علوم و مهندسی کامپیوتر
نشریه علمی پژوهشی انجمن کامپیوتر ایران
مجلد ۱۰، شماره ۱ و ۳ (الف)، ۱۳۹۱
صفحات ۲۶-۳۹
مقاله عادی

موضوعات تحقیقاتی مطرح در مهندسی نرم‌افزار سیستم‌های مبتنی بر عامل

احمد عبداله‌زاده بارفروش شیوا وفادار

دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، ایران

چکیده

مهندسی نرم‌افزار مبتنی بر عامل، به عنوان رویکرد بعدی تولید نرم‌افزار (بعد از شی گرایی) مطرح است. پس از گذشت یک دهه از معرفی این حوزه تحقیقاتی، در این مقاله جایگاه مهندسی نرم‌افزار مبتنی بر عامل بررسی و تحلیل شده است. بدین منظور، در این مقاله یک روش سیستماتیک برای تبیین فضای تحقیقاتی موجود معرفی شده و با پیروی از این روش، حوزه‌های تحقیقاتی در زمینه مهندسی نرم‌افزار مبتنی بر عامل شناسایی گردیده است. در این راستا، تحقیقات انجام شده در هر یک از حوزه‌های متدولوژی و فرآیند، مدل‌سازی و ابزارهای پشتیبانی مهندسی نرم‌افزار مبتنی بر عامل، به صورت اجمالی معرفی شده است. همچنین، با بررسی سیر تکاملی تحقیقات انجام شده در زمینه‌های مختلف، گرایش‌های موجود در این حوزه‌ها مشخص و تحلیل و بر این مبنا، موضوعات مطرح برای تحقیقات آینده در این حوزه معرفی شده است.

کلمات کلیدی: مهندسی نرم‌افزار، عامل، مهندسی نرم‌افزار مبتنی بر عامل.

۱- مقدمه

با بررسی سیر تکاملی دیدگاه‌های تولید نرم‌افزار می‌توان سه مرحله اصلی برای تکامل دیدگاه جدید در تولید نرم‌افزار در نظر گرفت [۱]: ۱) معرفی دیدگاه جدید ۲) به کارگیری ایده در تولید سیستم و جمع‌آوری تجربیات ۳) فراگیر شدن دیدگاه و ورود به حوزه مهندسی نرم‌افزار. در مرحله ورود به حوزه مهندسی نرم‌افزار، براساس تجربیات موفق تولید کنندگان نرم‌افزار، متدولوژی‌ها و فرآیندهای تولید مشخص می‌شوند و به صورت مدون و سیستماتیک ارائه می‌گردند. روش‌های مختلف تحلیل، طراحی، تست و مدل‌سازی مورد استفاده به صورت مشخص و استاندارد برای دیدگاه مذکور تهیه می‌گردند و ابزارهای مختلف برای پشتیبانی فرآیند تولید نرم‌افزار براساس دیدگاه جدید توسعه می‌یابند.

با توجه به قابلیت‌های دیدگاه مبتنی بر عامل از جنبه مهندسی نرم‌افزار، از قبیل نزدیک بودن به دیدگاه انسان برای شکستن مساله، افزایش سطح تجرید و بسته‌بندی بلوکهای پایه، افزایش امکان استفاده مجدد، افزودن یافته‌های هوش مصنوعی به نرم‌افزار و افزایش توانایی حل مسایل پیچیده، مهندسی نرم‌افزار مبتنی بر عامل رویکرد بعدی مهندسی نرم‌افزار پس از شی‌گرایی معرفی شده است [۲].

پس از معرفی دیدگاه عامل در اواسط دهه ۵۰ میلادی [۳] و کاربردی شدن این دیدگاه با معرفی هوش مصنوعی توزیع شده، در اواسط دهه ۷۰ میلادی [۴]، استفاده از عامل‌ها در ساخت سیستم‌های نرم‌افزاری گسترش یافت. بدین ترتیب در طی دو دهه، تجربیات متعددی در ساخت سیستمهای مبتنی بر عامل حاصل گردید [۵]. در سال ۱۹۹۹، نظریه مهندسی نرم‌افزار مبتنی بر عامل مطرح گردید. براساس این نظریه، به علت فراگیر شدن دیدگاه مبتنی بر عامل و وجود تجربیات موفق برای تولید سیستم‌های مبتنی بر عامل، امکان ورود به مرحله سوم از سیرتکاملی دیدگاه مبتنی بر عامل احساس شد. بدین ترتیب حوزه تحقیقاتی مهندسی نرم‌افزار مبتنی بر عامل معرفی گردید [۱].

بنابر تعریف IEEE مهندسی نرم‌افزار به کارگیری یک رویکرد سیستماتیک، باقاعده و کمی برای تولید، عملیاتی نمودن و نگهداری نرم‌افزار است. روشی که منجر به تولید نرم‌افزار به صورت مهندسی می‌گردد [۶]. مهندسی نرم‌افزار مبتنی بر عامل به جنبه مهندسی سیستم‌هایی می‌پردازد که از تکنولوژی عامل در ساخت آنها استفاده می‌شود و سعی در ارائه روش‌ها و ابزارهایی دارد که به صورت خاص

در ساخت سیستم‌های مبتنی بر عامل مورد نیاز است و از ویژگی‌های عامل و جامعه عامل‌ها ناشی می‌شود.

با گذشت بیش از یک دهه از معرفی مهندسی نرم‌افزار مبتنی بر عامل، تبیین و تحلیل جایگاه آن به عنوان رویکرد بعدی تولید نرم‌افزار، انگیزه این تحقیق بوده است. بدین منظور مجموعه‌ای از تحقیقات انجام شده در این زمینه، در بین سال‌های ۱۹۹۹- ۲۰۰۸ گردآوری و گزارش شده است. علت انتخاب این دوره، شکوفایی رویکرد مبتنی بر عامل در طی دهه اول ظهور آن بوده است. بدیهی است در سال‌های بعد نیز تحقیقات مختلفی در این حوزه صورت گرفته است که از نمونه‌های آن می‌توان از [۷]، [۸]، [۹] در زمینه متدولوژی و فرآیند، [۱۰]، [۱۱] در زمینه روش‌های مدل‌سازی و [۱۲]، [۱۳] در زمینه ابزارها نام برد. در این مقاله، تحلیلی آماری بر روی تحقیقات انجام شده در این زمینه صورت گرفته است. همچنین گرایش‌های موجود برای تحقیق در این زمینه مشخص و نقاط ضعف، قوت و کاستی‌های تحقیقات در این حوزه معرفی می‌گردد.

