نشریه علمی- ترویجی محاسبات نرم

شمارهٔ ششم/ بهار و تابستان ۹۴/ صفحهٔ ۱۴ـ۲۹

دریافت مقاله: ۹۳/۰۶/۱۰

پذیرش مقاله: ۹۳/۰۹/۷

ارائهٔ راهکار ترکیبی به‌منظور بهبود و توسعهٔ متدولوژی‌های عامل‌گرا

عرفان قندهاری[۱]، فاطمه سعادت‌جو*[۲]، محمدعلی زارع چاهوکی[۳]

[۱] دانشجوی کارشناسی ارشد، دانشکدهٔ فنی و مهندسی، دانشگاه علم و هنر، یزد، ایران

erfan.ghandehari@sau.ac.ir

[۲] استادیار، دانشکدهٔ فنی و مهندسی، دانشگاه علم و هنر، یزد، ایران

saadatjou@sau.ac.ir

[۳] استادیار، دانشکدهٔ برق و کامپیوتر، دانشگاه یزد، یزد، ایران

chahooki@yazd.ac.ir

چکیده: مهندسی نرم‌افزار عامل‌گرا یکی از زمینه‌های جدید و در حال توسعهٔ علوم کامپیوتر است که در قالب متدولوژی‌های عامل‌گرا، رویکردهای سیستماتیک برای تحلیل، طراحی، پیاده‌سازی و نگهداری سیستم‌های چندعاملی ارائه می‌دهد. در بحث مهندسی نرم‌افزار عامل‌گرا یکی از چالش‌های مهم و اساسی این است که علی‌رغم متدولوژی‌های متعددی که در این حیطه معرفی شده، هنوز کاستی‌هایی در فازهای مختلف متدولوژی‌ها وجود دارد. در این مقاله با توجه به نقاط ضعف متدولوژی‌های عامل‌گرا در فازهای مختلف و تأثیر بسزایی که این کاستی‌ها می‌تواند در کیفیت و کارایی پروژه‌های نرم‌افزاری داشته باشد، به ارائهٔ راهکاری ترکیبی برای معرفی متدولوژی (IR) از متدولوژی‌های **RICA** و **INGENIAS** خواهیم پرداخت. بدین منظور با توجه به اینکه متدولوژی **INGENIAS** مراحل طراحی را به‌خوبی پشتیبانی می‌کند و مرحلهٔ تحلیل در آن به‌دلیل مدل نکردن نقش‌های سیستم، کامل نیست و از طرف دیگر با توجه به اینکه متدولوژی **RICA** مرحلهٔ مدل کردن نقش‌های سیستم را به‌خوبی پشتیبانی می‌کند، با استفاده از مزایای دو متدولوژی یک متدولوژی ترکیبی ارائه می‌گردد که در آن، از مرحلهٔ تحلیل و طراحی متدولوژی **INGENIAS** و مرحلهٔ تحلیل متدولوژی **RICA** استفاده می‌شود. همچنین به‌منظور افزایش کارایی متدولوژی پیشنهادی مدل‌های قابلیت، برنامه‌ریز و دانش نیز به مدل‌های موجود در متدولوژی پیشنهادی اضافه شده و به‌منظور تشریح فازهای مختلف آن، در یک مطالعهٔ موردی (سیستم خرید و فروش مسکن) مورد استفاده قرار گرفته است. همچنین به‌منظور سنجش توانایی متدولوژی پیشنهادی، این متدولوژی براساس معیارهای مفاهیم و ادراک‌ها و عمل‌گرایی ارزیابی شده است.

واژه‌های کلیدی: مهندسی نرم‌افزار عامل‌گرا، سیستم‌های مبتنی بر عامل، **RICA**، **INGENIAS**.

۱. مقدمه

مهندسی نرم‌افزار عامل‌گرا[1] نوعی مهندسی است که انتزاع[2] اصلی آن عامل است؛ به عبارتی، عامل‌ها اصلی‌ترین موجودیت‌های تشکیل‌دهندهٔ این نرم‌افزارها هستند. رویکرد عامل‌گرا برای مهندسی نرم‌افزار به معنی تجزیهٔ مسئله به چند عامل خودمختار[3] و تعامل‌کننده[4] است که این عامل‌ها برای رسیدن به اهدافی که برای آن طراحی شده‌اند، با یکدیگر و با محیط تعامل می‌کنند [۱].

مهندسی نرم‌افزار عامل‌گرا به‌منظور پاسخ‌گویی به نیازهای اساسی مهندسی نرم‌افزار و محاسبات مبتنی بر عامل مطرح شد [۲]. هدف اصلی مهندسی نرم‌افزار عامل‌گرا ایجاد متدولوژی، ابزار و امکاناتی برای تهیهٔ ساده و نگهداری از نرم‌افزارهای مبتنی بر عامل است [۳]. ازآنجاکه مهندسی نرم‌افزار شیءگرا قادر به پاسخ‌گویی نیاز نرم‌افزار مبتنی بر عامل نبود، نیاز مبرم به مهندسی جدید و هماهنگ با دیدگاه عامل‌ها به‌وجود آمد که باعث توسعهٔ مهندسی نرم‌افزار عامل‌گرا از مهندسی نرم‌افزار شیءگرا گردید [۴]. در بحث مهندسی نرم‌افزار عامل‌گرا یکی از چالش‌های مهم و اساسی فقدان متدولوژی توسعهٔ نرم‌افزاری کامل برای سیستم‌های براساس عامل است. گرچه تعداد زیادی متدولوژی عامل‌گرا پیشنهاد شده است، تعداد کمی از آن‌ها به‌طور کامل همهٔ فعالیت‌ها در مهندسی نرم‌افزار را پوشش می‌دهند و هیچ‌کدام از آن‌ها به‌طور کامل، نیازمندی‌های توسعهٔ سیستم‌های مبتنی بر عامل را پشتیبانی نمی‌کنند. بنابراین در حال حاضر نیاز به توسعهٔ متدولوژی‌ها به‌منظور رسیدن به متدولوژی‌های یکپارچه و جامع، کاملاً ضروری به نظر می‌رسد [۵ـ۸]. در ادامهٔ کارهای صورت‌گرفته در توسعهٔ متدولوژی‌های عامل‌گرا که بیشترین اهمیت را در بین توسعه‌های صورت‌گرفته دارند، بررسی خواهیم کرد.

Zambonelli و همکاران توانایی مدل کردن سیستم‌های قابل پیاده‌سازی در اینترنت را به متدولوژی GAIA اضافه کردند. در این پژوهش با توجه به ویژگی‌های باز بودن و تضاد اهداف در عامل‌ها، قابلیت مدل‌سازی روابط بین عامل‌ها به متدولوژی اضافه شده است [۹]. Jaun و همکاران نیز در سال ۲۰۰۲ با اضافه کردن ساختار سلسله‌مراتبی نقش‌ها، ایجاد مدل رسمی برای محیط سیستم و ایجاد توانایی مدیریت تغییرات پویا به متدولوژی GAIA، متدولوژی جدیدی با عنوان ROADMAP ارائه کردند [۸]. بهبود دیگری که بر روی متدولوژی GAIA صورت گرفت، توسط Garcia و همکاران انجام شد که با استفاده از ترکیب این متدولوژی با روش AUML[5] مراحل تعامل، عامل، پروتکل و مدل توسعه‌یافته UML را بهبود دادند [۱۰]. Gonzalez و همکاران نیز با اضافه کردن مراحل طراحی عامل و همچنین رویکرد مبتنی بر تکرار بهبودهایی در متدولوژی GAIA به‌وجود آوردند [۱۱]. کارهای توسعهٔ متدولوژی‌های عامل‌گرا تنها به متدولوژی GAIA محدود نبوده است و کارهای توسعه‌ای بر روی متدولوژی‌های MASE و TROPOS و همچنین متدولوژی‌های دیگر ادامه دارد. در یکی از بهبودهای متدولوژی MASE، مرحلهٔ مدل‌سازی هستان‌شناسی[6] به مرحلهٔ تحلیل متدولوژی توسط Dileo و همکاران اضافه شده است [۵]. همچنین امکان مدل‌سازی روابط سازمانی میان عامل‌ها نیز در توسعهٔ دیگری به متدولوژی MASE اضافه شده است [۱۲]. در متدولوژی TROPOS به‌منظور بهبود، یک مدل رسمی تحلیل هدف به متدولوژی اضافه شده است [۱۳]. همچنین یک روش تخصیص هدف به نقش نیز در این متدولوژی ارائه شده است [۱۴]. آنچه در توسعهٔ این متدولوژی‌ها و کارهای توسعه‌ای دیگر به‌عنوان چالش محسوب می‌شود، این است که هنوز متدولوژی‌های توسعه‌یافته نمی‌توانند همهٔ فعالیت‌های مهندسی نرم‌افزار را پوشش دهند و به‌منظور توسعهٔ نسل بعدی متدولوژی‌های عامل‌گرا لازم است که کارهای تحقیقاتی بیشتری به‌منظور توسعهٔ متدولوژی‌های دیگر صورت گیرد تا در خلال آن بتوان با ایجاد همگرایی بیشتر در بین فازهای تحلیل و طراحی

1. Agent Oriented Software Engineering
2. Abstraction
3. Autonomous
4. Interacting
5. Agent Unified Modeling Language
6. Ontology Population

متدولوژی‌های عامل‌گرا، امکان تطبیق این متدولوژی‌ها با سیستم‌های چندعامله را افزایش داد. با توجه به این چالش‌ها در این مقاله سعی شده که با تفکیک مناسب مدل‌ها در فازهای مختلف تحلیل و طراحی، همگرایی مناسبی را بین این فازها ایجاد کرد. این همگرایی به میزان چشمگیری توانایی‌های متدولوژی پیشنهادی را که از ترکیب متدولوژی‌های RICA[۱۵] و INGENIAS [۱۶] به‌دست می‌آید، افزایش می‌دهد و زمینه را برای توسعهٔ نسل بعدی متدولوژی‌های عامل‌گرا فراهم می‌آورد؛ البته برای رسیدن به این هدف بایستی تحقیقات به‌منظور بهبود و توسعهٔ متدولوژی‌های عامل‌گرا ادامه داشته باشد.

