $d_{jk}<d^*$ باشند، معیار یا راه‌حل c_j بر c_k برتری دارد.

همواره یک مجموعه‌ی مشخص از راه‌حل‌ها و یا معیارها بر یک راه‌حل یا معیار مشخص برتری دارند که این مجموعه بسیار ارزشمند می‌باشد. با تغییر در آستانه‌ها می‌توان این مجموعه را بزرگ‌تر یا کوچک‌تر نمود.

PROMETHEE[۱]

از این روش تنها پس از تعیین اوزان معیارها و به شرطی که مجموع اوزان برابر یک باشد استفاده می‌شود. در این روش از یک تابع ترجیح ($P_i(A_j,A_k)$) استفاده می‌شود که میزان ترجیح راه‌حل A_j نسبت به A_k ، در معیار c_i به صورت زیر نشان می‌دهد:

[۱] Preference Ranking Organization Method for Enrichment Evaluation

تابع ترجیح

$0< P_i(A_j,A_k)<1$	
$P_i(A_j,A_k)=0$	عدم ترجیح
$P_i(A_j,A_k)\approx 0$	ترجیح ضعیف
$P_i(A_j,A_k)\approx 1$	ترجیح قوی
$P_i(A_j,A_k)=1$	ترجیح اکید

مقدار تابع ترجیح به صورت زیر محاسبه می‌شود:

$$P_i(A_j, A_k) = 0, \quad if \quad a_{ij} - a_{ik} < 0$$
$$otherwise:$$
$$P_i(A_j, A_k) = a_{ij} - a_{ik}$$

محاسبه‌ی تابع ترجیح [۴۲]

برای محاسبه‌ی ترجیح کلی یک راه‌حل بر دیگری (در جمیع معیارها) از نمایه‌ی ترجیح (π) استفاده می‌شود:

$$0 < \Pi(A_j, A_k) = \sum_{i=1}^{m} w_i P_i(A_j, A_k) < 1$$

نمایه‌ی ترجیح [۴۲]

برای رتبه‌بندی راه‌حل‌های مختلف از توابع جریان رتبه‌بندی مثبت و منفی و ساده استفاده می‌شود که فرمولشان در زیر آمده است:

$$\Phi^{+}(A_j) = \frac{1}{n-1}\sum_{k=1}^{n}\Pi(A_j, A_k)$$

$$\Phi^{-}(A_j) = \frac{1}{n-1}\sum_{k=1}^{n}\Pi(A_k, A_j)$$

$$\Phi(A_j) = \Phi^{+}(A_j) - \Phi^{-}(A_j)$$

توابع جریان [۴۲]

چنانچه تابع جریان ساده‌ی یک راه‌حل از دیگری بزرگتر باشد، این راه‌حل ارجح می‌باشد و اگر توابع جریان ساده‌ی دو راه‌حل یکسان باشند، هر دو راه حل دارای ارجحیت برابر می‌باشند.

TOPSIS[۱]

این روش یکی از پر کاربردترین روش‌های تصمیم‌گیری چند شاخصه می‌باشد. اساس این روش بر این مفهوم استوار است که گزینه انتخابی بایستی کمترین فاصله را با راه حل ایده آل مثبت (بهترین حالت ممکن) و بیشترین فاصله را با ایده آل منفی (بدترین حالت ممکن) داشته باشد.

در ابتدا مقادیر درون ماتریس تصمیم‌گیری نرمال (r_{ij}) می‌شوند؛ به این ترتیب که هر کدام از مقادیر بر اندازه بردار مربوط به همان شاخص تقسیم می‌شود. با این کار عملاً اعداد درون جدول تصمیم‌گیری بی‌مقیاس می‌شوند.

$$r_{ij} = \frac{a_{ij}}{\sqrt{\sum_{i=1}^{m} {a_{ij}}^2}}$$

نرمال‌سازی در **TOPSIS** [۴۲]