برای دستیابی به اهداف فوق و گزارش فرآیند انجام تحقیق، ساختار این مقاله به صورت زیر تنظیم شده است. در بخش ۲، روش مورد استفاده در انجام تحقیق و نحوه گردآوری مجموعه منابع مورد استفاده، معرفی می‌شود و فضای تحقیق در این حوزه ترسیم می‌گردد. تحلیل آماری تحقیقات انجام شده در بخش ۳ ارائه شده است. با دسته‌بندی تحقیقات انجام شده، هفت زمینه تحقیقاتی اصلی در مهندسی نرم‌افزار مبتنی بر عامل تشخیص داده شده است؛ متدولوژی‌ها و فرآیندهای تولید نرم‌افزار، روش‌های مدل‌سازی، مباحث مرتبط با نیازمندی‌ها، روش‌های تحلیل و طراحی، پیاده‌سازی، تست و ابزار. در این مقاله تحقیقات مرتبط با متدولوژی و فرآیند (بخش ۴)، مدل‌سازی (بخش ۵) و ابزارها (بخش ۶) مورد تحلیل و بررسی قرار گرفته است. در هر یک از این بخش‌ها، تحقیقات انجام شده در هر یک از این مباحث، گرایش‌های موجود و سیر تکاملی تحقیق براساس دسته‌بندی تحقیقات مورد بررسی صورت گرفته است. در بخش ۷ نیز، تحلیل فضای تحقیق و تعیین کاستی‌های موجود انجام شده است.

۲- روش تحقیق

در این قسمت، روش تحقیق استفاده شده برای تبیین فضای تحقیق در زمینه مهندسی نرم‌افزار سیستم‌های مبتنی بر عامل ارائه می‌گردد. فرآیند تحقیق را می‌توان در پنج مرحله اصلی خلاصه نمود:

۱- تعیین فضای تحقیق

۲- جمع‌آوری مقالات منبع

۳- دسته‌بندی منابع و تعیین زمینه‌های تحقیقاتی

۴- مطالعه منابع و گزارش تحقیقات انجام شده

۵- تحلیل مباحث مطرح در هر یک از زمینه‌های تحقیقاتی

برای تعیین فضای تحقیق، تحقیقات انجام شده به دو نیمه - قبل از سال ۲۰۰۵ و پس از آن- تقسیم شده است. برای بررسی تحقیقات انجام شده در قبل از سال ۲۰۰۵ از تعدادی مقاله مروری استفاده شده است [۲، ۱۴، ۱۵، ۱۶] این مقالات مجموعه‌ای از تحقیقات در زمینه مهندسی نرم‌افزار سیستم‌های مبتنی بر عامل را در سال‌های مختلف گزارش نموده‌اند. مجموعه مقالات مروری، به عنوان منبع اصلی گزارش تحقیقات انجام شده قبل از سال ۲۰۰۵ درنظرگرفته شده‌اند. برای تکمیل مجموعه مقالات منبع لازم است سایر تحقیقات انجام شده در این زمینه، در دوره زمانی مورد بررسی نیز در نظر گرفته شوند. برای گزارش تحقیقات جدیدی که در زمینه مهندسی نرم‌افزار مبتنی بر عامل پس از سال ۲۰۰۵ انجام شده است، تعدادی از مجلات و کنفرانس‌های معتبر مرتبط با عامل و سیستم‌های مبتنی بر عامل (و مهندسی نرم‌افزارسیستم‌های مبتنی بر عامل) انتخاب شده‌اند. معیار اصلی انتخاب مجلات، میزان ارتباط آنها با موضوع عامل و سیستم‌های مبتنی بر عامل و اعتبار و عمومیت آنها در جامعه محققان سیستم‌های مبتنی بر عامل بوده است.

کنفرانس‌های مرتبط با موضوع یکی دیگر از مراجع مورد استفاده در این تحقیق بوده است. مهمترین معیارهای انتخاب کنفرانس‌های مربوطه سرفصل مطالب مورد توجه در کنفرانس، نرخ پذیرش مقالات و تعداد مقالات دریافت شده توسط کنفرانس‌ها بوده است. با توجه به اینکه موضوع اصلی این مقاله، مهندسی نرم‌افزارسیستم‌های مبتنی بر عامل بوده است، برخی کارگاه‌های خاص مرتبط با موضوع نیز در نظر گرفته شده اند که در آنها به صورت خاص به موضوع مهندسی نرم‌افزار در این سیستم‌ها پرداخته شده است. جهت بررسی تحقیقات داخلی انجام شده در ایران در زمینه عامل و سیستم‌های مبتنی بر عامل، کنفرانس انجمن کامپیوتر ایران به عنوان معتبرترین و عمومی‌ترین کنفرانس داخلی در زمینه مباحث مرتبط با علوم کامپیوتر نیز در این بررسی در نظر گرفته شده است. لیست کامل مجلات و کنفرانس‌های مرجع در [۱۷] ارائه شده است.

برای جمع‌آوری مقالات منبع، مجموعه مقالات منتشر شده در مراجع و کنفرانس‌های تعیین شده در سال‌های اخیر، گردآوری شده است. در مجموع ۴۶ مقاله مجله و ۴۰۶ مقاله کنفرانس براساس منابع فوق جمع‌آوری شده و به عنوان مقالات منبع مورد استفاده قرار گرفته است.

پس از جمع‌آوری مجموعه مقالات منبع، گام بعدی دسته‌بندی منابع و تعیین زمینه‌های تحقیقاتی بوده است. در این دسته‌بندی، هدف مشخص کردن موضوعات تحقیقاتی مختلفی است که در سیستم‌های مبتنی بر عامل وجود دارد.

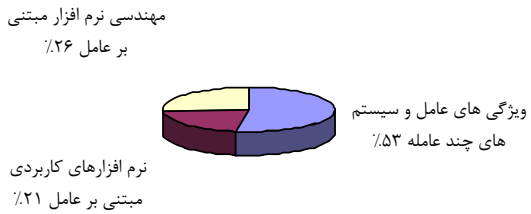
گام بعدی فرآیند تحقیق، مطالعه منابع و گزارش تحقیقات انجام شده می‌باشد. نتایج این مرحله در بخش‌های ۴ تا ۶ مقاله به صورت مفصل در زمینه متدولوژی و فرآیندهای تولید نرم‌افزار، روش‌های مدل‌سازی و ابزارهای اتوماتیک برای پشتیبانی فرایند تولید نرم‌افزار تشریح شده است. حاصل این مرحله گزارش آخرین تحقیقات انجام شده در زمینه هر یک از موضوعات می‌باشد.

