مروری بر سیستم‌های خودتطبیق و رویکردهای تطبیق در سیستم‌های نرم‌افزاری

آذین بزم‌آرا [۱]، سید مهران شرفی [۲]

[۱] دانشجوی کارشناسی ارشد دانشکده مهندسی کامپیوتر-دانشگاه آزاد اسلامی-واحد نجف‌آباد؛a.bazmara@yahoo.com
[۲] استادیار دانشکده مهندسی کامپیوتر-دانشگاه آزاد اسلامی-واحد نجف‌آباد؛mehran_sharafi@iaun.ac.ir

چکیده

سیستم‌های نرم‌افزاری غالبا در محیط‌های متغیر هستند و به نظارت انسان نیاز دارند تا به عملیات خود در تمامی شرایط ادامه دهند. این عملیات شامل تنظیم مجدد، رفع‌اشکال و خطایابی و به طور عمومی نگهداری هستند که منجر به صرف هزینه و زمان در فاز اجرای برنامه می‌شود. این مشکل در نتیجه ساختار حلقه‌ی باز در توسعه نرم‌افزار می‌باشد. بنابراین تقاضای زیادی برای مدیریت خودکار و به دست آوردن تمام نیازمندی‌های مورد انتظار با حداقل هزینه وزمان وجود دارد. سیستم‌های خودتطبیق پاسخ به این تقاضا هستند. این سیستم‌ها یک حلقه بسته با یک چرخه باز خورد برای تغییر در عملیاتشان هستند. این تغییرات می‌تواند از درون خود یا از محیط نشأت گرفته باشد. این سیستم‌ها باید خودشان و محیط را مانیتور کنند، تغییرات عمده را ردیابی کرده و تصمیم بگیرند چگونه واکنش نشان دهند و سرانجام تغییر را اجرا کنند. این فرآیندها به ویژگی‌های تطبیق ذینفعان و شرایط دامنه اطلاعات و مدل محیط بستگی دارد. این مقاله به معرفی سیستم‌های نرم‌افزاری خودتطبیق و مروری بر مفاهیم پایه‌ی تطبیق خصوصا انواع رویکردهای فرآیند تصمیم‌گیری جهت تطبیق و مکانیسم انتخاب عملیات تطبیق اشاره دارد.

کلمات کلیدی

سیستم‌های خودتطبیق، ویژگی‌های کیفی، ویژگی‌های خود-*، فرآیندهای تطبیق، مکانسیم انتخاب عملیات تطبیق

۱- مقدمه

سیستم‌های نرم‌افزاری غالبا در محیط‌های متغیر و پویا هستند. شرایط ناخواسته مانند (خطاها و شکست‌ها در سیستم، تغییر در شرایط محیط عملیاتی، تغییر در ویژگی‌های غیر عملیاتی، تعریف نیازمندی‌های جدید، تغییر در قوانین و اهداف و ...در سیستم‌های نرم‌افزاری امروزی بسیار زیاد است. قبلاً بیشتر تغییرات در زمان توسعه به صورت آفلاین بود ولی اکنون به دلیل شرایط ناهمگن در محیط عملیاتی، فرکانس بالای تغییرات در شرایط محیط، اهداف، نیازمندی‌ها و ظهور مفاهیم جدیدی مانند اینترنت، حرکت[۲] و سیستم‌های همه‌جا حاضر[۳] و... شرایط را پیچیده‌تر تغییر کرده است. این پیچیدگی به صورت اساسی در نتیجه ساختار حلقه‌ی باز[۴] در فرآیند توسعه نرم‌افزارها می‌باشد. به همین علت، تقاضای زیادی برای مدیریت این پیچیدگی از طریق مدیریت خودکار سیستم‌ها وجود دارد که باعث به دست آوردن تمام نیازمندی‌های مورد انتظار با حداقل هزینه و زمان می‌شود. سیستم‌های خودمدیر یا خود تطبیق[۵] پاسخ به این تقاضاها هستند.

۲- سیستم‌های نرم‌افزاری خودتطبیق

از میان تعاریف برای سیستم‌های خودتطبیق BAA[۶] یک تعریف را بطور رسمی اعلام کرده است: نرم‌افزاری که رفتار و عملیات خود را ارزیابی کرده و عملیات و رفتار خود را تغییر می‌دهد هنگامی که درمی‌یابد آنچه انجام می‌دهد، مورد انتظار سیستم نرم‌افزاری نیست یا کارایی و نتیجه بهتری برای سیستم ممکن است. خودتطبیق کردن سیستم‌ها، در نقطه مقابل نگهداری آفلاین یک سیستم و یا عملیات کنترل یک سیستم، با

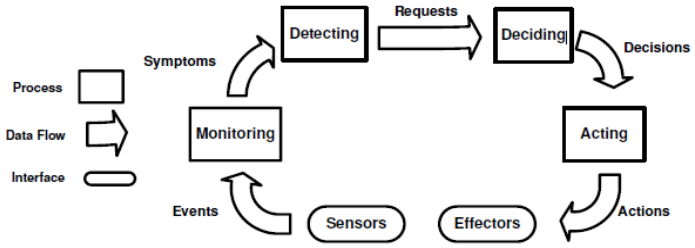
[۱] آذین بزم‌آرا

2 Mobility
3 Ubiquities
4 Open Loop
5 Self Managed or Self Adaptive(SA)
6 Broad Agency Announcement

ورژن‌گذاری آن است.