[۱] Technique for Order of Preference by Similarity to Ideal Solution

سپس بهترین مقدار (برای معیارهایی که مقادیر بیشترشان برای کارایی نهایی راه‌حل اثر مثبت دارند) و بدترین مقدار (برای معیارهایی که مقادیر بیشترشان برای کارایی نهایی راه‌حل اثر منفی دارند) نرمال‌شده برای معیارها (v_j^+ و v_j^-) شناسایی شده و سپس فاصله اقلیدسی هر راه‌حل از ایده آل مثبت و منفی (d_i) شناسایی می‌شود. در انتها نزدیکی نسبی (CL_i) میزان کارایی هر راه‌حل را نشان می‌دهد:

$$d_i^+ = \sqrt{\sum_{j=1}^{n} (r_{ij} - v_j^+)^2}, \quad i = 1,.., n.$$

$$d_i^- = \sqrt{\sum_{j=1}^{n} (r_{ij} - v_j^-)^2}, \quad i = 1,.., n.$$

$$CL_i = \frac{d_i^-}{d_i^- + d_i^+}$$

نزدیکی نسبی [۴۲]

مراجع

[1] Q. Yang, J. Lü, J. Li, X. Ma, W. Song and Y. Zou, "Toward a fuzzy control-based approach to design of self-adaptive Software," In *Proceedings of the Second Asia-Pacific Symposium on Internet Ware*, pp.1-4, 2010.

[2] M. C. Huebscher and J. A. Mccann, "A survey of autonomic computing—degrees, models and applications," In *ACM Computing Surveys*, vol. 40, no. 3, pp.1-28, 2008.

[3] Q. Wang, "Towards a rule model for self-adaptive software," In *ACM SIGSOFT Software Engineering Notes*, vol. 30, no. 1, pp.8-12, 2005.

[4] P. Oreizy and et al, "An architecture-based approach to self-adaptive software," In *IEEE Intelligent Systems*, vol. 14, Issue 3, pp. 23–34, 1999.

[5] V.Krishnamurthy and C. Babu, "Effective self adaptation by integrating adaptive framework with architectural patterns," In *Proceedings of the 1st Amrita ACM-W Celebration on Women in Computing in India*, pp.67-70, 2010.

[6] E. Truyen and W. Joosen, "Towards an aspect-oriented architecture for self-adaptive frameworks," In *Proceedings of the 2008 AOSD Workshop on Aspects, Components, and Patterns for Infrastructure software (ACP4IS '08)*, pp.1-8, 2008.

[7] L. Murray, D. Carrington and P. Strooper, "An approach to specify software frameworks," In *Proceeding of the 27th Australasian Computer Science Conference*, pp.185-192, 2004.

[8] M. R. Nami and K. Bertels, "A survey of autonomic computing systems," In *Proceedings of the 3rd IEEE International Conference on Autonomic and Autonomous Systems*, p. 26-38, 2007.

[9] A. Khalid, M. A. Haye, M. J. Khan and S. Shamail, “Survey of frameworks, architectures and techniques in autonomic computing,” In *Proceedings of the 2009 Fifth International Conference on Autonomic and Autonomous Systems (ICAS '09),* pp.220-225, 2009.

[10] P. V. Roy, “Self management and the future of software design,” In *Electron. Notes Theor*, vol. 182, pp.201-217, 2007.

[11] C. Ballagny, N. Hameurlain and F. Barbier, “MOCAS: a state-based component model for self-adaptation,” In *Proceedings of the Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO ’09),* pp.206-215, 2009.

[12] F. Baude, L. Henrio and P. Naoumenko, “A component platform for experimenting with autonomic composition,” In *Proceedings of the 1st International Conference on Autonomic Computing and Communication Systems (Autonomics '07),* pp.1-9, 2007.

[13] N. M. Villegas, G. Tamura and R. Casallas, “A framework for evaluating quality-driven self-adaptive software systems,” In *Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS '11),* pp.80-89, 2011.

[14] J. O. Kephart and W. E. Walsh, “An artificial intelligence perspective on autonomic computing policies,” In *Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY '04),* pp.3-12, 2004.

[15] A. Mukhija and M. Glinz, “Runtime adaptation of applications through dynamic recomposition of components,” In *Proceedings of the 18th International Conference on Architecture of Computing Systems (ARCS 2005),* pp. 124–138, 2005.

[16] S. Parekh, N. Gandhi, J. Hellerstein, D. Tilbury, T. Jayram, and J. Bigus "Using control theory to achieve service level objectives in performance management," In *Real-Time Systems*, vol. 23, pp.127–141, 2002.