تحلیل فضای تحقیق و دسته‌بندی تحقیقات انجام شده در هر یک از زمینه‌های مهندسی نرم‌افزار مبتنی بر عامل، آخرین مرحله در روش تحقیق معرفی شده است. معیارهای تحلیلی انتخاب شده گرایش‌های موجود در هر یک از زمینه‌های تحقیقاتی در سال‌های مختلف و نوع تحقیقات انجام شده در سال‌های اخیر و سال‌های اولیه می‌باشد. همچنین با استفاده از این تحلیل‌ها، کمبودهای موجود مشخص و زمینه‌های موجود برای تحقیقات آینده مشخص شده است. نتایج این تحلیل‌ها در پایان بخش‌های ۴ تا ۶ ارائه شده است.

۳- تحلیل آماری جایگاه مهندسی نرم‌افزار مبتنی بر عامل

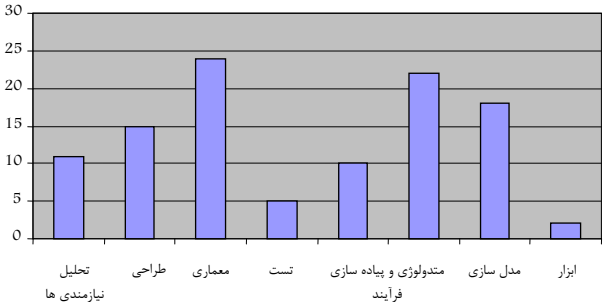
براساس روش تحقیقی که در بخش ۲ تشریح گردید، ۴۶ مقاله مجله و ۴۰۶ مقاله کنفرانس جمع‌آوری گردیده است. از میان این مقالات ۲۳۸ مقاله به موضوع عامل و ویژگی‌های عامل پرداخته‌اند که در آنها مباحثی مانند مفاهیم شناختی، یادگیری، جستجو، استنتاج، برنامه‌ریزی، محیط عامل، همکاری و هماهنگی میان عامل‌ها، نحوه ارتباط در سیستم‌های مبتنی بر عامل، ویژگی‌های اجتماعی عامل، ساختارهای اجتماعی سیستم‌های مبتنی بر عامل، مذاکره میان عامل‌ها و واژگان شناختی پرداخته شده است. تحقیقاتی که در آنها نتایج پیاده‌سازی سیستم با استفاده از تکنولوژی عامل گزارش شده است، مشتمل بر ۹۶ مقاله می‌باشند. در مجموع ۱۱۹ مقاله از مقالات جمع‌آوری به مبحث مهندسی نرم‌افزار سیستم‌های مبتنی بر عامل پرداخته‌اند. لیست کامل این مقالات در [۱۷] ارائه شده است. شکل

۱، سهم هر یک از موضوعات فوق را در فضای تحقیق در سیستم‌های مبتنی بر عامل نشان می‌دهد.



شکل۱- سهم موضوعات تحقیقاتی در هوش مصنوعی توزیع‌شده

با دسته‌بندی مقالات در حوزه مهندسی نرم‌افزار مبتنی بر عامل، سهم هر یک از موضوعات مهندسی نرم‌افزار در این حوزه مشخص می‌شود. نمودار توزیع تحقیقات انجام شده در این زمینه در شکل ۲ نشان داده شده است. با بررسی شکل می‌توان دید که از میان تحقیقات انجام شده در این حوزه، بیشترین سهم به ترتیب مربوط به موضوعات مرتبط با معماری عامل و سیستم چندعامله، متدولوژی و فرآیند و مدل‌سازی است. درحالی که تعداد مقالاتی که در آنها موضوع اصلی ابزارهای یکپارچه مهندسی نرم‌افزار مبتنی بر عامل است، در این بین کمترین سهم را داشته‌اند.



شکل ۲- تحقیقات گزارش شده در موضوعات مختلف مهندسی نرم‌افزار مبتنی بر عامل

در این مقاله، تحقیقات انجام شده در موضوعات مرتبط با متدولوژی و فرآیند، مدل‌سازی و ابزارهای اتوماتیک برای پشتیبانی فرآیندهای مهندسی نرم‌افزار، به عنوان مباحث عمومی مطرح در این حوزه تحقیقاتی مورد بررسی و تحلیل قرار می‌گیرند. در ادامه زمینه‌های تحقیقاتی مورد توجه در هر یک از این موضوعات به تفصیل ارائه می‌گردد.

۴- متدولوژی و فرآیندهای تولید سیستم‌های مبتنی بر عامل

هدف از ارائه متدولوژی‌ها آن است که طراحان سیستم بتوانند به صورت سیستماتیک نیازمندی‌های کاربران سیستم را به مدل طراحی تبدیل کنند، به طوری که این طراحی دارای جزییات کاملی از سیستم باشد و در نتیجه به راحتی قابل پیاده‌سازی باشد. پس از معرفی دیدگاه مهندسی نرم‌افزار مبتنی بر عامل، متدولوژی‌های مهندسی نرم‌افزار مبتنی بر عامل یکی از زمینه‌های تحقیقاتی بودند که بسیار مورد توجه قرار گرفتند. به طوری که متدولوژی‌های مختلفی در طی سال‌های اولیه ارائه شد. حجم بالایی از Surveyهای موجود در مهندسی نرم‌افزار که در سال‌های اولیه ارائه شده است، به متدولوژی‌ها اختصاص یافته است [۱۴، ۱۵]. اما در سال‌های اخیر، تمرکز و توجه از تولید متدولوژی به بهبود و اصلاح آن سوق یافته است. مجموعه مقالاتی که در گروه متدولوژی و فرآیندهای مهندسی نرم‌افزار مبتنی بر عامل دسته‌بندی شده‌اند را می‌توان به دسته‌های زیر تقسیم کرد:

- بهبود و توسعه متدولوژی‌های موجود
- ارائه متدولوژی‌های خاص منظوره
- ارزیابی متدولوژی‌های موجود
- تلفیق و استفاده مجدد از اجزای متدولوژی

در ادامه تحقیقات انجام شده در زمینه متدولوژی‌های مهندسی نرم‌افزار مبتنی بر عامل به صورت خلاصه گزارش می‌گردند.