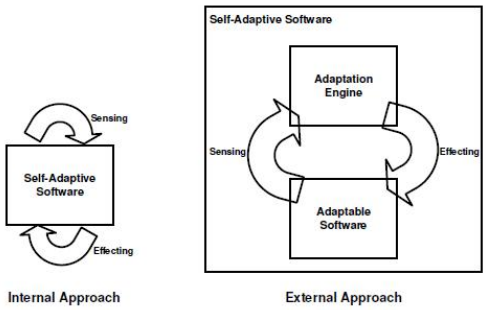
با این وجود منظور از تغییرات در اینجا، تغییر در محیط عملیاتی وسیستم نرم‌افزاری در زمان اجرای سیستم و درحالت عملیاتی سیستم می‌باشد. باید یادآوری گردد کلیه متغیرهای دخیل در سیستم عملیاتی پویا شامل ویژگی‌های وظیفه‌مندی، ویژگی‌های کیفی(ویژگی غیروظیفه‌مندی)، محدودیت‌ها و اهداف سیستم در فرایند تطبیق در نظر گرفته می‌شود ولی در این تحقیق بر روی اهداف و ویژگی‌های غیروظیفه‌مندی تمرکز می‌شود.

این سیستم‌ها یک حلقه بسته با یک چرخه باز خورد[۷] برای انجام عملیاتشان هستند. این تغییرات و نیازمندی‌ها می‌تواند از درون خود سیستم یا از محیط عملیاتی نشأت گرفته باشد (کاربران دیگر،سیستم‌های دیگر). به طور کلی بر اساس این چرخه‌ی بسته، مکانیسم تطبیق دارای ۴ فرآیند است شکل(۲). در فرآیند اول، این سیستم‌ها باید خودشان و محیط را مانیتور[۸](نظارت) کنند، سپس تغییرات را ردیابی[۹](آنالیز) کرده وتصمیم بگیرند[۱۰](طرح ریزی) چگونه واکنش نشان دهند وسرانجام تغییر را اجرا[۱۱]کنند. مکانیسم این فرایندها به اهداف سیستم[۱۲]، ویژگی‌های تطبیق[۱۳]، ترجیحات ذینفعان، شرایط و محدودیت‌ها، اطلاعات و مدل سیستم ومحیط عملیاتی بستگی دارد.



شکل(۱): چهار فرآیند تطبیق در سیستم‌های خودتطبیق[1]

در اینجا ما تمرکز بر روی سیستم‌های خود تطبیق نرم‌افزاری داریم. برای اینکار نرم‌افزار به صورت نرمال به عنوان یک سیستم حلقه باز ایجاد می‌شود و سپس با یک چرخه بازخورد[۱۴]، بسته خواهد شد. با این حلقه یک نگرش کلی از اینکه چه اتفاقی در سیستم می‌افتد ایجاد خواهد شد. این چرخه هم از محیط عملیاتی (هر آنچه در محیط عملیاتی بر رفتار و ویژگی‌های سیستم تأثیر می‌گذارد) و هم از خود سیستم نرم‌افزاری بازخورد خواهد داشت. معمولاً سیستم‌های خودتطبیق دارای ۲ بخش المان تطبیق‌پذیر[۱۵] و موتور تطبیق[۱۶] هستند. همچنین دارای یک پایگاه دانش[۱۷] هستند که اطلاعاتی درباره دامنه المان تطبیق‌پذیر، اهداف کیفی و ترجیحات ذینفعان است.

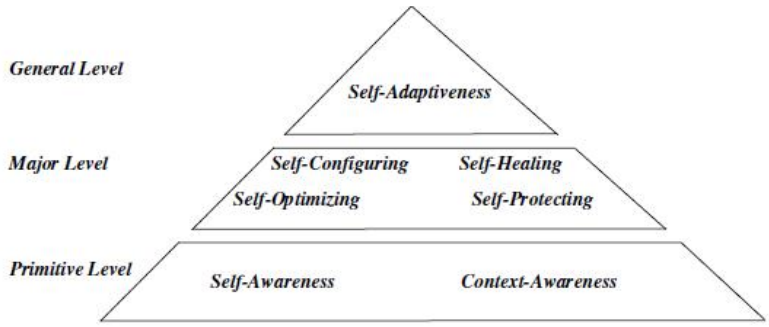


شکل(۲):المان تطبیق‌پذیر و موتور تطبیق به دو صورت داخلی و خارجی[2]

7 Feedback Loop
8 Monitor Process
9 Detect (Analysis) Process
10 Decision (Planning) Process
11 Act (Execute) Process
12 System Goals
13 Adaptation Attribute
14 Feedback Loop
15 Adaptable Element
16 Adaptive Engine
17 Knowledge-base

۲-۱- ویژگی های خودتطبیقی

سیستم‌های خودتطبیق قادرند اعمال خودتنظیمی[۱۸]، خودالتیامی[۱۹]، خودپیکربندی[۲۰]، خودنظارتی[۲۱]، خودتعمیری[۲۲]، خودبهینگی[۲۳]، خودحفاظتی[۲۴] و... را به طور خودکار انجام دهند. ویژگی‌های تطبیق ذکر شده در بالا و اولین بار IBM[3] این ویژگی‌های تطبیق را معرفی کرده است شکل(3). در این شکل ویژگی‌های تطبیق آورده شده است که در ۳ سطح تجزیه شده‌اند.