[17] J. Zhang and B. H. C. Cheng, "Model-based development of dynamically adaptive software," In *Proceedings of the 28th International Conference on Software Engineering (ICSE '06)*, pp.371–380, 2006.

[18] Henry S. Kenyon, "Battlefield cognizance: tool points to future," In *Signal Magazine*, vol. 55, no. 10, pp. 21-23, 2001.

[19] N. Muscettola, P. P. Nayak, B. Pell and B. C. Williams. "Remote agent: to boldly go where no AI system has gone before," In *Artif. Intell*, vol. 103, pp. 5-47, 1998.

[20] M. Fowler, *Enterprise application integration*, 1st ed., Addison-Wesley, 2002.

[21] O. Babaoglu, G. Canright, A. Deutsch, G. A. D. Caro, F. Ducatelle, L. M. Gambardella and et al, "Design patterns from biology for distributed computing," In *ACM Trans. Auton. Adapt. Syst*, vol. 1, no. 1, pp. 26–66, 2006.

[22] K. Appleby, S. Fakhouri, L. Fong, G. Goldszmidt, M. Kalantar, D. Pazel and et al "Oceano - SLA based management of a computing utility," In *Proceedings of the 7th IFIP/IEEE International Symposium on Integrated Network Management*,pp. 855–868, 2001.

[23] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon and J. Kubiatowicz "Maintenance-free global data storage," In *IEEE Internet Computing*, vol. 5, no. 5, pp.40–49, 2001.

[24] G. M. Lohman and S. S. Lightstone, "Smart: making db2 (more) autonomic," In *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB '02),* pp. 877–879, 2002.

[25] S. Chaudhuri and V. Narasayya, "Self-tuning database systems: a decade of progress," In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB '07),* pp. 3–14, 2007.

[26] D. M. Chess, A. Segal and I. Whalley, "Unity: Experiences with a Prototype Autonomic Computing System," In *Proceedings of the First International Conference on Autonomic Computing (ICAC '04),* pp.140-147, 2004.

[27] X. Dong, S. Hariri, L. Xue, H. Chen, M. Zhang, S. Pavuluri and S. Rao "Autonomia: an autonomic computing environment," In *Proceedings of the Performance, Computing, and Communications Conference*, pp. 61–68, 2003.

[28] H. Liu and M. Parashar "A component based programming framework for autonomic applications," In *Proceedings of the International Conference on Automatic Computing*, pp. 10–17, 2004.

[29] X. Chen and M. Simons "A component framework for dynamic reconfiguration of distributed systems," In *Proceedings of the IFIP/ACM Working Conference on Component Deployment (TeX90 CD '02),* pp. 82–96, 2002.

[30] A. Ranganathan and R. H. Campbell "Autonomic pervasive computing based on planning," In *Proceedings of the International Conference on Autonomic Computing*, pp. 80 – 87, 2004.

[31] S. Abdelwahed, N. Kandasamy and S. Neema "Online control for self-management in computing systems," In *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium,* pp. 368–375, 2004.

[32] S. A. Gurguis and A. Zeid "Towards autonomic web services: achieving self-healing using web services," In *SIGSOFT Softw. Eng. Notes,* vol. 30, no. 4, pp.1–5, 2005.

[33] G. Kaiser, P. Gross, G. Kc, J. Parekh and G. Valetto "An approach to autonomizing legacy systems," In *Workshop on Self-Healing, Adaptive and Self-Managed Systems,* pp. 185-195, 2002.

[34] C. Anglano and S. Montani "Achieving self-healing in autonomic software systems: a case-based reasoning approach," In *Proceedings of the International Conference on Intelligent Computing,* pp.1–19, 2007.

[35] Z. Stanfel, Z. Hocenski and G. Martinovic "A self manageable rule driven enterprise application," In *Proceedings of the 29th International Conference on Information Technology Interfaces,* pp. 717–722, 2007.

[36] J. C. Georgas and R. N. Taylor, "Towards a knowledge-based approach to architectural adaptation management," In *Proceedings of the 1st ACM SIGSOFT Workshop on Self-Managed Systems, pp.59-63, 2004.*

[37] D. Garlan, S. W. Cheng, A. C. Huang, B. R. Schmerl and P. Steenkiste, "Rainbow: architecture-based self-adaptation with reusable infrastructure," In *IEEE Computer*, vol. 37, no. 10, pp.46-54, 2004.