۴-۱- متدولوژی Gaia و توسعه‌های آن

یکی از نخستین متدولوژی‌های ارائه شده، Gaia نام دارد که در آن مراحل تحلیل و طراحی پشتیبانی شده است. تحلیل در متدولوژی Gaia شامل مراحل مدل‌سازی نقش‌ها و مدل‌سازی تعامل است. در مرحله طراحی مدل عامل‌ها، مدل سرویس و مدل شناخت سیستم تولید می‌گردد [۱۸]. در این متدولوژی عامل‌ها در دو سطح مختلف، سطح اجتماع عامل‌ها و ساختار سازمانی سیستم و سطح ساختار عامل مورد بررسی قرار می‌گیرند. با توجه به اینکه متدولوژی Gaia یکی از نخستین متدولوژی‌های ارائه شده برای سیستم‌های مبتنی بر عامل بوده است، در نسخه اصلی این متدولوژی کمبودهایی وجود داشته و تحقیقات مختلفی برای بهبود آن، صورت گرفته است.

یکی از بهبودهایی که برای متدولوژی Gaia ارائه شده است، توانایی مدل‌کردن سیستم‌های قابل پیاده‌سازی در اینترنت را به این متدولوژی اضافه نموده است [۱۹] در این تحقیق به ویژگی‌هایی از قبیل بازبودن و تضاد اهداف عامل‌های شرکت کننده در سیستم توجه شده است. بدین منظور قابلیت مدل‌سازی روابط اجتماعی عامل به متدولوژی اضافه شده است. در این بهبود، براساس توصیف نیازمندی‌های سیستم، مدل تعامل و قوانین اجتماعی به مرحله تحلیل متدولوژی اضافه شده است. همچنین در مرحله طراحی، مدل رفتار رسانه هماهنگ‌کننده به متدولوژی اضافه شده است. علاوه بر اینکه مدل قوانین اجتماعی و مدل تعامل که در مرحله تحلیل اضافه شده‌اند، بر سایر مدل‌های طراحی مانند مدل عامل و مدل سرویس نیز تاثیرگذار است. بدین ترتیب رفتار سیستم در قبال تعامل‌هایی که در آنها شرکت می‌کند، مشخص می‌شود.

با استفاده از بهبود دیگری که بر روی متدولوژی Gaia صورت گرفته است، متدولوژی جدیدی با عنوان ROADMAP ارائه شده است. در این متدولوژی جدید توانایی مدل‌کردن سیستم‌های باز به متدولوژی Gaia اضافه شده است [۲۰]. در این تحقیق، چهار بهبود در متدولوژی Gaia ارائه شده است. مدلهای رسمی برای دانش و محیط سیستم، ساختار سلسله مراتبی نقش‌ها، ارتباط صریح ساختارهای اجتماعی و توانایی مدیریت تغییرات پویا.

بهبود دیگری که بر روی متدولوژی Gaia صورت گرفته است [۲۱]، با استفاده از ترکیب آن با روش مدل‌سازی AUML می‌باشد. در این بهبود، مرحله طراحی در متدولوژی Gaia درنظر گرفته شده است و در چهار مرحله پروتکل، تعامل، عامل و سازمان مدل‌های توسعه یافته UML برای عامل‌ها بهبود یافته است.

در [۲۲] دو بهبود دیگر برای متدولوژی پیشنهاد شده است. نخست آنکه مرحله طراحی عامل به متدولوژی اضافه شده است. براساس این بهبود، مرحله طراحی عامل که در متدولوژی اولیه به صورت یک جعبه سیاه درنظر گرفته شده

است، براساس معماری‌های شناخته شده برای عامل، گسترش یافته است. بدین منظور، مدلها و فعالیت‌هایی درنظر گرفته شده است تا مشخص شود که چگونه عامل، اهدافی که به آن تخصیص یافته است را برآورده می‌نماید. مدل ساختاری (شامل کلاس‌هایی که یک نقش به آنها نیاز دارد) و مدل عملکرد (شامل سناریوهایی که برای انجام فعالیت‌های موردنیاز نقش، مورد استفاده قرار می‌گیرد) به متدولوژی اضافه شده است. این مدل‌ها در فعالیت‌های انتخاب معماری عامل (براساس الگوهای طراحی)، ایجاد نمودار کلاس، ایجاد سناریوها در مرحله طراحی تولید می‌شوند. همچنین برای بهبود فرایند تولید سیستم مبتنی بر عامل با استفاده از Gaia یک رویکرد مبتنی بر تکرار در متدولوژی درنظر گرفته شده است. براساس این پیشنهاد، برای جلوگیری از تولید سیستمی که با نیازهای کاربر منطبق نیست و برای غلبه بر پیچیدگی سیستم، بهتر است سیستم در چندین تکرار تولید شود.

۴-۲- متدولوژی MaSE و توسعه‌های آن

متدولوژی MaSE (Multiagent System Engineering) یک متدولوژی برای تحلیل و طراحی سیستم‌های مبتنی بر عامل می‌باشد و از نقاط قوت آن این است که در آن ابزاری به نام agentTool ارائه شده است که کلیه مراحل تحلیل و طراحی سیستم در متدولوژی MaSE را پشتیبانی می‌نماید [۲۳]. مرحله تحلیل در MaSE، شامل سه مرحله اصلی تعیین اهداف سیستم، اعمال موارد کاربرد و بازبینی نقش‌ها است. مرحله طراحی در MaSE شامل چهار مرحله ایجاد کلاس‌های عامل، ساخت گفتگوها، ترکیب کلاس‌های عامل و طراحی سیستم می‌باشد.

متدولوژی MaSE نیز از زمان ارائه نسخه اولیه تاکنون، به صورت‌های مختلف بهبود و توسعه یافته است. در یکی از بهبودهای متدولوژی MaSE، مرحله مدل سازی هستان‌شناسی به مرحله تحلیل متدولوژی اضافه شده است [۲۴]. در این توسعه برای متدولوژی، ابتدا مشخص می‌گردد که هدف و محدوده هستان‌شناسی موردنیاز عامل چیست. سپس داده‌های موجود در دامنه سیستم که بدین منظور مورد استفاده قرار می‌گیرند، جمع‌آوری می‌گردد.

همچنین امکان مدل‌سازی روابط سازمانی میان عامل‌ها نیز در توسعه دیگری به متدولوژی MaSE افزوده شده است [۲۵]. در این توسعه، پس از مرحله مدل سازی هستان شناسی عامل در مرحله تحلیل، تحلیل و مدل‌سازی ساختار سازمانی انجام می‌شود. در این مرحله، قوانین سازمانی که محدودیت‌های رفتار عامل و ارتباط و تعامل میان عامل‌های یک سازمان است، مشخص می‌شود. بدین ترتیب قوانین سازمانی در قالب تعدادی محدودیت بر روی رفتار نقش (در مرحله تحلیل) ارائه می‌شود. برای این منظور، نمادهایی برای توصیف این محدودیت‌ها درنظر گرفته شده است.