در ادامه به‌منظور تفکیک رویکردهای عامل‌گرا و شیءگرا به بررسی تفاوت‌های عامل‌ها و اشیا خواهیم پرداخت. سپس در بخش سوم به معرفی سیستم انتخاب‌شده در این مقاله اشاره خواهیم کرد. در بخش چهارم به معرفی متدولوژی پیشنهادی و تشریح ویژگی‌ها می‌پردازیم. در بخش پنجم به مقایسهٔ متدولوژی‌ها براساس شاخص‌های، مفاهیم و ادراک‌ها و عمل‌گرایی خواهیم پرداخت و در نهایت در بخش ششم به نتیجه‌گیری و راهکارهای آینده اشاره خواهیم کرد.

۲. مقایسهٔ رویکرد شیءگرا و عامل‌گرا

مهندسی نرم‌افزار عامل‌گرا تکاملی از نرم‌افزار شیءگراست. به‌عبارت دیگر می‌توان گفت عامل‌ها از اشیا مشتق شده‌اند [۱۷]. LIND در سال ۲۰۰۱ [۱۸] به مقایسهٔ سیستم‌های شیءگرا و عامل‌گرا در پنج سطح سخت‌افزار، تئوری، زمان اجرا[1]، زبان‌های برنامه‌نویسی[2] و زبان‌های طراحی پرداخته است که در ادامه، برخی از تفاوت‌ها بیان شده است: ۱. اشیا دارای سازمان متمرکزی هستند، درحالی‌که عامل‌ها محاسبهٔ توزیعی را مجاز می‌دانند. ۲. یکپارچگی اشیا در سیستم بیشتر از یکپارچگی عامل‌هاست. ۳. عامل‌ها نمی‌توانند مانند اشیا آزادانه ایجاد شده و از بین بروند. ۴. رفتار و ساختار شی ثابت است و تغییر نمی‌کند، درحالی‌که عامل‌ها از تجربه‌هایشان یاد می‌گیرند و رفتارشان را تغییر می‌دهند. ۵. تعاملات اشیا اکثراً توسط شیء دیگر درخواست می‌شود، درحالی‌که تعاملات عامل‌ها شامل واکنش به وقایع محیطشان یا درخواست از سایر عامل‌هاست. ۶. تعاملات اشیا معمولاً همگام است، درحالی‌که تعاملات عامل‌ها معمولاً ناهمگام است. ۷. کپسوله‌بندی عامل قوی‌تر از کپسوله‌بندی شیء است.

البته ازآنجاکه عامل‌ها از اشیا مشتق شده‌اند، می‌توان گفت شباهت‌هایی نیز بین عامل‌ها و اشیا وجود دارد. با وجود این تفاوت‌ها و شباهت‌ها می‌توان پارامترهایی از هر دو رویکرد را به هم نگاشت کرد. جدول (۱) یک نگاشت نوعی بین رویکردهای شیءگرا و عامل‌گرا را نشان می‌دهد.

جدول (۱): نگاشت بین رویکردهای عامل‌گرا و شیءگرا [۱۸]

رویکرد شیءگرا	رویکرد عامل‌گرا
کلاس انتزاعی	نقش کلی[3]
کلاس	نقش مخصوص یک قلمرو[4]
متغیرهای کلاس	دانش، باور
متد	قابلیت
وراثت[5]	انقیاد نقش[6]
نمونه‌سازی	نقش خاص+ دانش فردی
ترکیب	عامل‌های سلسله‌مراتبی
فرخوانی متد	تبادل پیغام
همکاری	تعامل

براساس جدول (۱) می‌توان گفت که رویکرد عامل‌گرا برای تمامی قابلیت‌های رویکرد شیءگرا راهکار ارائه کرده است. این راهکارها برای تحلیل و طراحی سیستم‌های مبتنی بر عامل مناسب‌اند.

۳. مطالعهٔ موردی: سیستم خرید و فروش مسکن

در این مقاله، سیستم خرید و فروش مسکن در قالب بستر اینترنتی تعریف شده و بخش‌های مختلفی را که برای عملیات

1. Run-time
2. Programming Language
3. Generic role
4. Domain-Specific role
5. Inheritance
6. Role binding

خرید و فروش لازم است، ارئه می‌دهد. در این سیستم، افراد با مراجعه به سایت می‌توانند اطلاعات معاملات صورت‌گرفته به‌صورت ماهانه را مشاهده کنند و پس از بررسی دربارهٔ خرید یا فروش مسکن تصمیم بگیرند. هنگام درخواست خرید، داشتن یک‌سوم مبلغ کف مشخص‌شده توسط وزارت مسکن و شهرسازی، بایستی در حساب کاربر باشد و هنگام درخواست فروش، ارائهٔ سند معتبر به کارشناس وزارت الزامی است (تمام مراحل به‌صورت الکترونیکی صورت می‌گیرد). هر درخواست خرید و فروش شامل تعیین متراژ و محل، سال ساخت و نحوهٔ انجام معامله از نظر قیمت است؛ یعنی حداکثر قیمت خرید یا حداقل قیمت فروش (از سوی خریدار و فروشنده) تعیین می‌شود. همچنین خریداران و فروشندگان می‌توانند در درخواست، شروطی را لحاظ کنند؛ برای مثال با توجه به قیمت نوسانی بازار مسکن، فروشنده می‌تواند یک ماه خاص را در درخواست خود برای نمایش اطلاعات مسکن به‌منظور سود بیشتر انتخاب کند (با توجه به اینکه در برخی از ماه‌های سال قیمت مسکن افزایش بیشتری دارد) و این موضوع، به نوعی استراتژی فروشنده محسوب می‌شود. از طرف دیگر، درصورتی‌که خریدار بخواهد بخشی از پول را به‌صورت اقساط پرداخت کند، می‌تواند اعلام کند. پس از انجام درخواست خرید و فروش از سوی خریدار و فروشنده، کارشناس وزارت به بررسی صحت و سقم درخواست‌های خریداران و فروشندگان می‌پردازد و درصورتی‌که اطلاعات کامل و دقیق باشند، آن‌ها را در سیستم وارد می‌کند، پس از ورود درخواست، سیستم با توجه به اولویت‌ها، معاملات مسکن را سامان می‌دهد و با توجه به محدودیت‌های اعمال شده توسط طرفین، خریداران و فروشندگان می‌توانند در طول ۴۸ ساعت آینده، لیستی از مواردی را که با توجه به شرایط درخواستی (حداقل و حداکثر قیمت و دیگر شرایط) در آن‌ها امکان معامله وجود دارد، مشاهده کنند. پس از مشاهدهٔ لیست، خریداران و فروشندگان می‌توانند براساس اولویت، مواردی را که مد نظرشان است، انتخاب و بر روی آن‌ها درخواست بدهند. در درخواست، اولویت معامله براساس درخواست‌هایی است که زودتر داده شود (هم در بین فروشندگان و هم در بین خریداران)، پس از انجام معامله، سیستم مسکن مدارک معامله را صادر می‌کند که بر این اساس مدارک شامل گواهی کارشناس وزارت، گواهی خرید، گواهی فروش و گواهینامهٔ موقت معامله است. پس از تهیهٔ مدارک، گواهی‌نامه‌های مربوط، به طرفین هر معامله تحویل می‌شود. در این سیستم کارمزد هر معامله براساس مصوبهٔ وزارت مسکن و شهرسازی بین طرفین تقسیم می‌شود.

۴. معرفی متدولوژی IR

IR یک متدولوژی توسعهٔ نرم‌افزاری عامل‌گراست که به‌صورت ترکیبی از دو متدولوژی RICA و INGENIAS و همچنین مدل‌های پیشنهادی قابلیت، برنامه‌ریز و دانش به‌دست آمده است. در تعیین مراحل فازهای متدولوژی IR سه ایدهٔ اصلی مورد استفاده قرار گرفته است که عبارت‌اند از: ۱. تصور و درک از عامل و همهٔ تصورات ذهنی (اهداف، برنامه‌ریزی‌ها و موارد مشابه) در مراحل فازهای توسعه نرم‌افزار از تحلیل اولیه تا طراحی مورد استفاده قرار گرفته است. ۲. به‌منظور درک کامل از محیط و تعامل میان نرم‌افزار و عامل‌ها، مدل دانش به فاز تحلیل در متدولوژی IR اضافه شده است. ۳. به‌منظور تعیین قابلیت‌های عامل‌ها به‌صورت جزئی و همچنین چگونگی اجرای این قابلیت‌ها به‌وسیلهٔ عامل‌ها، دو مدل قابلیت و برنامه‌ریز در فاز طراحی متدولوژی IR پیشنهاد شده است.