شکل(3):سلسله مراتب ویژگی‌های خود-* [3]

-**خودتنظیمی**[۲۵] :توانایی دوباره تنظیم شدن سیستم به صورت اتوماتیک و پویا در پاسخ به تغییرات هنگام نصب، به‌روزرسانی، تجمیع، ایجاد/جداشدن موجودیت‌های نرم‌افزاری.

-**خودالتیامی**[۲۶]:که با خود تعمیری ارتباط دارد، توانایی کشف، تشخیص و واکنش به اختلال در نرم‌فزار است. همچنین واکنش به مشکلات بالقوه مثلاً شکست در سیستم یا خطا در سیستم است.

-**خودبهینه‌سازی**[۲۷]:توانایی مدیریت کارایی و تخصیص منابع است. زمان پاسخ، بهره‌وری، بارکاری، توان‌عملیاتی، مهمترین فاکتورهای مرتبط به این ویژگی هستند.

-**خودحفاظتی**[۲۸]:توانایی شناسایی شاخص‌های امنیت و بازیابی از تأثیرات امنیتی است. که دو جنبه دارد یکی دفاع از سیستم در برابر حملات خرابکارانه و دومی انجام عملیات جلوگیری از بروز حملات.

سطوح ابتدایی:2

-**خودآگاهی**[۲۹]:یعنی سیستم از وضعیت درونی خودش آگاهی دارد.

-**محیط‌آگاهی**[۳۰] :یعنی سیستم از شرایط بیرونی و محیط عملیاتی‌اش آگاه است.

۲-۲- ارتباط میان فاکتورهای کیفی و ویژگی های خود تطبیقی

خود تنظیمی بالقوه نشان‌دهنده ویژگی‌های کیفی قابلیت نگهداری، وظیفه‌مندی، قابلیت‌جابجایی و قابلیت‌استفاده است. خود التیامی قابلیت دسترسی و قابلیت نگهداری و قابلیت اطمینان را افزایش می‌دهد. خود بهینه‌سازی با کارایی ارتباط دارد و خود حفاظتی باقابلیت اطمینان ارتباط دارد ویژگی های ابتدایی خود آگاهی و محیط آگاهی با قابلیت نگهداری، وظیفه‌مندی و قابلیت جابجایی در ارتباط هستند.

18 Self Tuning
19 Self healing
20 Self Configuration
21 Self Monitoring
22 Self Repairing
23 Self Optimizing
24 Self Protecting
25 Self Configuring
26 Self Healing
27 Self Optimizing
28 Self Protecting
29 Self Awareness
30 Context Awareness

۲-۳- فرآیند تطبیق

-**فرآیند مانیتورینگ**: مسئول جمع‌آوری داده‌ها از حسگرها و تبدیل آنها به نشانه‌ها و الگوهای رفتاری است. این پروسه می‌تواند با یک چک کردن حد آستانه‌ی ساده هم انجام شود.

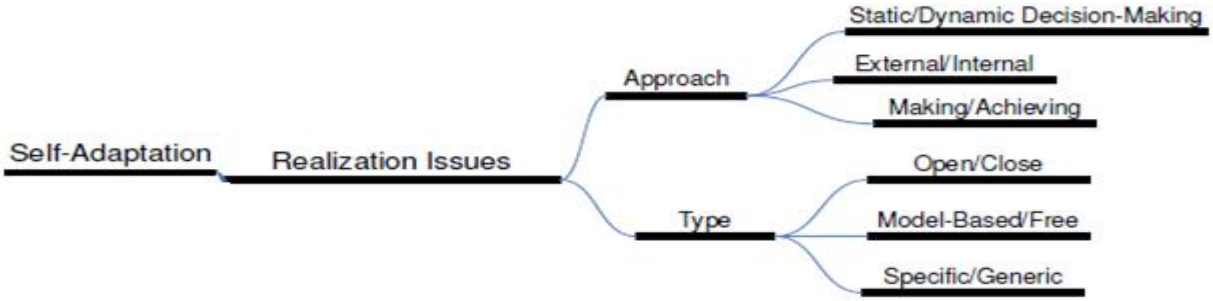
-**فرآیند شناسایی**: مسئول آنالیز نشانه‌های حاصل از مانیتورینگ و بررسی فایل‌های تاریخ‌نگاری[۳۱] در سیستم است.

-**فرآیند تصمیم‌گیری**: مسئول تعیین اینکه چه چیزی و چگونه باید تغییر کند، است. یکی از مهمترین فعالیت‌های این فرآیند تصمیم‌گیری در مورد مکانیسم انتخاب عملیات تطبیق است که در بخش [6-2] بیشتر به این موضوع پرداخته شده است.

-**فرآیند اجرا**: مسئول اجرای تغییر تعیین شده در فرآیند قبل است و شامل مدیریت عملیات غیر ابتدایی[۳۲] و ابتدایی[۳۳] برای انجام توسط بازوهای[۳۴] اجرا است .