افزودن قابلیت حرکت به عامل‌های سیستم از دیگر بهبودهای متدولوژی MaSE می‌باشد [۲۶]. بدین منظور، در مرحله تحلیل، در مدل‌سازی فعالیت‌ها که در قالب نمودار حالت انجام می‌شود، دستور حرکت به متدولوژی اضافه شده است. در این مرحله، نتیجه حرکت و علت شکست آن (در صورت شکست) مشخص می‌شود تا امکان بازیابی عامل از این شکست، فراهم شود. همچنین در مرحله طراحی، این امکان به عامل‌ها اضافه شده است که با دریافت دستور حرکت، وضعیت داخلی همه اجزای خود را ذخیره نمایند تا پس از انتقال به مقصد امکان شروع مجدد فعالیت‌ها از وضعیتی که در آن قرار داشته‌اند، فراهم شود.

همچنین در توسعه دیگری برای این متدولوژی که منجر به ارائه متدولوژی Ex-MaSE شده است، مدل محیط و مدل دانش به مرحله تحلیل متدولوژی اضافه شده است. همچنین مرحله ساخت کلاس‌های عامل در متدولوژی به مرحله ساخت «کلاس عامل- شی» تغییر یافته است [۲۷].

همچنین به منظور افزودن قابلیت توانایی پشتیبانی مفاهیم مرتبط با مدل‌سازی سیستم‌های مبتنی بر عامل که بر ساختار سازمانی بنا می‌شود، مفاهیمی مانند هدف، نقش، عامل، توانایی‌ها و انتساب عامل به نقش‌ها به عنوان الزامات موردنیاز تشخیص داده شده‌اند و متدولوژی برای پشتیانی این مفاهیم توسعه داده شده است [۲۸].

بهبود دیگری که در متدولوژی MaSE صورت گرفته است، استفاده از روش Method Fragment برای ارائه یک فرآیند تولید نرم‌افزار است [۲۹]. در این بهبود، فرآیند قابل سازمان‌دهی تعریف شده است که با استفاده از آن می‌توان متدولوژی را برای کاربردهای مختلف اختصاصی‌سازی نمود. بدین منظور، Meta Model مورد نیاز فرآیند، مجموعه‌ای از Method Fragmentها و مجموعه‌ای از راهنمایی‌ها برای اختصاصی نمودن و استفاده از آنها ارائه شده است. در این فرآیند، سه مرحله مهندسی نیازمندی‌ها، تحلیل و طراحی درنظر گرفته شده است. فعالیت‌های مهندسی نیازمندی‌ها، مدل‌سازی اهداف و اصلاح اهداف هستند که با استفاده از درخت اهداف AND_OR مدل‌سازی می‌شود. در مرحله تحلیل، مدل‌سازی ساختار سازمانی، مدل سازی نقش‌ها، مدل سازی دامنه و اصلاح نقش‌ها انجام می‌شود. در مرحله طراحی، فعالیت‌های مدل کلاس عامل‌ها، مدل سازی پروتکل‌ها، مدل‌سازی برنامه‌ها، مدل‌سازی توانایی‌ها، مدل‌سازی فعالیت‌ها و مدل‌سازی سرویس‌ها صورت می‌گیرد.

۴-۳- متدولوژی Tropos و توسعه‌های آن

Tropos یکی از متدولوژی‌های نسبتا جدید در مهندسی نرم‌افزار مبتنی بر عامل است [۳۰]. تفاوت اصلی این متدولوژی، با سایر متدولوژی‌ها تمرکز آن بر روی تحلیل نیازمندی‌ها است به طوری که در مرحله اول ذینفعان سیستم و تمایلات آنها مورد بررسی و تحلیل قرار می‌گیرد. فرآیند تولید نرم‌افزار در این متدولوژی شامل پنج مرحله تحلیل اولیه نیازمندی‌ها، تحلیل ثانویه نیازمندی‌ها، طراحی معماری، طراحی جزیی و پیاده‌سازی است. در طراحی متدولوژی Tropos، از تئوری‌های ساختارهای اجتماعی مانند تئوری سازمانی، طراحی فرآیند و پیوستگی استراتژیک الهام گرفته شده است.

با وجود اینکه متدولوژی Tropos، به عنوان یک متدولوژی نسبتا جدید در سیستم‌های مبتنی بر عامل ارائه شده است، توسعه‌های متعددی برای این متدولوژی پیشنهاد شده است. در [۳۱]، یک مدل رسمی تحلیل هدف به متدولوژی Tropos اضافه شده است که باعث تقویت این متدولوژی در مرحله تحلیل نیازمندی‌ها گردد. این تحقیق، با استفاده از ابزارها و مدل‌های استدلال رو به جلو و رو به عقب، و با به کارگیری روش تحلیل هدف رسمی، در مرحله تحلیل نیازمندیها، امکان یافتن اهداف متضاد تعریف شده در مرحله تحلیل نیازمندی‌ها و تعیین اهدافی که در صورت برآورده شدن، هدف سیستم به صورت کلی برآورده می‌شود را فراهم نموده است. همچنین با استفاده از این روش، این امکان وجود دارد که با استفاده از استدلال رو به جلو مشخص گردد، برای اینکه هدف سیستم برآورده شود، کدام مجموعه اهداف سیستم باید برآورده شوند. با به کارگیری این روش، امکان وجود گزینه‌های مختلف در مرحله تحلیل سیستم فراهم می‌شود و تحلیلگر سیستم امکان انتخاب میان اهداف مختلف را خواهد داشت. بدین منظور گراف هدف و نمادهای ارتباط میان اهداف برای توصیف هدف‌ها و ارتباط میان آنها به صورت رسمی، ارائه شده است.

همچنین در [۳۲] تحلیل اهداف سیستم‌ها و گزینه‌های مختلف برای اختصاص هدف به نقش در سیستم مبتنی بر عامل در مرحله تحلیل اهداف سیستم به

متدولوژی Tropos درنظر گرفته شده است. بدین منظور یک روش تخصیص هدف به نقش ارائه شده است که امکان تولید گزینه‌های مختلف برای نقش‌های سیستم و انتخاب میان آنها را فراهم می‌کند.