تحلیل و طراحی عامل‌گرا مجموعهٔ زیادی از مفاهیم را شامل می‌شود و این موضوع فهم همهٔ جنبه‌های مدل تحلیل و طراحی را از دیدگاه خاص مشکل می‌سازد. به همین دلیل در متدولوژی IR، تعدادی مدل که روی جنبه‌های مختلف تأکید دارند، تعریف شده است. این مدل‌ها جنبه‌های مختلفی را در برمی‌گیرند، اما به تنهایی کامل نیستند، بنابراین با کنار هم قرار دادن آن‌ها، یک دیدگاه کامل و قابل فهم درخصوص سیستم حاصل می‌شود. در انتخاب مدل‌های متدولوژی IR از یک چارچوب ارزیابی جامع که چهار ناحیهٔ عمدهٔ مهندسی نرم‌افزار عامل‌گرا شامل [۱۹-۲۲] مفاهیم و ادراک‌ها، زبان مدل‌سازی، فرایند و عمل‌گرایی را پوشش می‌دهد، استفاده شده است.

براساس این چارچوب و پارامترهای آن میزان پشتیبانی هریک از متدولوژی‌های مورد بررسی و همچنین مدل‌های پیشنهادی قابلیت، برنامه‌ریز و دانش از پارامترهای ارزیابی براساس نظر خبرگان مورد بررسی قرار گرفته است. بنابراین مدل‌های استفاده‌شده در متدولوژی IR، مفاهیم عامل‌گرایی را تا حد زیادی پوشش می‌دهد و این مدل‌ها هیچ‌گونه هم‌پوشانی با یکدیگر ندارند. در ادامه به‌منظور سنجش توانایی متدولوژی پیشنهادی، از آن در یک مطالعهٔ موردی که در بخش سوم تشریح گردید، استفاده شده است.

۱.۴. تحلیل سیستم خرید و فروش مسکن با استفاده از فازهای متدولوژی پیشنهادی

در این قسمت، روشی برای یکسان‌سازی دو متدولوژی RICA و INGENIAS به‌وسیلهٔ ترکیب نقاط قوت و نیز اجتناب از محدودیت‌های آن‌ها ارائه می‌شود. در حقیقت از برخی از قسمت‌های این متدولوژی‌ها، برای ایجاد متدولوژی جدید، با توجه به چارچوب ارائه‌شده در مرحلهٔ قبل استفاده می‌شود. استفاده از روش ترکیبی می‌تواند تأثیر مناسبی بر روی متدولوژی پیشنهادی به‌منظور پوشش حداکثری فعالیت‌های مهندسی نرم‌افزار عامل‌گرا داشته باشد و در توسعهٔ نسل بعدی متدولوژی‌های عامل‌گرا مؤثر باشد. در این مقاله، برای ادغام متدها از روش فرایند توسعه استفاده می‌شود و با توجه به اهمیت فازهای تحلیل و طراحی در ایجاد محصول نرم‌افزاری با کیفیت و قابلیت اطمینان بالا، بر روی این دو فاز تمرکز خواهیم کرد.

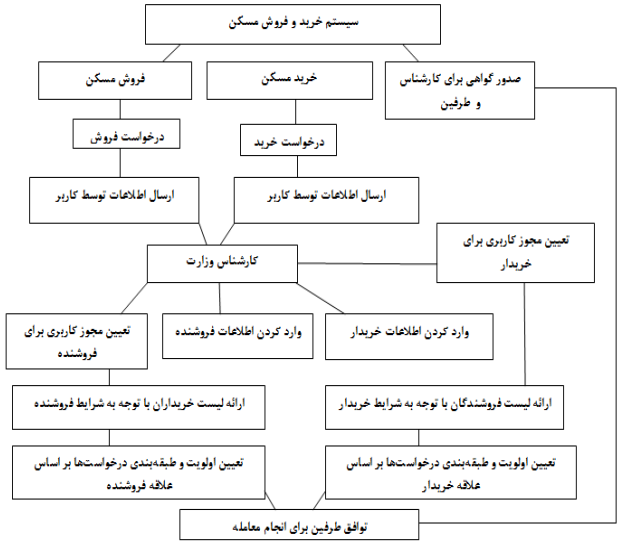
۱.۱.۴. فازهای متدولوژی پیشنهادی

۱.۱.۱.۴. فاز تحلیل

متدولوژی پیشنهادی در فاز تحلیل شامل مدل هدف و وظیفه، مدل دانش، مدل محیط و مدل نقش می‌باشد که این چهار مدل پشتیبانی قدرتمندی برای مشخص نمودن اهداف، وظیفه و دانش سیستم، به‌دست آوردن محیط و تعریف نقش‌های کلیدی در سیستم ارائه می‌دهند. این مدل‌ها آگاهی توسعه‌دهندگان از نیازمندی‌های سیستم را ارتقا می‌دهند و ورودی‌هایی برای مرحلهٔ بعدی یعنی طراحی فراهم می‌آورند (تحلیل و طراحی سیستم با توجه به فرضیات بخش سوم صورت می‌گیرد).

۱.۱.۱.۱.۴. مدل هدف

سازمان‌دهی و مشخص کردن اهداف، مرحلهٔ مهمی در استخراج نیازمندی‌هاست. در این مدل، اهداف کلی سیستم در سطح‌های مختلف مشخص می‌شود. شکل (۱) مدل هدف و وظیفهٔ سیستم خرید و فروش مسکن را نشان می‌دهد.



شکل (۱): مدل هدف و وظیفه سیستم خرید و فروش مسکن

همان‌طور که در شکل (۱) نمایش داده شده، سیستم خرید و فروش مسکن شامل سه بخش خرید مسکن، فروش مسکن و صدور گواهی است. در این سیستم، هر کدام از خریداران و فروشندگان بایستی در ابتدا درخواستی را مبنی بر خرید یا فروش بدهند (شرایط این درخواست در بخش سوم به تفضیل بیان شد). پس از انجام درخواست اطلاعات باید به‌صورت کامل و دقیق ارسال شود. پس از این مرحله کارشناس وزارت، صحت و سقم اطلاعات را بررسی می‌کند و در صورت نبودن مشکل، مجوز کاربری را تعیین کرده و اطلاعات مربوط را وارد می‌کند. پس از انجام این مرحله با توجه به اطلاعات و شرایط اعلام‌شده توسط خریداران و فروشندگان لیستی(لیست خریداران برای فروشندگان و لیست فروشندگان برای خریداران) در اختیار آنان قرار می‌گیرد که با مجوز کاربری می‌توانند از آن استفاده کنند و براساس آن لیست، اولویت‌ها و

علاقه‌مندی‌های هریک از خریداران و فروشندگان طبقه‌بندی می‌شود و در صورت توافق، سیستم گواهی را صادر می‌کند.

۲.۱.۱.۱.۴. مدل دانش

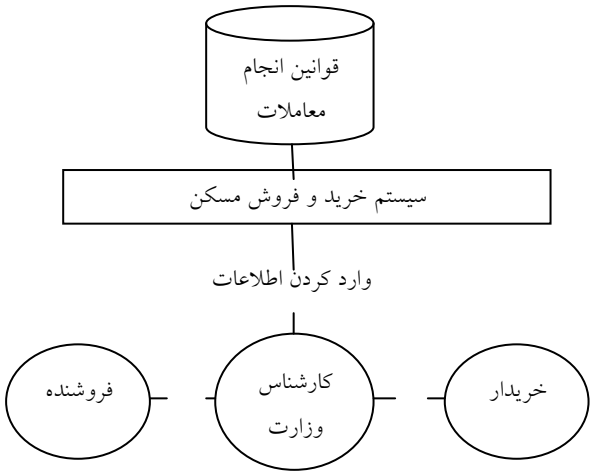
مدل دانش شامل قواعد، مراحل و محدودیت‌های سیستم است. قواعد مربوط به اصولی است که سیستم براساس آن تصمیم‌گیری را انجام می‌دهد. مراحل، انجام عملیات‌های مربوط به سیستم را مشخص می‌کنند و محدودیت‌ها نیز شامل مواردی است که ضروری است برای استفاده از سیستم در نظر گرفته شوند. جدول (۲) مدل دانش سیستم خرید و فروش مسکن را نشان می‌دهد.

جدول (۲): مدل دانش سیستم خرید و فروش مسکن

قواعد	-اگر قیمت خریدار برابر یا بیشتر قیمت فروشنده بود، اطلاعات مربوطه برای خریدار نمایش داده شود. -ارائه سند معتبر از ملک قابل فروش توسط فروشنده -اگر قیمت فروشنده برابر یا کمتر قیمت خریدار باشد، اطلاعات مربوط به خریدار برای فروشنده نمایش داده شود.
مراحل	-مرحله خرید مسکن: داشتن یک سوم مبلغ کف تعیین شده توسط وزارت مسکن و شهرسازی، ارائه درخواست خرید، گرفتن مجوز کاربری، بررسی لیست و طبقه‌بندی بر اساس اولویت. -مرحله فروش مسکن: رائه سند معتبر، ارائه درخواست فروش، گرفتن مجوز کاربری، بررسی لیست و طبقه‌بندی بر اساس اولویت.
محدودیت‌ها	-ارائه حساب بانکی توسط خریدار که مبلغ یک سوم کف تعیین شده وزارت مسکن و شهرسازی را دارا باشد. -ارائه سند مالکیت معتبر -فاصله زمانی ارائه لیست به خریداران و فروشندگان، ۴۸ ساعت بعد از وارد کردن اطلاعات توسط کارشناس وزارت می‌باشد.