۲-۴- انواع و رویکردهای تطبیق

بر اساس شکل(۴) تطبیق از منظر انواع و رویکردها دارای طبقه‌بندی زیر است که به آن اشاره می‌شود:



شکل(۴): طبقه‌بندی مفاهیم خودتطبیقی [2]

رویکردهای تطبیق

-**تصمیم‌سازی ایستا/پویا**: این قسمت به طور خاص با چگونگی فرآیند تصمیم‌گیری سر و کار دارد. در شکل ایستا، فرآیند تصمیم‌سازی در سیستم درونی و کد شده (مثلاً درخت تصمیم گیری) و تغییر آن نیازمند دوباره کامپایل کردن سیستم و کامپونت‌هایش است. در تصمیم‌سازی پویا سیاست‌ها[۳۵]، قوانین و کیفیت سرویس[۳۶] در خارج تعریف و مدیریت می شوند و بنابراین می توان آنها را در زمان اجرا تغییر داد و یا به صورت جدید آنها را تعریف کرد .

-**تطبیق داخلی/خارجی**: در روش داخلی برنامه کابردی مورد تطبیق و موتور تطبیق‌دهنده درون یک سیستم هستند. این روش بر اساس توانایی و ویژگی‌های زمان برنامه‌سازی ایجاد می‌شود و از نظر قابلیت نگهداری و مقیاس پذیری ضعیف است.. در روش خارجی از موتور تطبیقی خارجی استفاده می‌شود. موتور خارجی با کمک یک میان افزار یا یک موتور تولیدکننده سیاست‌ها، منطق تطبیق را پیاده می‌کند. بر این اساس مزیت روش خارجی این است که قابلیت تست و همچنین استفاده مجدداً افزایش می‌دهد .

ایجاد/دستیابی به تطبیق: در حالت کلی، خود تطبیقی درسیستم نرم‌افزاری از ۲ استراتژی استفاده می کند. اولین استراتژی مهندسی خود تطبیقی در سیستم در فاز توسعه و دومی دستیابی به آن از طریق یادگیری تطبیق است که اولی ایجاد[۳۷] و دومی دستیابی[۳۸] نامیده می‌شود. اولی از دیدگاه مهندسی و دومی از دیدگاه هوش‌مصنوعی به رفتار سیستم نگاه می‌کند.

انواع تطبیق

-**باز/بسته**: یک تطبیق بسته دارای تعداد محدودی عملیات تطبیق است و هیچ رفتار جدید یا جایگزینی در زمان اجرا را شامل نمی‌شود .

31 Log File
32Non Primitive
33 Primitive
34 actuator
35 Policy
36 QOS
37 Making
38 Making -Achieving

برعکس در تطبیق باز، سیستم می‌تواند توسعه یافته و نتیجتاً پیشنهاد جایگزین[۳۹] داشته باشد و یا حتی یک موجودیت جدید برای مکانیسم تطبیق تعریف شود.

-**مدل مبنا/آزاد**: در روش‌های آزاد یک مدل از پیش تعیین شده برای سیستم و یا شرایط محیط نداریم. در واقع با دانستن اهداف، نیازمندی‌ها وپیشنهادهای جایگزین مکانیسم تطبیق سیستم انجام می‌شود. برای مثال Dowling از یک یادگیری تشویقی بدون مدل برای تطبیق استفاده کرده است [4]. برعکس در تطبیق مدل مبنا، مکانیسم از یک مدل برای سیستم و محیط استفاده می‌کند. مدل‌های زیادی مانند مدل صف برای خود بهینه‌سازی، مدل معماری مبنا برای خود التیامی و یا مدل خاص دامنه وجود دارد.

-**خاص/عمومی**: بعضی روش‌ها راه‌حل خاص منظوره برای تطبیق را ارائه می دهد مانند یک دیتابیس.[IBM Smart] با این وجود روش‌های عمومی نیز وجود دارد که برای دامنه های مختلف می‌تواند به کار رود. روش‌های خاص منظوره تنها برروی تطبیق فرآورده یا ویژگی به‌جای سیستم نرم‌افزار تمرکز دارند.

۲-۵- اصول حمایت‌کننده در ایجاد سیستم خودتطبیق

به طور کلی سیستم خود تطبیق یک موضوع میان‌رشته‌ای است و از ایده‌هایی در مهندسی نرم‌افزار، هوش مصنوعی، تئوری تصمیم‌گیری، تئوری کنترل و محاسبات توزیع شده و شبکه خصوصا در فرآیند تصمیم‌گیری و مکانسیم انتخاب عمل تطبیق استفاده کرده‌است. در ادامه این اصول را دنبال می‌کنیم.