در تحقیق دیگری که در ارتباط با متدولوژی Tropos انجام شده است، به نیازمندی‌های تعامل میان عامل در سیستم‌های مبتنی بر عامل و متدولوژی Tropos پرداخته است [۳۳]. بنابر نظر نویسندگان مقاله نیازمندی‌های تعامل عامل در این متدولوژی در مراحل اولیه موردتوجه قرار نمی‌گیرد و پروتکل‌های ارتباطی تا مرحله طراحی جزیی سیستم مشخص نمی‌شوند. از طرف دیگر، در نظریه Commitment Protocol پروتکل‌های تعامل و زمینه اجرای آنها در سیستم کاربردی مشخص نمی‌شود. برای حل این مساله در این مقاله، از ترکیب این دو موضوع استفاده شده است به طوری که Commitment Protocolها به متدولوژی Tropos اضافه شده است. بدین منظور راهنمایی‌هایی برای تعیین تعامل میان بازیگران مختلف سیستم براساس مفاهیم ارائه شده در متدولوژی (مانند وابستگی میان بازیگران) ارائه شده است. براساس این راهنمایی‌ها به صورت سیستماتیک می‌توان مشخص کرد که چه پروتکل‌هایی میان بازیگران مختلف سیستم باید تعریف گردد.

۴-۴- سایر متدولوژی‌ها

علاوه بر متدولوژی‌های فوق که در مهندسی نرم‌افزار مبتنی بر عامل بسیار مورد توجه و استفاده هستند، متدولوژی‌های دیگری نیز در این حوزه معرفی شده‌اند، اما تحقیقات زیادی برای توسعه و بهبود آنها گزارش نشده است. در این بخش به معرفی اجمالی این متدولوژی‌ها می‌پردازیم.

- متدولوژی MASSIVE که در آن هفت دیدگاه محیط، وظیفه، نقش، تعامل، جامعه، معماری و سیستم مدل‌سازی می‌شود [۳۴].
- متدولوژی MESSAGE که در آن مفاهیم نقش، سازمان، منابع در قالب موجودیت‌های سیستم، وظیفه و پروتکل‌های تعامل در قالب فعالیت‌های موجود در سیستم و هدف به عنوان حالت ذهنی عامل تحلیل می‌شوند، [۳۵]. و در پی آن متدولوژی INGENIAS که در آن پنج مدل عامل، تعامل، وظیفه و هدف، سازمان، محیط تولید می‌شود [۳۶].
- متدولوژی MAS-CommonKADS که توسعه متدولوژی‌های مبتنی بر دانش است و در آن فعالیت‌های مدل‌سازی عامل، مدل کردن وظایف، مدل کردن هماهنگی، مدل کردن دانش، مدل کردن ساختار سازمانی در مرحله تحلیل صورت می‌گیرد. طراحی در این متدولوژی شامل طراحی شبکه عامل‌ها، طراحی عامل و طراحی ساختار زیربنایی عامل است [۳۷].
- متدولوژی PASSI که در آن مدل‌های نیازمندی‌های سیستم، جامعه عامله، پیاده‌سازی عامل، کد برنامه و آرایش ساختاری ایجاد می‌شود [۳۸].
- متدولوژی Premetheus که در آن مراحل توصیف سیستم، طراحی معماری و طراحی جزییات و خطایابی درنظر گرفته شده است [۳۹].
- RICA که برپایه نقش، تعامل و ارتباط بنا شده است [۴۰].
- ODAC [۴۱] که در آن پنج دیدگاه Enterprise، اطلاعات، محاسبات، تکنولوژی و مهندسی [۴۲] مدل می‌شوند.

۴-۵- دیدگاه‌های خاص در تولید متدولوژی

در این قسمت، متدولوژی‌هایی که براساس دیدگاه‌های خاص ایجاد شده‌اند، معرفی می‌شوند. این متدولوژی‌ها براساس دیدگاه عمومی در تولید متدولوژی‌ها شکل نگرفته‌اند. بلکه دیدگاه‌های خاصی در آنها وجود داشته است که کل مراحل تولید متدولوژی را تحت تاثیر قرار داده است.

دیدگاه درنظر گرفتن ذینفعان سیستم در فرآیند تولید نرم‌افزار موجب معرفی متدولوژی شده است که رشد عامل نرم‌افزاری مبنای فرآیند توسعه سیستم قرار می‌گیرد. براساس این دیدگاه که ذینفعان سیستم باید در تولید سیستم دخالت داشته باشند، شرایطی که در آن عامل در سیستم عمل می‌کند، شبیه‌سازی می‌شود. بدین ترتیب شرایط مختلفی که عامل در آن قرار می‌گیرد، مورد بررسی قرار می‌گیرد. یکی از این شرایط وضعیتی است که عامل خود قادر به اجرای رفتار خاصی نیست و برای تصمیم‌گیری به کمک نیاز دارد. در این حالت ذینفع سیستم موظف به راهنمایی عامل برای تصمیم‌گیری است [۴۳].

برای افزایش قابلیت تغییر سیستم‌های تولید شده، متدولوژی ارائه شده است که در آن امکان نگاشت میان اهداف سیستم (به عنوان نیازمندی‌ها) و رفتارهای عامل وجود دارد. بدین ترتیب، این امکان وجود دارد که با ارزیابی رفتارهای عامل و میزان موثر بودن آنها، در صورت نیاز به تغییر رفتار عامل، اهداف متناظر با رفتار را نیز تغییر داد و برعکس. بدین ترتیب امکان اصلاح طراحی براساس تغییر نیازهای سیستم، در یک فرآیند تکرارشونده به وجود می‌آید [۴۴].

برای افزایش سهم انسان در تولید سیستم‌های مبتنی بر عامل، متدولوژی ارائه شده است که در آن روش‌های جهان مجازی سه‌بعدی و موسسه‌های الکترونیکی، ترکیب شده است [۴۵].

دیدگاه دیگری که برای تولید سیستم‌های مبتنی بر عامل ارائه شده است، عامل‌ها را براساس نیاز کاربر در زمان اجرا ایجاد می‌نماید. براساس این دیدگاه، لزوما هر آنچه درزمان طراحی به عنوان عامل درنظر گرفته شده است، در زمان اجرا ایجاد نمی‌شود بلکه با درنظر گرفتن هزینه‌های ایجاد عامل و نیاز کاربر، عامل‌های فعال در سیستم انتخاب می‌شوند [۴۶].

۴-۶- ارزیابی متدولوژی‌های موجود

از آنجا که متدولوژی‌های مختلفی برای سیستم‌های مبتنی بر عامل ارائه شده است، تحقیقات متعددی نیز برای ارزیابی متدولوژی‌های موجود صورت گرفته است. دراین تحقیقات سعی شده است که متدولوژی‌های ارائه شده براساس معیارهای مختلف مورد ارزیابی قرار گیرند.