۳.۱.۱.۱.۴. مدل محیط

مدل‌سازی محیط به‌منظور مشخص شدن محدودهٔ سیستم و همچنین شناخت بیشتر سیستم چندعامله صورت می‌گیرد. در همین راستا عوامل محیطی و نحوهٔ ارتباط آن‌ها با سیستم مشخص می‌شود. همچنین در این مرحله، مؤلفه‌های موجود در سیستم و وظایفی که هر کدام بر عهده دارند، مشخص می‌شود.. شکل (۲) مدل محیط در سیستم خرید و فروش مسکن را ارائه می‌دهد.



شکل (۲): مدل محیط خرید و فروش مسکن

۴.۱.۱.۱.۴. مدل نقش

در رویکرد عامل‌گرا، عامل به‌عنوان موجودیتی کلیدی در نظر گرفته می‌شود. بنابراین می‌توان گفت یکی از نیازهای اساسی متدولوژی عامل‌گرا، کمک به توسعه‌دهندگان برای مشخص کردن عامل‌های سیستم است. در متدولوژی پیشنهادی برای مشخص کردن عامل‌ها از تکنیک مدل نقش استفاده می‌شود، به این ترتیب که ابتدا نقش‌های موجود در سیستم به‌طور دقیق استخراج می‌شود و سپس براساس نقش‌های شناسایی شده در سیستم، عامل‌ها تعریف می‌شوند. در واقع می‌توان گفت که مدل نقش، رکن اصلی برای مشخص کردن عامل‌هاست، زیرا عامل‌ها به‌منظور انجام نقش‌ها، بایستی در سیستم به‌کار گرفته شوند. این مدل شامل اهداف، زیرنقش‌ها و مسؤلیت‌های نقش‌های موجود در سیستم است. نقش‌های سیستم خرید و فروش مسکن شامل جست‌وجوی اطلاعات، ذخیره و بازیابی اطلاعات، مجوز کاربری، طبقه‌بندی اطلاعات و به‌روزرسانی اطلاعات است که در ادامه در مورد آن‌ها بحث خواهیم کرد.

- **نقش جست‌وجوی اطلاعات**

این نقش به‌منظور اینکه کاربران (خریداران و فروشندگان) بتوانند براساس لیستی که در اختیار آن‌ها قرار می‌گیرد، عمل جست‌وجو را انجام دهند، تعریف می‌شود. براساس این نقش، کاربران می‌توانند با مشخص کردن اطلاعات پایه (کد ملی شخص مدّنظر که در لیست مشخص شده است) عمل جست‌وجوی جزئیات اطلاعات در مورد مسکن شخص مورد نظر را انجام دهند. در واقع با توجه به اینکه کاربران موظف هستند جزئیاتی را درخصوص مسکن از قبیل متراژ، سال ساخت و موارد مشابه را مشخص کنند، خریداران و فروشندگان می‌توانند با جست‌وجوی کد ملی شخص از لیست مورد نظر اطلاعات کامل را به‌صورت جزئی، در لیستی جداگانه مشاهده کنند. همچنین کاربران می‌توانند اطلاعات مربوط به اولویت‌های خود و دیگر موارد مدنظر خود را جست‌وجو کنند. جدول (۳) مدل نقش را برای نقش جست‌وجوی اطلاعات ارائه می‌دهد.

جدول (۳): مدل نقش برای نقش جست‌وجوی اطلاعات

نام نقش: جست‌وجو اطلاعات
هدف نقش: ارائه اطلاعات بر اساس کد ملی وارد شده از سوی کاربر
زیرنقش: خواندن اطلاعات محدودیت (کد ملی وارد شده) که بر اساس آن می‌بایست عملیات جست‌وجو انجام شود.
مسؤلیت: ارائه لیست بر اساس کد ملی مورد نظر

- **نقش ذخیره و بازیابی اطلاعات**

این نقش به‌منظور ذخیره و بازیابی اطلاعات در سه قالب ذخیره اطلاعات براساس اولویت‌های علایق کاربران، ذخیرهٔ اطلاعات سابق که شامل عملیات‌هایی است که کاربر تاکنون در سیستم انجام داده است و ذخیرهٔ اطلاعات کاربری تعریف می‌شود. جدول (۴) مدل نقش برای نقش ذخیره و بازیابی اطلاعات را نشان می‌دهد.

جدول (۴): مدل نقش برای نقش ذخیره و بازیابی اطلاعات

نام نقش: ذخیره و بازیابی اطلاعات
هدف نقش: ذخیره اطلاعات اولویت، سابقه و کاربری
زیرنقش: خواندن اطلاعات وارد شده توسط کاربر
مسؤلیت: ایجاد لیست اطلاعات برای کاربر

- **نقش مجوز کاربری**

این نقش به‌منظور مجوز کاربری، برای کارشناسان، خریداران و فروشندگان که سعی در ورود به سیستم دارند مطرح می‌باشد. در واقع نقش مجوز کاربر، تأیید اعتبار اطلاعات کاربری را بر عهده دارد. جدول (۵) مدل نقش برای نقش مجوز کاربری را نشان می‌دهد.

جدول (۵): مدل نقش برای نقش مجوز کاربری

نام نقش: مجوز کاربری
هدف نقش: تأیید اطلاعات کاربر، ایجاد سطح دسترسی
زیرنقش: خواندن اطلاعات وارد شده توسط کاربر و جست‌وجو در پایگاه داده برای تست اعتبار
مسؤلیت: حفظ امنیت سیستم

- **نقش طبقه‌بندی اطلاعات**

نقش طبقه‌بندی به‌منظور طبقه‌بندی علاقه‌مندی‌های خریداران و فروشندگان تعریف می‌شود. این نقش دو لیست سطح یک و سطح دو ایجاد می‌کند و بر این اساس، علایق کاربر در لیست سطح یک و دیگر موارد در لیست سطح دو قرار می‌گیرد. جدول (۶) مدل نقش برای نقش طبقه‌بندی را نشان می‌دهد.

جدول (۶): مدل نقش برای نقش طبقه‌بندی

نام نقش: طبقه‌بندی اطلاعات
هدف نقش: طبقه‌بندی اطلاعات بر اساس اولویت علایق کاربر در سطح یک و دو
زیرنقش: خواندن اولویت‌های کاربر
مسؤلیت: ایجاد طبقه‌بندی در اولویت‌های کاربر در دو سطح یک و دو

- **نقش به‌روزرسانی اطلاعات**

نقش به‌روزرسانی اطلاعات، به‌منظور ارائهٔ لیست موارد جدید اضافه شده برای خرید و فروش مسکن و همچنین ارائهٔ اطلاعات معاملاتی که انجام شده و به‌روزرسانی لیست اولویت‌ها و علایق کاربران تعریف می‌شود. این نقش، اطلاعات خرید و فروش را از پایگاه داده سیستم خرید و فروش مسکن و اطلاعات معاملات را از پایگاه داده سیستم وزارت مسکن و شهرسازی دریافت می‌کند و در قالب لیست در اختیار کاربران قرار می‌دهد. جدول (۷) مدل نقش برای نقش به‌روزرسانی اطلاعات را ارائه می‌دهد (در این قسمت فرض می‌شود که تنها اجازه به‌روزرسانی در موارد ذکرشده در سیستم داده شده است).

جدول (۷): مدل نقش برای نقش به‌روزرسانی اطلاعات

نام نقش: به‌روزرسانی اطلاعات
هدف نقش: به‌روزرسانی اطلاعات و نمایش وضعیت معاملات انجام شده، به‌روزرسانی اطلاعات علایق اولویت‌ها
زیرنقش: خواندن اطلاعات از پایگاه داده سیستم خرید و فروش مسکن و وزارت مسکن و شهرسازی و علایق کاربر
مسؤلیت: ارائه لیست مسکن‌های اضافه شده برای خرید و فروش معاملات انجام شده، اعمال تغییرات در لیست علایق

در مرحلهٔ تحلیل، براساس مدل‌های هدف و وظیفه، دانش، محیط و نقش، نیازمندی‌های سیستم و همچنین قواعد و محدودیت‌های آن استخراج شد. در ادامه دربارهٔ مراحل فاز طراحی در متدولوژی پیشنهادی بحث خواهیم کرد.

۲.۱.۱.۴. فاز طراحی

متدولوژی پیشنهادی در فاز طراحی شامل مدل عامل، مدل تعامل، مدل قابلیت و مدل برنامه‌ریز است. در مدل عامل، نقش‌ها به هر عامل نگاشت می‌شوند، مدل تعامل مشخص‌کنندهٔ روابط بین عامل‌ها به‌منظور انجام نقش‌هاست. مدل قابلیت و برنامه‌ریز نیز به‌منظور مدل کردن قابلیت‌های عامل و برنامه‌ریزی برای چگونگی انجام این قابلیت‌ها به‌کار می‌روند.