مهندسی نرم‌افزار

بین نیازمندی‌های کیفی و ویژگی‌های خود-* یک ارتباط وجود دارد. اصولاً برآورده‌کردن نیازمندی‌های کیفی محرک برای ایجاد تطبیق است. نیازمندی‌های وظیفه‌مندی نیز می‌تواند مرتبط با ویژگی‌های خودتطبیق باشد مثلاً تغییر در رفتار کامپوننت به یک سطح پایین‌تر برای ایجاد کارایی بالاتر که به آن مهندسی نیازمندی‌ها می‌گویند. چندین تحقیق از مدل‌سازی نیازمندی‌های غیروظیفه‌مندی خصوصاً مدل هدف[۴۰] در سیستم خودتطبیق استفاده کرده‌اند مثل [5]. به دلیل تفاوت میان نرم‌افزارهای خود تطبیق و نرم‌افزارهای سنتی، روش‌های رسمی سنتی برای سیستم‌های تطبیق پذیر به کار نمی‌رود. البته روش‌هایی بر مبنای مدل‌های رسمی، مانند محاسبات مدل تجمیع[۴۱]برای تغییرات زمان اجرا به کار رفته است. مدل معماری نرم‌افزار در زبان‌هایی مثل زبان‌های توصیف معماری[۴۲] برای مدل کردن و مدیریت نرم‌افزار در زمان اجرا مفیدند. Bradbury، مروری بر انواع ADLها برمبنای گراف‌ها، فرآیندهای جبری و روش‌های رسمی دیگر برای معماری نرم‌افزارهای پویا داشته است[6]. garlan از زبان Acme برای توصیف معماری قابل تطبیق جهت یافتن مغایرت‌ها با محدودیت‌های تعریف شده استفاده کرده است [7]. مهندسی نرم‌افزار کامپوننت مبنا[۴۳] نیز به توسعه‌دهندگان در ۲ جهت کمک می‌کند: اول ساده‌ترسازی، طراحی و پیاده‌سازی بر مبنای مدل کامپوننتی و دوم قابلیت استفاده مجدد از موتور تطبیق که با ماژولار بودن بدست می‌آید. همچنین برنامه‌نویسی جنبه‌گرا[۴۴] و مخصوصاً ADP پویا می‌تواند در کپسوله کردن ارتباط تطبیق برای تطبیق در زمان اجرا به کار رود[8]. معماری سرویس‌گرا(SOA) در ساده‌کردن ایجاد سرویس‌های ضعیف–پیوسته[۴۵] و تکنولوژی وب سرویس‌ها گزینه مناسبی برای پیاده سازی کسب و کارهای قابل تطبیق پویا به جهت انعطاف پذیری آنها هستند.

هوش مصنوعی

به طور کلی تکنیک های AI مثل طرح‌ریزی[۴۶] و استدلال احتمالی[۴۷] در توسعه و مدیریت سیستم‌های نرم‌افزاری به کار گرفته نشده است ولی سیستم‌های خودتطبیق یک زمینه جالب توجه در AI هستند که مثلاً در فرآیندهای کشف، AI می‌تواند در آنالیز نشانه‌ها یا الگوها برای تشخیص شرایط مشکوک بکار برده شود. همچنین AI برای فرآیند تصمیم‌گیری یا طرح‌ریزی، استدلال و یادگیری بسیار مفید است. در این روش‌ها سیستم نرم‌افزاری عملیات تطبیق را بارها طرح‌ریزی می‌کند به جای اینکه از یک الگوریتم خاص استفاده نماید و این مسأله در فرآیند تصمیم‌گیری درست انتخاب عملیات مناسب تطبیق استفاده می‌شود. تطبیق بر اساس طرح‌ریزی باید در هر زمان فعال باشند. یعنی موتور از طریق طرح‌ریزی‌های اتفاقی و یا طرح‌ریزی مجدد در هر زمان پیوسته طرح‌ریزی می‌کند. اولی یک طرح‌ریزی شرطی براساس اطلاعات حس شده را برای مسیرهای

39 Alternative
40 Goal Model
41 Model Integrated Computing (MIC)
42Architecture Description Language(ADL)
43 Component Base Software Engineering(CBSE)
44 Aspect Oriented Programming(AOP)
45 Loosely Coupled
46 Planning
47 Probabilistic Reasoning

جایگزین ایجاد می‌کند و دومی طرح‌های جایگزین در مواردی که طرح اصلی دچار شکست یا خطا شده را ایجاد می‌کند. یک مفهوم مهم در نرم‌افزارهای خود تطبیق استفاده از عامل نرم‌افزاری، مدل‌ها، دامنه، اهداف و ویژگی‌ها و روش‌های تصمیم‌گیری در آنهاست. یک موضوع دیگر در اینجا MAS[۴۸] است که مدل‌های هماهنگی و همکاری و تکنیک‌های بهینه سازی توزیع‌شده برای سیستم‌های SA است. در این سیستم‌ها اهداف محلی و سراسری تعریف می‌شود. نمونه‌های معماری‌های تعریف شده شامل [9]هستند. یادگیری ماشین و محاسبات نرم از حوزه‌های بالقوه برای استفاده در SA هستند. مخصوصاً در روش "دستیابی" بکار برده می‌شوند. یادگیری AI مبنا و برنامه نویسی‌های تکاملی[۴۹] در دسته روش‌های تغییر بی‌تاریخچه[۵۰] هستند. این الگوریتم از ویژگی‌های محیط و دانش دریافت شده از تلاش‌های قبلی برای تولید الگوریتم جدید استفاده می‌کنند. الگوریتم‌های ژنتیک و الگوریتم‌های آنلاین دیگر مثل یادگیری تقویتی[۵۱] نیز می‌توانند برای این هدف استفاده شوند. RL یک روش مناسب انتخاب عملیات دینامیک است [10]. همچنین برای سیستم‌های توزیع شده و خودتطبیق مناسب است. tesauro گفته که RL قابلیت دستیابی به کارایی بالاتر نسبت به متدهای سنتی دارد و درحالیکه به دانش دامنه کمتری نیز نیازمند است [11]. روش‌های فازی نیز برای آدرس دهی برای کیفیت، اهداف، سیاست‌ها به صورت فازی مناسبند[12]. درعمل به خاطر عدم قطعیت، استدلال احتمالی و تئوری تصمیم‌گیری در طرح‌ریزی‌ها ما به تصمیم سازی نیازمندیم. فرآیند تصمیم‌گیری مارکوف (MDP) و شبکه‌های بیزین دو تکنیک خوش‌ساخت برای این هدفند. این تکنیک‌ها برای درک ویژگی‌های خودتطبیق بخاطر ویژگی‌های غیرقطعی‌شان قابل اعمال هستند.