در یکی از تحقیقاتی که در این زمینه انجام شده است ، سه معیار اصلی میزان مستندات موجود در ارتباط با متدولوژی، میزان شناخته شده بودن در جامعه محققان و کاربران و خاص منظوره نبودن متدولوژی برای ارزیابی متدولوژی‌ها انتخاب شده است. بنابر ارزیابی انجام شده در این تحقیق، براساس این معیارها متدولوژی‌های Gaia, MaSE, Tropos, Agent SE, MASSIVE, Promehteous, MAS-CommonKADS, PASSI, MESSAGE متدولوژی‌های برتر هستند. در این تحقیق همچنین متدولوژی‌های مذکور براساس معیارهای مرتبط با پشتیبانی ویژگیهای عامل بودن، مدل‌سازی، ارتباط، فرآیند، برنامه‌کاربردی و کاربر نیز مورد ارزیابی قرار گرفته‌اند. اگرچه در ارزیابی هر یک از معیارها، متدولوژی‌های مختلف در رده های مختلف قرار می‌گیرند، اما براساس نتایج ارزیابی براساس کلیه معیارها، متدولوژی MaSE برترین متدولوژی معرفی شده است [۴۷].

در تحقیق دیگری که در [۴۸] گزارش شده است، متدولوژی‌های مبتنی بر عامل از دیدگاه فرآیندی مورد ارزیابی قرار گرفته‌اند. براساس نتایج این ارزیابی، متدولوژی‌های مبتنی بر عامل بیشتر براساس دیدگاه آبشاری و تکرار شونده و تکاملی شکل گرفته‌اند و دیدگاه‌های دیگری مانند دیدگاه Agile، دیدگاه‌های مبتنی بر ارائه چندین جنبه از سیستم، فرآیندهای تلفیق متدولوژی‌های موجود،

افزایش کیفیت و ابزارها از مسایلی هستند که نیاز بـه تحقیقـات بیشـتری در آنهـا وجود دارد.

ارزیابی دیگری که بر روی متدولوژی‌های مبتنی بر عامل صورت گرفته اسـت، میزان پیچیدگی متدولوژی‌های ارائه شده را براساس نمودارها، جـداول و الگوهـای متنی در چهار سطح، تحلیل سطح صفر، تحلیل سطح یک، طراحی کلی و طراحـی جزیی مقایسه نموده است [۴۹].

۴-۷- حرکت به سمت تلفیق متدولوژی‌های موجود

مهندسی متدولوژی (Method Engineering) یـک رویکـرد جدیـد بـرای تولیـد متدولوژی‌های مهندسی نرم‌افزار است که در آن به جای ایجاد یک متـدولوژی کـه در آن اجزای متدولوژی دارای اجزایی هستند که امکان عملکرد آنها با هـم وجـود دارد، می‌توان یک متدولوژی را بـه اجـزا یـا قسـمت‌هـایی تجزیـه کـرد کـه دارای واسطهایی هستند که امکان ترکیب آنها با هم وجود دارد. بدین ترتیب می‌تـوان از اجزای متدولوژی برای تولید متدولوژی‌های مختلف استفاده کرد.

تعدادی از تحقیقات انجام شده در زمینه مهندسی نرم‌افزار سیستم‌های مبتنی بر عامل نیز به مهندسی متدولوژی پرداخته اسـت. بـرای شکسـتن متـدولوژی بـه اجزای کوچکتر از چارچوب ارائـه شـده در OPF (Open Process Framework) استفاده شده است که در آن فرامدل، مجموعه‌ای از اجزای موجود و راهنمایی برای نمادهای مورد استفاده و ابزارهای لازم برای شکستن متدولوژی ارائه شـده اسـت و سعی شده است که اجزای لازم برای مهندسی متدولوژی در سیستم‌های مبتنی بر عامل مشخص گردند. در این تحقیقات متدولوژی‌هـای MaSE [۵۰]، Gaia [۵۱]، Premetheous [۵۲] و PASSI [۵۳] بـه تعـدادی اجـزا شکسـته شـده اسـت و براساس آن انباره اجزای متدولوژی برای متـدولوژی‌هـای مبتنـی بـر عامـل تهیـه شده است. بدین ترتیب یک متدولوژی جامع براساس این دیدگاه ارائه شـده اسـت [۵۴].

در تحقیق دیگری که با هـدف مهندسـی متـدولوژی انجـام شـده اسـت، یـک فرامدل به عنوان زیربنایی برای بازنمایی جهت یکسان‌سازی متدولوژی‌های موجود و حرکت به سمت یک توصیف واحد از متدولوژی ارائه شده است. در ایـن تحقیـق در یک فرامدل مفاهیم مرتبط با طراحی و زمان اجرا و طراحـی داخلـی عامـل بـه صورت مدل کلاس بازنمایی شده است [۵۵].

در تحقیـق دیگـری کـه در زمینـه method Fragment انجـام شـده اسـت، تحقیقات FIPA برای استانداردسازی مورد بررسی قرار گرفته است و براسـاس آن، اجزای اصلی متدولوژی‌ها مبتنی بر چندین دیدگاه (شامل فرآیند، استفاده مجـدد، ذخیره‌سازی و پیاده سازی) مشخص شده است [۵۶].

۴-۸- تحلیل تحقیقات انجام شده در زمینه متدولوژی‌ها و فرآیندها

در زمینه متدولوژی‌های تولید سیستم‌های مبتنـی بـر عامـل، در ایـن تحقیـق ۳۸ مقاله گزارش شده است. نیمی از این تحقیقات مربوط به سال‌های قبـل از ۲۰۰۵ و نیمی از آن مربوط به سال‌های پس از ۲۰۰۵ است. با بررسی این مقالات مـی‌تـوان نتیجه گرفت که در سال‌های اولیه (۲۰۰۰ تا ۲۰۰۵) حجـم بـالایی از مقـالات بـه معرفی متدولوژی‌های جدید و توسعه و بهبود آنها اختصاص داشـته اسـت. ایـن در حالی است که در سال‌های بعد بحث ارزیابی متدولوژی‌ها بیشتر موردتوجه محققان قرار گرفته است. این در حالی است که مباحـث مـرتبط بـا Method Fragmen و استفاده مجدد از اجزای متدولوژی و حرکت به سمت استفاده از متـدولوژی‌هـا بـه صورت تلفیقی براساس نیازهای سیستم بیشتر مورد توجه قرار گرفته اسـت. بـدین ترتیب می‌توان نتیجه گرفت که در زمینه متدولوژی‌های مهندسی نرم‌افـزار مبتنـی بر عامل تحقیقات از تنوع و تکثر روش‌ها در سال‌ها اولیه به سمت استانداردسازی و تلفیق متدولوژی‌ها در حرکت است. علاوه بر اینکـه دیـدگاه‌هـای خـاص در تولیـد متدولوژی که منجر به ارائه شیوه‌های نو یا توجه بـه مسـایل خـاص در متـدولوژی می‌گردد، هنوز نیز زمینه تحقیقاتی مناسبی است.