۱.۲.۱.۱.۴. مدل عامل

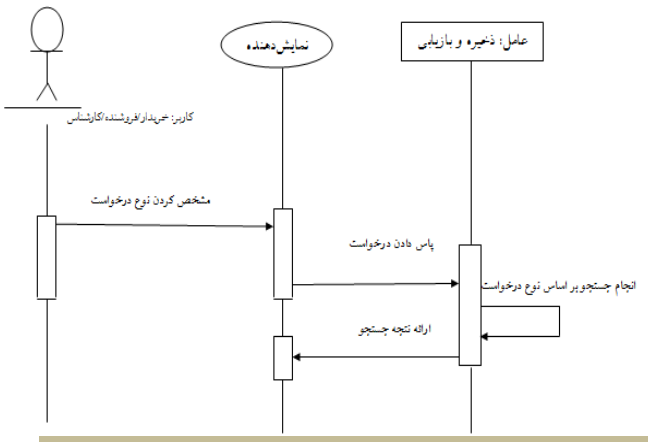
مدل عامل مکمل مدل نقش در فاز تحلیل است. در این مدل، براساس نقش‌های تعیین‌شده در مدل نقش، عامل تعریف می‌گردد و پس از آن هر نقش به یک عامل نگاشت می‌شود. سیستم خرید و فروش مسکن شامل پنج عامل است که در جدول (۸) نقش‌ها به این عامل‌ها نگاشت شده‌اند.

جدول (۸): تخصیص نقش‌ها به عامل‌ها

عملیات جست‌وجو	عامل جست‌وجو
عملیات ذخیره و بازیابی	عامل ذخیره و بازیابی
عملیات تأیید مجوز کاربری	عامل مجوز کاربری
عملیات طبقه‌بندی علایق کاربری	عامل طبقه‌بندی
عملیات به‌روزرسانی اطلاعات	عامل به‌روزرسانی

۲.۲.۱.۱.۴. مدل تعامل

مدل تعامل مشخص‌کنندهٔ روابط بین عامل‌ها برای انجام نقش‌ها می‌باشد. به‌عبارت دیگر نحوهٔ انجام گرفتن نقش‌ها به‌وسیلهٔ عامل‌ها با استفاده از مدل تعامل، مدل می‌شود. در مدل نقش پنج نقش برای سیستم تعیین شد. اولین نقش، نقش جست‌وجوی اطلاعات می‌باشد که مدل تعامل این نقش در شکل (۳) نمایش داده شده است.

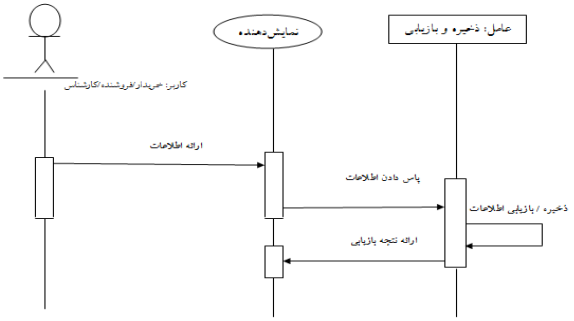


شکل (۳): مدل تعامل نقش جست‌وجو اطلاعات

براساس شکل (۳)، کاربر (خریدار/ فروشنده/ کارشناس) ابتدا نوع درخواست خود را مشخص می‌کند، سپس این درخواست در اختیار عامل جست‌وجو قرار می‌گیرد و این

عامل، جست‌وجو را براساس نوع درخواست از پایگاه دادهٔ سیستم انجام و نتیجه را بازگشت می‌دهد.

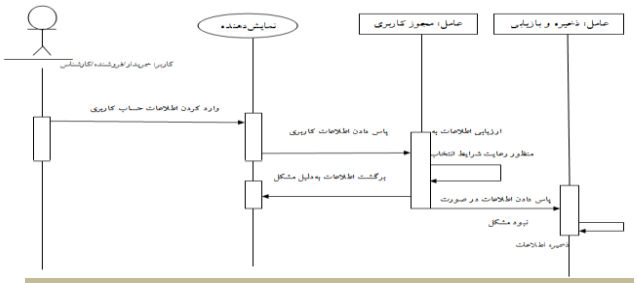
یکی دیگر از نقش‌هایی که در مدل نقش تعیین شد، نقش ذخیره و بازیابی است که شکل (۴) نمودار تعامل این نقش را نشان می‌دهد.



شکل (۴): مدل تعامل نقش ذخیره و بازیابی

براساس شکل (۴)، کاربر (خریدار/ فروشنده/ کارشناس) ابتدا بایستی اطلاعاتی را که می‌خواهد ذخیره و بازیابی شوند، ارئه کند، پس از آن این اطلاعات در اختیار عامل ذخیره و بازیابی قرار می‌گیرد و این عامل، عمل ذخیره و بازیابی را براساس نوع اطلاعات انجام می‌دهد و درصورتی‌که عملیات بازیابی باشد، نتیجه را بازگشت می‌دهد.

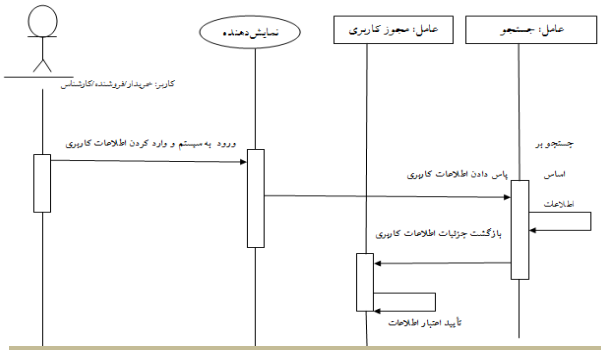
در ادامه، دربارهٔ مدل‌های تعامل نقش مجوز کاربری بحث خواهیم کرد. مدل تعامل نقش مجوز کاربری از دو حیطه، ذخیرهٔ اطلاعات کاربری و تأیید اعتبار اطلاعات کاربری مورد بررسی است. از نظر ذخیرهٔ اطلاعات کاریری، کاربر ابتدا اطلاعات حساب کاربری خود را وارد می‌کند و این اطلاعات در اختیار عامل مجوز کاربری قرار می‌گیرد. این عامل، ارزیابی اطلاعات وارد شده را (از نظر اینکه کلمه و رمز عبور تکراری نباشد) انجام می‌دهد، سپس درصورتی‌که این قاعده رعایت نشده باشد، پیغام خطا (با توجه به نوع مشکل) و برگشت اطلاعات صورت می‌گیرد و در صورت نبود مشکل، اطلاعات را به عامل ذخیره و بازیابی پاس می‌دهد و عامل ذخیره و بازیابی، اطلاعات را ذخیره می‌کند. شکل (۵) مدل تعامل نقش مجوز کاربری در حیطهٔ ذخیره را نشان می‌دهد.



شکل (۵): مدل تعامل نقش مجوز کاربری در حیطهٔ ذخیره

البته باید توجه داشت که کاربر (خریدار و فروشنده منظور است و شامل کارشناس نمی‌شود) فقط در زمانی که می‌خواهد درخواست خرید یا فروش بدهد، لازم است که در سیستم ثبت‌نام کند. بعد از اینکه درخواست انجام شد، درصورتی‌که در درخواست و مدارک مشکلی نداشته باشند، کارشناس رمز و کلمهٔ عبور جداگانه به کاربران اختصاص می‌دهد و دیگر نیازی به رمز و کلمه عبور قبلی نیست و کاربر تنها با استفاده از رمز و کلمهٔ عبور کارشناس می‌تواند از سیستم استفاده کند.

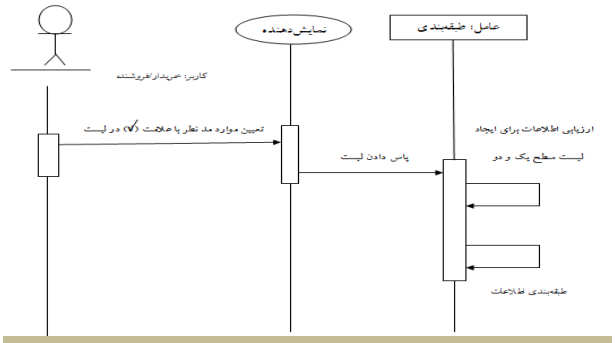
در حیطهٔ تأیید اعتبار اطلاعات کاربری، کاربر به‌منظور ورود به سیستم، اطلاعات کاربری خود را وارد می‌کند و این اطلاعات در اختیار عامل جست‌وجو قرار می‌گیرد و عامل جست‌وجو براساس اطلاعات وارده، عمل جست‌وجو در پایگاه داده سیستم را انجام می‌دهد. شکل (۶) مدل تعامل نقش مجوز کاربری در حیطه تأیید اعتبار را نشان می‌دهد.