مهندسی/تئوری کنترل

مهندسی/تئوری کنترل، مشابه سیستم‌های خودتطبیق با سیستم‌هایی سروکار دارد که مرتباً با محیط اطراف از طریق یک چرخه بازخورد خود تعامل دارند. سیستم‌های کنترل دارای۲ بخش، ماشین قابل کنترل و موتور کنترل است. که اولی دارای ۲ دسته ورودی است. ورودی کنترل که رفتار سیستم را کنترل می‌کند و دوم پارازیت که رفتار سیستم را به سوی حالت ناخواسته هدایت می‌کند. هدف کنترل مبنا معمولاً بر روی یک مدل از سیستم قابل کنترل است. مثلاً مدل صف‌بندی لایه‌ای سلسله مراتبی[۵۲]که برای سیستم نرم‌افزاری جهت تنظیم‌کردن[۵۳] پارامترها به کار می‌رود. اگرچه چرخه های بسته بیشتر برای مدل‌سازی سیستم‌های کنترل خودتطبیق بکار می‌رود، مدل‌های قابل تنظیم‌مجدد و قابل سازگاری هم استفاده می‌شود. به صورت سنتی تئوری کنترل برای سیستم‌هایی که قوانین فیزیک بر آنها حاکم است به کار می‌رود و به همین دلیل معمولاً کاربرد آنها برای سیستم‌های نرم‌افزار پیچیده‌تر از سیستم‌های کنترل سنتی است و در آنها حضور یا عدم حضور ویژگی‌هایی چک می‌شود که معمولاً از جنس نرم‌افزاری نیستند.

محاسبات شبکه‌ای و توزیع شده

تکنیک‌های استفاده شده در محاسبات توزیع‌شده و شبکه‌ای می‌تواند برای سیستم‌های خودتطبیق توسعه یابند زیرا قسمت عمده سیستم‌های نرم‌افزاری توزیع‌شده یا شبکه‌ای هستند. خط دیگر تحقیقات در این زمینه شامل برنامه‌های کاربردی نقطه به نقطه[۵۴] و شبکه‌های خاص‌منظوره[۵۵] که با تغییرات در محیط معماری و کیفیت و نیازمندی‌ها مواجهند است. مدیریت سیاست مبنا یکی از روش‌های بسیار موفق در شبکه و محاسبات توزیع شده است. مدیریت سیاست مبنا[۵۶] توصیف می‌کند که چگونه با موقعیت‌هایی که ممکن است اتفاق بیفتند برخورد شوند(اولویت بندی و قوانین دستیابی به منافع سیستم). این تکنیک‌ها دارای قوانین رویداد–شرط–عمل است و می‌تواند دارای سیاست‌های هدف (شرایط هدف) و سیاست‌های سودمندی (ارزش هر وضعیت) باشد. سیاست‌های تغییر[۵۷] براساس تغییرات محیط یا نیازمندی‌ها باید بتواند تغییر کند که برای این کار از تکنیک‌های تئوری کنترل استفاده شده است. همچنین مدیریت سیاست مبنا برای چندین سیستم خودتطبیق انتخاب شده‌است[8]. یک تکنولوژی دیگر که از سیستم‌های شبکه‌ای گرفته شده‌است میان‌افزار است . تطبیق بر اساس میان‌افزار برای فرآیند های تطبیق قابل بکاگیری است. برای مثال کامپوننت های عمومی در تصمیم‌سازی و کشف تغییر می‌تواند در سطح میان‌افزار پیاده‌شود. همچنین مفهوم دیگری که از شبکه و سیستم‌های توزیع‌شده وارد این حیطه شده است مفهوم مجازی سازی[۵۸]است . مجازی سازی دامنه عمل موتور تطبیق را کم می‌کند و نتیجتاً

48 Multi Agent System
49 Evolutionary
50 History less
51 Reinforcement Learning
52 LQM
53 Tune
54 Peer-To- Peer
55 adhoc
56 Policy Base
57 change policy
58 Virtualization

مدیریت منابع پویا را آسانتر می کند همچنین یک راه کارا برای سیستم‌های موروثی برای کارکردن در محیط‌های عملیاتی جدید است.