[38] C. Ballagny, N. Hameurlain, and F. Barbier, "Endowing software components with autonomic capabilities based on modeling language executability," In *Proceedings of the 1st Workshop on Model-driven Software Adaptation (MADAPT07),* pp. 55–60, 2007.

[39] A. Carzaniga, D. S. Rosenblum and A. L. Wolf, "Design and evaluation of a wide-area event notification service," In *ACM Transactions on Computer Systems,* vol. 19, no. 3, pp.332–383, 2001.

[40] L. Baresi, S. Guinea and G. Tamburrelli, "Towards decentralized self-adaptive component-based systems," In *Proceedings of the 2008 International Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS '08),* pp.57-64, 2008.

[41] C. Hofmeister, “Dynamic reconfiguration of distributed applications,” Ph.D. *Dissertation, University of Maryland*, 1993.

[42] J. Fülöp, "Introduction to decision making methods," In *BDEI-3 Workshop*, Washington. 2005.

[43] L. Simpson, “Do decision makers know what they prefer? MAVT and ELECTRE II,” In *Journal of the Operational Research Society*, vol. (47), pp. 919-929, 1996.

[44] A. Guitouni, J. Martel and P. Vincke, “A framework to choose a discrete multicriterion aggregation procedure,” Defense Research Establishment Valcatier (DREV), 1998.

[۴۵] مرادی، الف. و اختر کاوان، م. "روش‌شناسی مدل‌های تحلیل تصمیم‌گیری چند معیاره،" *آرمانشهر*، سال ۲، شماره ۲، صفحه ۱۱۳-۱۲۵، ۱۳۸۸.

[۴۶] فتحیان، م. و محمدی، ع. "شناسایی و انتخاب چارچوب مناسب معماری سازمانی برای یک سازمان بیمه‌گر مجازی،" *مدیریت فناوری اطلاعات*، دوره ۵، شماره ۱، صفحه ۱۲۷-۱۴۶، ۱۳۹۲.

[۴۷] محرابیون محمدی، م. ، جهانگیر فیض آبادی، م. ، مدیری‌زاده، م. "ارائه چارچوبی جهت انتخاب چارچوب معماری سازمانی سیستم های فرماندهی و کنترل بر اساس مدل زنجیره ارزش پورتر و کارت امتیازی متوازن" *ششمین کنفرانس ملی انجمن علمی فرماهندهی و کنترل (C4I) ایران*، دانشگاه شهید بهشتی، صفحه ۵۱-۵۸، ۱۳۹۱.

Abstract

Today, the the use of the IT in pulic and private sector has propagated in every dimension of planning, decision making and operation. Considering that and the background of enterprise applications, we believe using enterprise applications hase been of the most successful usage of software for delivering the organization goals. Such softwares are usually developed in large scale, composed of various components, carring out multiple functions. It is also very common that the drivers for developing these systems change over time. Their lifetime can be very long with respect to their strategic value to the businesses and they may get linked or become integrated with other systems. These points cause the needs for changing and adapting them to new situations.

One of the most promising solutions to this problem is applicating the self-adaptive softwares. A self-adaptive software can change its functions and behavior in run time, conforming its goals that where defined by users, so they can deliver these goals with less expenses and effort [1]. Up to now, there are several frameworks introduced to develop self-adaptive systems. Any of them hase different features, pros and con with respect to different classifications but none them has a clear help for exploition. We discus these questions here:

1- Which features and advantages does any of these frameworks have?
2- How can developers adopt a framework?

In this research, we introduce a decision making model for selecting the development framkwork of self-adaptive enterprise softwares. Designing this model needs appointment of the characteristics and features of these frameworks and also it needs introducing a decision making method for evaluating the framework.

Keywords – Self-adaptive systems, Self-adaptation, Framework, Software framework, Decision making.

University of Shahid Beheshti
Faculty of electrical and computer engineering

Introducing a Decision-making Model for Selecting the Development Framework of Self-adaptive Enterprise Systems

Thesis report of master's degree of information technology
Enterprise architecture

Hamid Reza Aghiri

Supervisor:
Dr. Eslam Nazemi

2013