شکل (۶): مدل تعامل نقش مجوز کاربری در حیطهٔ تأیید اعتبار

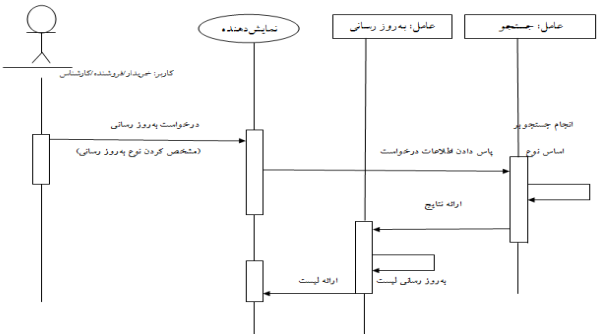
نقش دیگری که در مدل نقش تعیین شد، نقش طبقه‌بندی اطلاعات است. در مدل تعامل این نقش، ابتدا کاربر در لیست ارائه‌شده از مسکن‌ها، مواردی را که مد نظرش است، مشخص

می‌کند، پس از آن این اطلاعات در اختیار عامل طبقه‌بندی قرار می‌گیرد و این عامل پس از تفکیک اطلاعات به دو سطح یک (علاقه‌مندی‌های کاربر) و سطح دو (دیگر موارد)، اطلاعات را طبقه‌بندی می‌کند. شکل (۷) مدل تعامل نقش طبقه‌بندی اطلاعات را ارائه می‌دهد.



شکل (۷): مدل تعامل نقش طبقه‌بندی اطلاعات

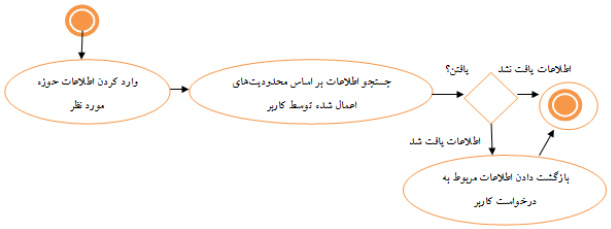
دربارهٔ نقش به‌روزرسانی، کاربر ابتدا باید نوع به‌روزرسانی (به‌روزرسانی اطلاعات مسکن، به‌روزرسانی علایق کاربر) را مشخص کند (فرض بر این است که کاربر اجازه به‌روزرسانی موارد دیگر را ندارد). پس از مشخص شدن نوع به‌روزرسانی، درخواست به عامل جست‌وجو داده می‌شود و این عامل براساس نوع به‌روزرسانی، اطلاعات را جست‌وجو می‌کند و نتایج را در اختیار عامل به‌روزرسانی می‌گذارد و عامل به‌روزرسانی عملیات به‌روزرسانی لیست را انجام می‌دهد و نتایج را در اختیار کاربر قرار می‌دهد. شکل (۸) مدل تعامل نقش به‌روزرسانی را ارائه می‌دهد.



شکل (۸): مدل تعامل نقش به‌روزرسانی

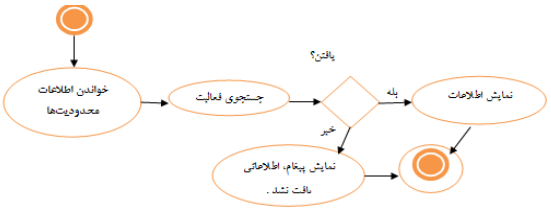
۳.۲.۱.۱.۴. مدل قابلیت و برنامه‌ریز

این مدل، قابلیت‌های عامل‌ها را مدل می‌کند. همچنین مراحل اجرای قابلیت‌ها، به‌وسیلهٔ این مدل برنامه‌ریزی می‌شود. به‌عبارت دیگر، در این مدل محدودهٔ وظایف عامل‌ها و چگونگی انجام وظایف توسط عامل‌ها مدل می‌شود. در بخش مدل عامل، پنج عامل تعیین شد. اولین عامل، عامل جست‌وجوی اطلاعات می‌باشد که مدل قابلیت این عامل در شکل (۹) ارائه شده است.



شکل (۹): مدل قابلیت عامل جست‌وجو

همچنین مدل برنامه‌ریز این عامل در شکل (۱۰) نشان داده شده است.



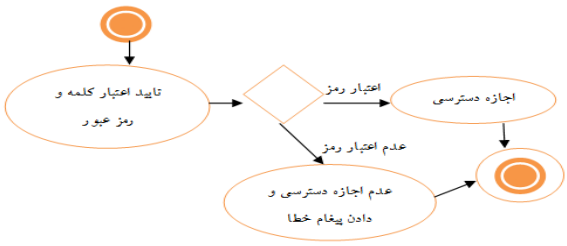
شکل (۱۰): مدل برنامه‌ریز عامل جست‌وجو

براساس شکل (۹) قابلیت عامل جست‌وجو با توجه به اطلاعات ارائه‌شده توسط کاربر مدل می‌شود، درحالی‌که شکل (۱۰) نحوهٔ انجام این قابلیت، توسط عامل جست‌وجو را مدل می‌کند.

عامل دیگری که در مدل عامل شناسایی شد، عامل ذخیره و بازیابی است که نیازی به مدل شدن توسط مدل‌های قابلیت و برنامه‌ریزی ندارد، زیرا استثنائات چندانی ندارد و به‌خوبی در مدل تعامل، قابلیت آن مدل شده است.

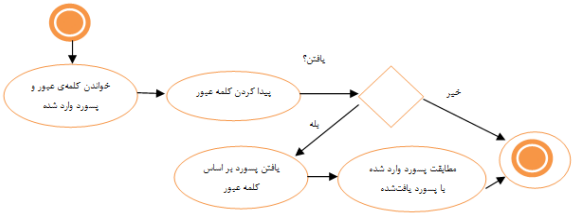
در ادامه دربارهٔ مدل قابلیت و برنامه‌ریزی عامل مجوز کاربر بحث خواهیم کرد. همان‌طور که قبلاً اشاره شد، عامل

مجوز کاربری وظیفه تأیید اعتبار کلمهٔ عبور و رمز عبور را بر عهده دارد. بر این اساس، اگر کاربر کلمه و رمز عبور معتبر وارد کند، اجازهٔ دسترسی به سیستم داده می‌شود، در غیر این صورت اجازهٔ دسترسی به کاربر داده نمی‌شود و پیغام خطا نمایش داده می‌شود. شکل (۱۱) مدل قابلیت عامل مجوز کاربری را نشان می‌دهد.



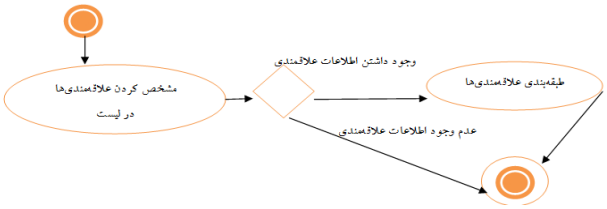
شکل (۱۱): مدل قابلیت عامل مجوز کاربری

در مدل برنامه‌ریزی نیز عامل مجوز کاربر، ابتدا کلمه و رمز عبور واردشده را می‌خواند و براساس کلمهٔ عبور جست‌وجو را انجام می‌دهد. اگر کلمهٔ عبور واردشده یافت نشود که عملیات خاتمه می‌یابد، در غیر این صورت پس از یافتن اطلاعات، رمز عبور واردشده را با رمز عبور پایگاه داده تطبیق می‌دهد. شکل (۱۲) مدل برنامه‌ریز عامل مجوز کاربری را ارائه می‌دهد.



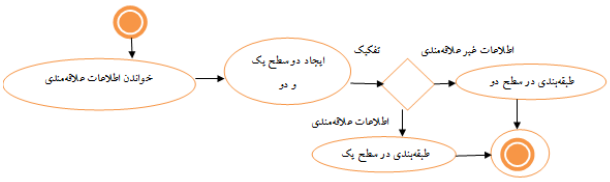
شکل (۱۲): مدل برنامه‌ریز عامل مجوز کاربری

عامل دیگری که در مدل عامل تعیین شد، عامل طبقه‌بندی اطلاعات است. در مدل قابلیت این عامل، در ابتدا کاربر علاقه‌مندی‌های خود را براساس لیست ارائه‌شده مشخص می‌کند. اگر موردی توسط کاربر انتخاب نشود، طبقه‌بندی توسط عامل صورت نمی‌گیرد و در صورت انتخاب، عامل طبقه‌بندی، علاقه‌مندی‌ها را براساس اولویت طبقه‌بندی می‌کند. در شکل (۱۳) مدل قابلیت، عامل طبقه‌بندی نشان داده شده است.



شکل (۱۳): مدل قابلیت عامل طبقه‌بندی

در مدل برنامه‌ریزی، عامل طبقه‌بندی ابتدا اطلاعات علاقه‌مندی مشخص‌شده از سوی کاربر را دریافت می‌کند و دو سطح یک و دو را ایجاد می‌کند، سپس به تفکیک اطلاعات براساس اولویت علاقمندی و غیرعلاقه‌مندی می‌پردازد و اطلاعات علاقه‌مندی را در سطح یک و اطلاعات غیرعلاقه‌مندی را در سطح دو طبقه‌بندی می‌کند. شکل (۱۴) مدل برنامه‌ریز عامل طبقه‌بندی را ارائه می‌دهد.



شکل (۱۴): مدل برنامه‌ریز عامل طبقه‌بندی

با توجه به اینکه قابلیت عامل به‌روزرسانی نیز مانند عامل ذخیره و بازیابی، به‌خوبی توسط مدل تعامل، مدل شده است، نیازی به استفاده از مدل‌های قابلیت و برنامه‌ریز برای این عامل نیست.

در ادامه، ارزیابی متدولوژی پیشنهادی براساس معیارهای مفاهیم و ادراک‌ها و عمل‌گرایی ارائه شده است.