۲-۶- فرآیند تصمیم‌گیری ومکانیسم‌های انتخاب عملیات

در زمینه روش‌های مختلف برای فرایند تصمیم‌گیری و مکانیسم انتخاب عملیات در سیستم‌های خودتطبیق تحقیقات زیر انجام شده است:
پویایی از پیش برنامه ریزی شده[۵۹] از نمونه اولیه و پایه‌ای از تطبیق است که تغییرات ثابت، بر اساس شرایط از پیش مشخص، در سیستم تعریف شده وبراساس نیاز اجرا می‌شود. این روش خودتطبیقی که بر مبنای قوانین اگر- آنگاه از پیش نوشته شده است را، تطبیق قانون مبنا یا سیاست مبنا[۶۰] می‌گویند. در این رویکرد طرح‌های تصمیم‌گیری معمولا به صورت رخداد-شرط – عمل[۶۱] (ECA) در سیستم تعریف می‌شود.. از عیب‌های بیان اهداف با ECA اینست که این روش‌ها معمولا اهداف چندگانه در سیستم را به‌سختی مدیریت می‌کنند، همچنین وقتی تعداد اهداف زیاد شود احتمال ناسازگاری[۶۲] بالا می‌رود(مثلا جایی که تعداد محدودی منبع داریم و چندین لایه از برنامه به منبع احتیاج دارند) در این حالت به یک مکانیسم حل ناسازگاری نیاز داریم. اضافه کردن یا حذف اهداف در این سیستم‌ها سخت است و در کل این روش انعطاف پذیر و مقیاس‌پذیر نیست. در تحقیق دیگر با استفاده از رویکرد هدف -مبنا[۶۳] مکانسیم انتخاب عملیات تطبیق را مدل‌سازی کرده است[13] وان را با روش قانون مبنا بر روی یک سیستم شبیه‌سازی شده مقایسه کرده است. این روش بی حافظه[۶۴] است و تنها از روی حالت جاری تصمیم‌گیری می‌کند. در واقع نگهداری اطلاعات جهت استدلال‌های بعدی و یادگیری جهت بهتر شدن فرآیند در ان وجود ندارد. همچنین این روش مزایای استفاده از الگوریتم‌های هوشمند جهت خودکارسازی یا بهینه‌سازی را ندارد. یادگیری تقویتی[۶۵] روش دیگری برای افزایش کارایی فرآیند تصمیم‌گیری در سیستم خودتطبیق است. در این روش از یادگیری نتایج قبلی، برای بدست آوردن قواعد(سیاست‌ها) از عملیات سیستم، استفاده می‌شود. مزیت این روش این است که به مدل صریحی که سیستم آن را مدیریت کند نیازی ندارد [14]. ولی مقیاس‌پذیری ضعیفی دارد و فضای حالتٍ دارای وضعیت‌های زیاد را نمی تواند نگهداری کند. بهمین دلیل چندین مدل ترکیبی معرفی شده است .همچنین در هر مرحله از انتخاب عملیات تنها یک عمل را انتخاب می‌کند.

یکی دیگر از کارهای انجام شده ایجاد یک فریمورک به‌نام FUSION است که شامل مدل دیگری از نمایش سیستم نرم‌افزاری خودتطبیق، در نرم‌افزارهای سیسنم خط تولید[۶۶] بر اساس مفهوم خصیصه گرایی[۶۷] و سپس استدلال جهت تصمیم‌گیری با استفاده از یادگیری آنلاین را معرفی کرده است [15]. این روش بر افزایش دقت و کارایی تصمیم‌گیری در سیستم خودتطبیق تمرکز دارد.

به طور کلی تکنیک های AI در فرآیند تصمیم‌گیری در سیستم های نرم‌افزاری خودتطبیق به میزان کمی بکار گرفته شده است. یکی از تکنیک‌ها الگوریتم‌های تکاملی از جمله الگوریتم ژنتیک است که امروزه به طور وسیعی در حوزه‌های مختلف از آن استفاده می‌شود.کاربردهای این الگوریتم در مسائل بهینه‌سازی بسیار مورد توجه بوده است لذا در این تحقیق این الگوریتم جهت بهینه‌سازی سیستم‌های خودتطبیق در فرایند تصمیم‌گیری و مکانیسم انتخاب عملیات مورد استفاده قرارگرفته است.

جدول (۱): مقایسه روش‌های مکانسیم انتخاب عملیات در سیستم‌های خودتطبیق

عنوان روش	محقق	ویژگی	سال	مزایا	معایب
Rule-base (Policy-base)	Kephart	این روش بعنوان روش پایه، استفاده از قوانین اگر- آنگاه به صورت Event-Condition-Action تعریف شده در پایگاه دانش، جهت انجام فرایند (ECA)	2004	به دلیلی سادگی در بیشتر پروژه‌های عملی از این روش استفاده می‌شود	عدم انعطاف‌پذیری عدم مقیاس‌پذیری

59 Programmed Dynamism
60 Rule-base - Policy Base
61 Event-Condition-Action
62 Conflict
63 Goal-Driven
64 Stateless
65 Reinforcement Learning
66 Product Line
67Feature Oriented