۵. ارزیابی متدولوژی ترکیبی (IR)

در این بخش، ارزیابی متدولوژی IR و مقایسهٔ این متدولوژی با متدولوژی‌های RICA و INGENIAS ارائه شده است. متدولوژی ترکیبی IR در فاز تحلیل، شامل مدل‌های هدف و

وظیفه، دانش، محیط و نقش و در فاز طراحی شامل مدل‌های عامل، تعامل، قابلیت و برنامه‌ریز است. برای ارزیابی از روش پرسشنامه‌ای استفاده شده است. پرسش‌ها توسط ۱۱ نفر خبره و کارشناس در زمینهٔ مهندسی نرم‌افزار پاسخ داده شده است. به‌منظور رفع مشکل ناهمگرایی موجود در نظریات از روش دلفی‌فازی [۲۳] استفاده شده و نتایج پس از نرمال شدن، در جدول‌های (۹) و (۱۰) قرار داده شده است. ارزیابی متدولوژی براساس معیارهای مفاهیم و ادراک‌ها و عمل‌گرایی انجام گرفته است، زیرا این دو معیار از مهم‌ترین معیارهای ارزیابی متدولوژی‌های عامل‌گرا هستند. در ادامه دربارهٔ هریک از پارامترهای این معیارها بحث خواهیم کرد.

۱.۵. مفاهیم و ادراک‌ها

این معیار به مفاهیم و خصوصیات عامل‌ها و سیستم‌های عامل‌گرا اشاره می‌کند. از مهم‌ترین پارامترهای این معیار عبارت‌اند از [۱۹-۲۲]:

۱. خودمختاری: این پارامتر به عامل، قدرت تصمیم‌گیری مستقل انجام کار در حالت‌های داخلی سیستم، بدون ناظر و کنترل خارجی را می‌دهد.

۲. هدف‌گرا بودن: این پارامتر به جای استفاده از پاسخ به رویدادها از کارهای ابتکاری برای نمایش رفتار هدف‌گرا استفاده می‌کند؛ یعنی سیستم‌های عامل‌گرا و خود عامل‌ها، اهدافی دارند که عامل‌ها رسیدن به آن‌ها را دنبال می‌کنند.

۳. واکنش‌دار بودن[1]: قابلیت ادراک محیط و پاسخ‌گویی به تغییراتی که در آن اتفاق می‌افتد.

۴. هم‌اجرایی: قابلیت عامل برای سروکار داشتن با چندین هدف یا حادثه در زمان مشابه.

۵. همکاری: این معیار مدل‌هایی را که توسط متدولوژی حمایت می‌شود، بیان می‌کند.

۶. پروتکل: این معیار سطوح پشتیبانی برای تعریف مذاکرات مجاز بر حسب رشتهٔ معتبر از پیام‌های مبادله‌شده میان دو عامل را بررسی می‌کند.

۷. عامل‌گرایی: میزانی که متدولوژی مبتنی بر عامل عمل می‌کند.

۲.۵. عمل‌گرایی

این معیار به جنبه‌های عملی گسترش و استفاده از متدولوژی اشاره دارد. مهم‌ترین پارامترهای این معیار عبارت‌اند از [۱۹-۲۲]:

۱. هزینه: انواع متفاوتی از هزینه‌ها مرتبط با اتخاذ متدولوژی است.

۲. قابلیت کاربرد دامنه: این پارامتر برای دامنهٔ کاربردهایی است که متدولوژی برای آن‌ها مورد استفاده قرار می‌گیرد.

۳. مقیاس‌پذیری: این پارامتر را می‌توان با یک سؤال توضیح داد و آن این است که آیا متدولوژی یا زیرمجموعه‌ای از آن می‌تواند برای مدیریت اندازه‌های مختلف کاربردها به کار روند.

۴. توزیع‌پذیری: این پارامتر اندازه‌گیری برای متدولوژی‌هایی است که به‌منظور طراحی سیستم‌های توزیع‌شده به‌کار می‌روند.

۳.۵. نتایج ارزیابی بخش مفاهیم و ادراک‌ها

در جدول (۹)، مقایسهٔ بین متدولوژی‌ها در بخش مفاهیم و ادراک‌ها ارائه شده است. در این جدول، عدد (۱) نشان می‌دهد آن پارامتر به‌صورت ضعیف در متدولوژی برآورده می‌شود. عدد (۲) نشان می‌دهد آن پارامتر تا حدی در متدولوژی برآورده می‌شود و عدد (۳) نشان‌دهندهٔ آن است که متدولوژی آن پارامتر را به‌طور کامل برآورده می‌کند (برای همگرا شدن نظریات خبرگان از روش دلفی‌فازی استفاده شده و پس از استفاده از این روش، نتایج نظریات براساس اعداد مشخص شده است).

با توجه به جدول (۹)، نتایج ارزیابی هریک از پارامترها در ادامه مورد بحث قرار می‌گیرد.

1. Reactivity

۱. خودمختاری: خودمختاری به‌عنوان یکی از خصوصیات کلیدی عامل‌ها مورد نظر است. این ویژگی، عامل‌ها را از اشیا متمایز می‌کند. براساس ارزیابی انجام‌شده، هر سه متدولوژی عامل‌گرا این ویژگی را دارند. سطوح پشتیبانی برای پارامتر خودمختاری برای همهٔ آن‌ها از متوسط تا خوب است. متدولوژی IR به‌دلیل اضافه شدن مدل برنامه‌ریز نسبت به دو متدولوژی دیگر، پشتیبانی مناسب‌تری برای این پارامتر ارائه می‌دهد.

۲. هدف‌گرا بودن: از دیدگاه خبرگان متدولوژی IR پشتیبانی مناسبی را از پارامتر هدف‌گرا بودن ارائه می‌دهد، زیرا این متدولوژی، ابتدا اهداف را به‌دست می‌آورد و سپس با استفاده از مدل برنامه‌ریز، برای دستیابی به اهداف برنامه‌ریزی می‌کند، درحالی‌که متدولوژی INGENIAS تنها اهداف را استخراج می‌کند و متدولوژی RICA هیچ مدلی را برای به‌دست آوردن اهداف ارائه نمی‌دهد.

۳. واکنش‌دار بودن: از دیدگاه خبرگان این پارامتر، ارتباط تنگاتنگی با پارامتر هدف‌گرا بودن دارد، لذا نتایج این پارامتر همانند پارامتر هدف‌گرا بودن می‌باشد.

۴. هم‌اجرایی: پشتیبانی از هم‌اجرایی میان دو متدولوژی INGENIAS و IR تا حدودی صورت می‌گیرد، زیرا در هر دو متدولوژی، در یک نقش منفرد (در مدل نقش)، اعمال می‌توانند به‌طور هم‌اجرا تعریف شوند.

۵. همکاری: همکاری شامل کار تیمی و برنامه‌ریزی چندعامله است. از دیدگاه خبرگان این پارامتر در هر سه متدولوژی پشتیبانی ضعیفی دارد و هیچ‌یک از متدولوژی‌ها مدلی را برای ایجاد همکاری ارائه نکرده‌اند.

۶. پروتکل: در هیچ‌یک از سه متدولوژی، مدلی به‌طور مجزا بدون استفاده از AUML برای این پارامتر تعریف نشده است، ولی تعامل‌های میان عامل‌ها در سطوح بالا توسط هر سه متدولوژی پشتیبانی می‌شود.

جدول (۹): مقایسهٔ متدولوژی‌ها براساس معیار مفاهیم و ادراک‌ها

متدولوژی پارامتر	RICA	INGENIAS	IR
خودمختاری	۲	۲	۳
هدف‌گرا بودن	۱	۲	۳
واکنش‌دار بودن	۱	۲	۲
هم‌اجرایی	۱	۱	۱
پروتکل	۲	۲	۲

مدل‌های اضافه‌شده به متدولوژی IR، به‌صورت ترکیبی از متدولوژی‌های RICA و INGENIAS و همچنین مدل‌هایی که به‌منظور افزایش کارایی متدولوژی IR به آن اضافه شد، باعث شده که متدولوژی توسعه‌یافته در اکثر پارامترهای ارزیابی قابلیت‌های بهتری نسبت به دو متدولوژی مرجع داشته باشد.

۴.۵. نتایج ارزیابی بخش عمل‌گرایی

در جدول (۱۰)، مقایسهٔ بین متدولوژی‌ها در بخش عمل‌گرایی ارائه شده است. در این جدول نیز اعداد به‌صورت جدول (۹) در نظر گرفته می‌شوند. همچنین علامت (؟) در این جدول نشان‌دهندهٔ آن است که هنوز نمی‌توان راجع به پارامتر اظهارنظر قطعی کرد. در ادامه نتایج ارزیابی برای هر کدام از پارامترها مورد بررسی قرار می‌گیرد.

۱. هزینه و رهنمودهای کیفیت: به‌علت عدم تکامل متدولوژی‌های عامل‌گرا، موضوعات مرتبط با تخمین هزینه یا تضمین کیفیت، هنوز در متدولوژی‌ها نهادینه نشده است و خبرگان معتقدند که بایستی به تجربیات مهندسان نرم‌افزار در این موارد تکیه کرد. همچنین آن‌ها اعتقاد دارند که حداقل متدولوژی‌ها باید در بیش از صد پروژه مورد استفاده قرار گیرند تا بتوان راجع به این پارامترها اظهار نظر کرد.

۲. قابلیت کاربرد دامنه: چندین محدودیت دامنه در تکنیک‌ها و مدل‌های متدولوژی‌های RICA و INGENIAS

به‌دلیل عدم مدل کردن قابلیت‌های عامل‌ها و همچنین عدم برنامه‌ریزی برای استفاده از قابلیت‌ها وجود دارد. برای مثال استفاده از این متدولوژی‌ها برای سیستم‌هایی که امکان برخورد در سیستم وجود دارد، مناسب نیستند. این موضوع در حالی است که متدولوژی IR با توجه به داشتن مدل‌های قابلیت و برنامه‌ریزی در اکثر پروژه‌های مبتنی بر عامل می‌تواند مورد استفاده قرار گیرد.

۳. مقیاس‌پذیری: از دیدگاه خبرگان متدولوژی RICA پشتیبانی ضعیفی از مدیریت اندازه‌های مختلف پروژه ارائه می‌دهد و این متدولوژی تنها می‌تواند برای پروژه‌های کوچک مفید واقع شود. این در حالی است که از دیدگاه آن‌ها متدولوژی INGENIAS تا حدی می‌تواند اندازه‌های مختلف پروژه را پشتیبانی کند. خبرگان دربارهٔ متدولوژی IR معتقدند که هنوز نمی‌توان اظهار نظر قطعی کرد. البته اعتقاد دارند که با توجه به اینکه این متدولوژی بیشتر مدل‌ها را پوشش می‌دهد، می‌تواند در اندازه‌های مختلف پروژه به‌کار رود، ولی برای اظهار نظر قطعی بایستی در پروژه‌های بیشتری مورد استفاده قرار گیرد.

۴. توزیع‌پذیری: درخصوص پارامتر توزیع‌پذیری، خبرگان اعتقاد دارند که هر سه متدولوژی به‌طور ضمنی از توزیع‌پذیری حمایت می‌کنند. از دیدگاه آن‌ها این سطوح خاص، ناشی از طبیعت سیستم‌های مبتنی بر عامل است. اما با توجه به اینکه هیچ‌یک از سه متدولوژی مدل استقرار را پوشش نمی‌دهند، نمی‌توانند پشتیبانی مناسبی از این پارامتر داشته باشند، زیرا محل استقرار عامل‌ها در شبکه توسط این سه متدولوژی مدل نمی‌شود.

جدول (۱۰): مقایسه متدولوژی‌ها براساس معیار عمل‌گرایی

متدولوژی پارامتر	RICA	INGENIAS	IR
هزینه	؟	؟	؟
قابلیت کاربرد دامنه	۱	۲	۳
مقیاس‌پذیری	۱	۲	؟
توزیع‌شدگی	۱	۱	۱
رهنمودهای کیفیت	؟	؟	؟

۶. نتیجه‌گیری

در این مقاله، یک متدولوژی ترکیبی توسعه‌یافته برای تحلیل و طراحی سیستم‌های مبتنی بر عامل معرفی شد. در این متدولوژی با استفاده از ترکیب نقاط قوت فازهای متدولوژی‌های RICA و INGENIAS و همچنین اضافه کردن مدل‌های قابلیت، برنامه‌ریز و دانش، امکان استفاده از تکنیک‌های سطح بالا به‌منظور مدیریت پیچیدگی مسائل، فراهم شده است. استفاده از راهکار ترکیبی در متدولوژی پیشنهادی باعث شد که دو هدف عمدهٔ استفاده از استانداردهای کاری و تعریف مجدد بلوک‌های اصلی برآورده شود. متدولوژی پیشنهادی، در قالب پارامترهای ارزیابی در مهندسی مبتنی بر عامل، دارای پشتیبانی مناسب‌تری از پارامترهای خودمختاری، هدف‌گرا بودن، واکنش‌دار بودن و قابلیت کاربرد دامنه، نسبت به دو متدولوژی دیگر است.

کارهای آینده در زمینهٔ مهندسی نرم‌افزار عامل‌گرا می‌تواند در دو بخش متمرکز شود، بخش اول ارزیابی متدولوژی‌های توسعه‌یافته با شیوه‌ای تجربی یا مقایسه‌ای است و بخش دوم توسعهٔ متدولوژی‌های عامل‌گرا با استفاده از راهکار ترکیبی و روش ادغام متدهاست. این تحقیقات می‌تواند زمینه را برای معرفی نسل بعدی متدولوژی‌های عامل‌گرا فراهم آورد.

مراجع

[1] Zambonelli, F., Jennings, N. R., & Wooldridge M, "Multi-Agent Systems as Computational Organizations: the Gaia methodology", Journal of Autonomous Agents and Multi-Agent Systems, 9(3), 2005, 136-171.

[2] Weiß, G., "Agent Orientation in Software Engineering", the Knowledge Engineering Review, 16(4), 2001, 349-373

[3] Tveit, A., "A survey of Agent-Oriented Software Engineering", In Norwegian University of Science and Technology, 2001, 104-125.

[4] Garcia-Ojeda, J. C. & DeLoach Robby, S. A., "Agent Tool Process Editor: Supporting the Design of Tailored Agent-Based Processes", In Procedings of the ACM Symposium on Applied Compting, ACM, 2009, 707-714.

[5] DiLeo, J., Jacobs, T., & DeLoach, S. A. "Integrating Ontologies into Multiagent Systems Engineering", 4th International Bi-conference Workshop on Agent-Oriented Information Systems, 2002, 23-34.

[6] DeLoach, S. A., "O-MaSE: An Extensible Methodology for Multi-agent Systems", In Agent-Oriented Software Engineering, 4(3), 2014, 173-191.

[7] DeLoach, S. A., & Garcia-Ojeda, J. C., "The O-MaSE Methodology", In Handbook on Agent-Oriented Design Processes, 2014, 253-285.

[8] Juan, T., Pearce, A., & Sterling, L., "ROADMAP: Extending the Gaia Methodology for Complex Open Systems", 1th International Joint Conference on Autonomous Agents & Multi-Agent Systems, 3002, 3-10.

[9] Zambonelli, F., Jennings, N. R., Omicini, A., & Wooldridge, M. J., "Agent-oriented Software Engineering for Internet Applications", In Coordination of Internet Agents: Models, Technologies and Applications, 2001, 326-346.

[10] García-Ojeda, J. C., Arenas, A. E., & Alcázar, J. P., "Paving the Way for Implementing Multiagent Systems: Integrating Gaia with Agent-UML. 6th International Workshop Agent-Oriented Software Engineering", Lecture Notes in Computer Science 3950, Springer-Verlag, 2005, 179-189.

[11] Gonzalez-Palacios, J., & Luck, M., "Extending Gaia with Agent Design and Iterative Development", 8th International Workshop Agent-Oriented Software Engineering, LNCS 4951, Springer-Verlag, 2007, 16-30.

[12] DeLoach, S. A., "Modeling Organizational Rules in the Multi Agent Systems Engineering Methodology", in Proceedings of the 15th Canadian Conference on Artificial Intelligence, 2002, 54-62.

[13] Giorgini, P., Mylopoulous, J., & Sebastiani, R., "Goal- Oriented Requirements Analysis and Reasoning in the Tropos Methodology", J. of Engineering Applications of Artificial Intelligence, Elsevier, 8(2), 2005, 159-171.

[14] Jureta, J., Faulkner, S., & Schobbens, P., Allocating Goals to Agent Roles During MAS Requirements Engineering. 7th International Workshop Agent-Oriented Software Engineering, LNCS 4405, Springer-Verlag, 2007, 19-34.

[15] Cossentino, M., Gaud, N., Hilaire, V., Galland, S., & Koukam, A., ASPECS: an Agent-Oriented Software Process for Engineering Complex Systems. J. of Autonomous Agents and Multiagent Systems, 20(2), 2009, 260–304.

[16] Pavón, J., & Gómez-Sanz, J., "Agent Oriented Software Engineering with INGENIAS", In Multi-Agent Systems and Applications III, Springer Berlin Heidelberg, 2003, 394-403.

[17] Odell, J., "Objects and Agents Compared", Journal of Object Technology, 1(1), 2002, 41-53.

[18] Lind, J., "Issues in Agent-oriented Software Engineering", In Agent Oriented Software Engineering, Springer Berlin Heidelberg, 2001, 45-58.

[19] Dam, K. H., & Winikoff, M., Comparing Agent-Oriented Methodologies, In Agent-Oriented Information Systems, Springer Berlin Heidelberg, 2004, 78-93.

[20] Sturm, A., & Shehory, O., "A Framework for Evaluating Agent-oriented Methodologies", In Agent-Oriented Information Systems, Springer Berlin Heidelberg, 2004, 94-109.

[21] Sudeikat, J., Braubach, L., Pokahr, A., & Lamersdorf, W. (2005). Evaluation of Agent–oriented Software Methodologies–examination of the Gap between Modeling and Platform. In Agent-Oriented Software Engineering, Springer Berlin Heidelberg, 2005, 126-141.

[22] Ghandehari, E., Saadatjoo, F., & Chahooki, M. A. Z., "AMA: a Compound Methodology for Designing and Implementing Agent-Based

Systems", Journal of Advances in Computer Science, 3(11), 2014, 107-114.

[23] Hsu, Y. L., Lee, C. H., & Kreng, V. B., "The Application of Fuzzy Delphi Method and Fuzzy AHP in lubricant Regenerative Technology Selection", Expert Systems with Applications, 37(1), 2010, 419-425.