		تصمیم‌گیری است			
Reinforcement Learning[16]	Amoui	استفاده از روش یادگیری تقویتی جهت ساختن سیاست‌های تصمیم‌گیری و انتخاب عملیات تطبیق، بر اساس بازخورد تنبیهی یا تشویقی عملیات تطبیق انجام شده در گذشته	2008	-عدم نیاز به تعریف مدل -افزایش صریح سیستم کارایی سیستم	زمان طولانی یادگیری و خصوصا عدم اطمینان به تصمیمات سیستم در فاز یادگیری
GAAM[17]	Salehie	مدل‌سازی با رویکرد هدف –مبنا جهت مکانسیم انتخاب عملیات تطبیق	۲۰۱۱	مدل‌سازی سیستم خودتطبیق و استفاده از روش‌های وزن‌دهی به اهداف جهت تصمیم‌گیری	-عدم استفاده از الکوریتم‌های خودکار و هوشمند در تصمیم‌گیری -عدم بهینه‌سازی
FUSION[15]	Esfahani	در این روش ابتدا سیستم با رویکرد خصیصه مبنا مدل‌سازی شده و سپس با روش‌های یادگیری انلاین تصمیم‌گیری انجام می‌دهد (این روش در سیستم‌های نرم‌افزاری خط تولید استفاده می‌شود)	۲۰۱۳	اضافه نمودن یادگیری آنلاین به فرآیند تصمیم‌گیری	-عدم تضمین بهینگی -عدم استفاده از الگوریتم‌های خودکار و هوشمند در تصمیم‌گیری

۳- نتیجه‌گیری و جمع‌بندی

اهمیت موضوع سیستم‌های نرم‌افزاری خودتطبیق درحال گسترش است و این دلیل محیط‌های پویا و دایما در حال تغییر امروزی است به همین جهت نیاز است تحقیقات بیشتری در این زمینه انجام گرفته و تئوری‌ها و مدل‌های جدید، چالش‌های پیش روی آن را از میان بردارد. این مقاله مروری بر مفاهیم پایه مرتبط با سیستم‌های نرم‌افزاری خود تطبیق بوده و پس از معرفی فرآیندهای مرتبط و خصوصا انواع رویکردهای تطبیق در فرآیند تصمیم‌گیری و مکانیسم انتخاب عملیات مقایسه‌ای بر روی روش‌های انجام گرفته تا کنون را ارایه می‌دهد.

۴- مراجع

[1] J.O. Kephart, D.M. Chess, The vision of autonomic computing. *IEEE Computer*, 2003.

[2] M.Salehie, L. Tahvildari, Self-Adaptive Software:Landscape and Research Challenges, Software *Technologies Applied Research (STAR) GroupUniversity of Waterloo*, 2009.

[3] IBM-AC. Autonomic computing 8 elements,2001, http://www.research.ibm.com/autonomic/overview/elements.html.

[4] J. Dowling,V. Cahill, Self-managed decentralised systems using K-components and collaborative reinforcement learning, In *Proc. of ACM Workshop on Self-Managed Sys.* 39{43.2004.

[5] A. Lapouchnian, S. Liaskos, J. Mylopoulos ,Y. Yu, Towards requirements-driven autonomic systems design, . In *Proc. of Workshop on Design and* Evolution *of Autonomic App.Software*. 1{7. 2005.

[6] J. S. Bradbury, J. R Cordy, J. Dingel, M. Wermelinger, A survey of self-management in dynamic software architecture speci¯cations. In *Proc. of ACM workshop on Self-managed systems*. 28{33. 2004.

[7] D. Garlan, S.-W. Cheng, A.C. Huang, B. Schmerl, P. Steenkiste, Rainbow:Architecture-based self-adaptation with reusable infrastructure. *IEEE Computer 37*, 10, 46{54. 2004.

[8] M. Salehie, L.Tahvildari, S. Asadollahi, L. Change support in adaptive software: A case study for ne-grained adaptation. In *Proc. of IEEE Conf. and Workshops on Engineering of Autonomic and Autonomous Systems*. TBA. . 2009

[9] D.Weyns, K.Schelfthout, T.Holvoet, Architectural design of a distributed application with autonomic quality requirements. *SIGSOFT Softw. Eng. Notes 30*, 4, 1{7. 2005.

[10] M. Amoui, , M. Salehie, , S. Mirarab, L.Tahvildari, Adaptive action selection in autonomic software using reinforcement learning. In *Proc. of Int. Conf. on Autonomic and Autonomous Systems*. 175{181. 2008

[11] G. Tesauro, Reinforcement learning in autonomic computing: A manifesto and case studies. *IEEE Internet Computing 11*, 1, 22{30, 2007

[12] M. Salehie, L. Tahvildari, A policy-based decision making approach for orches- trating autonomic elements. In *Proc. of IEEE Int. Workshop on Software Tech. & Eng. Prac.* 173{181.2005. 2005.

[13].M, Salehie,L. Tahvildari, Toward a Goal-Driven Approach to Action Selection in Self-Adaptive Software, Wiley Online Library,*IEEE*,2011.

[14] J. Dowling, E. Curran, R. Cunningham, V.Cahill, Building autonomic systems using collaborative reinforcement learning. Knowle. *Engin. Rev. J. Special Issue on Autonomic Computing. Cambridge University Press*.2006.

[15].N.Esfahani, A Learning-Based Framework for Engineering Feature-Oriented Self-adaptation Software System, *IEEE*, ۲۰۱۳.

[16],M. Amoui, M. Salehie,S. Mirarab, ,L.Tahvildari ,Adaptive Action Selection in Autonomic Software Using Reinforcement Learning,*IEEE*,2008.

[17] M.Salehie, L.Tahvildari, Toward a Goal-Driven Approach to Action Selection in Self-Adaptive Software, Wiley Online Library,*IEEE*,2011