همچنین در این بخش، فرآیندهای مهندسی نرم‌افزار مبتنی بر عامـل نیـاز بـه تحقیقات گسترده‌ای دارند. چنانکه نتایج این تحقیق نیز نظرات ارائه شده در [۴۸] را تایید می‌نماید، فرآیندهای مهندسی نرم‌افزاردر سیسـتم‌هـای مبتنـی بـر عامـل چندان مورد توجه قرار نگرفته‌اند (در این تحقیق ۲ مقاله از ۳۸ مقاله که در حدود ۵ درصد از تحقیقات انجام شده است، به موضـوع فرآینـدهای مهندسـی نـرم‌افـزار پرداخته است.)

در میان متدولوژی‌هایی که در این زمینـه ارائـه شـده‌انـد، متـدولوژی MaSE بیشترین حجم تحقیقات را (۷ مورد از ۳۸ که در حدود ۱۸ درصـد کـل تحقیقـات مورد بررسی در این زمینه در این گزارش می‌باشد) در زمینـه توسـعه و بهبـود بـه خود اختصاص داده است. این موضوع را می‌توان تاییدی بر ارزیابی انجـام شـده در [۴۷] دانست که موجب توجه بیشتر محققان به این متدولوژی شده است.

۵- مدل‌سازی در سیستم‌های مبتنی بر عامل

استفاده از زبان مدل‌سازی UML یکی از پرکاربردترین روش‌هـای مـدل‌سـازی در سیستم‌های مبتنی بر عامل است. براساس نتایج حاصـل از تجربیـات اولیـه در بـه کارگیری UML، نیاز به توسعه آن بـرای مـدل‌سـازی ویژگـی‌هـای خـاص عامـل مشخص شد. با استفاده از قابلیت‌های زبان UML برای توسعه، تحقیقات گوناگونی در زمینه توسعه آن برای کاربرد در مهندسی نرم‌افزارسیستم‌های مبتنی بـر عامـل صورت گرفته است.

بنابر روش تحقیقی که در بخش ۲ تشریح گردید، مجموعه مقالاتی کـه در ارتبـاط با بحث مدل‌سازی جمع‌آوری شده‌اند، را می‌توان درموضوعات زیر طبقه‌بندی نمود:

- استفاده از زبـان مـدل‌سـازی UML و ارائـه توسـعه‌هـایی بـر آن جهـت مدل‌سازی ویژگی‌های عامل و معرفی AUML
- استفاده از زبان مدل سازی i* به عنـوان یکـی از روش‌هـای مـدل‌سـازی عامل‌ها و روابط اجتماعی آنها
- AML به عنوان یک زبان مدل‌سازی برمبنای UML
- ارائه چارچوب‌های خاص برای مدل‌سازی عامل

در ادامه تحقیقات انجام شده در زمینه مدل‌سازی سیستم‌های مبتنی بر عامـل تشریح می‌گردند.

۵-۱- UML و AUML

در تحقیقات اولیه مرتبط با مدل‌سازی سیستم‌های مبتنی بر عامـل، از نمودارهـای استاندارد UML استفاده می‌شد. در تحقیقی که در [۵۷] گزارش شده است، برای مدل‌سازی عامل به عنوان بالاترین سطح تجرید در سیستم‌های مبتنـی بـر عامـل، چهار نمودار واژگان شناختی، معماری، پروتکل و نقش پیشنهاد شده است. در ایـن مدل‌ها از نمودارهای UML استاندارد (به ترتیـب نمـودار کـلاس، نمـودار آرایـش ساختاری، نمودار همکاری و نمودار کلاس) برای بازنمایی این ویژگی‌های سیسـتم استفاده شده است.

نخستین تلاش‌های مستمر برای استفاده از زبان مدل‌سازی UML برای بازنمایی ویژگی‌های عامل توسط Odell و همکارانش صورت گرفته است [۵۸، ۵۹]. بنابر این پیشنهادها، توسعه نمودارهای ترتیب در UML جهت پشتیبانی قابلیت‌ها و مفاهیم مطرح در سیستم‌های مبتنی بر عامل در هنگام تعامل مانند نقش، رشته‌های تعاملی میان عامل‌ها، پروتکل‌های تعاملی تودرتو و توسعه مفاهیم پیام‌هایی که امکان تبادل آنها میان عامل‌ها وجود دارد، ارائه شده است [۶۰]. همچنین برای توسعه نمودارهای کلاس UML جهت پشتیبانی از ویژگی‌های خاص عامل، یک ساختار جدید برای کلاس‌های عامل پیشنهاد شده است. در این ساختار اقلام اطلاعاتی از قبیل نام عامل، وضعیت ذهنی که در آن ویژگی‌هایی از قبیل باورها، تمایلات و قصد عامل مشخص می‌شوند، عملیاتی که عامل می‌تواند انجام دهد، سرویس‌ها و توانایی‌هایی که عامل دارد، ساختار سازمانی عامل و رفتارهای عامل مشخص می‌شوند [۶۱].

در این زمینه همچنین برای مدل‌سازی ساختارهای سازمانی موجود میان عامل‌ها، توسعه‌هایی بر UML ارائه شده است. بدین ترتیب امکان بازنمایی ساختار اجتماعی در سیستم‌های مبتنی بر عامل فراهم شده است [۶۲].

بدین ترتیب توسعه‌هایی برای UML معرفی شدکه منجر به معرفی Agent UML (AUML) گردید. هدف از این زبان، مدل‌سازی کلیه ویژگی‌های عامل با استفاده از UML است. بنابر این پیشنهادها، امکان توصیف نقش در قالب نمودارهای ترتیب، افزودن عامل به عنوان یک واسط، توانایی حرکت میان سیستم‌های خودمختار مختلف و تغییر مدل‌های آرایش ساختاری سیستم، از جمله پیشنهادهای این تیم بوده است. مجموعه این پیشنهادها به تیم استانداردسازی UML ارائه شده است تا در UML2 به عنوان استاندارد جدید در نظر گرفته شود [۱۴].

نحوه استفاده و کاربرد نمودارهای مختلف ارائه شده در UML2 برای عامل و سیستمهای مبتنی بر عامل در [۶۳] ارائه شده است. ازمیان نمودارهای ساختاری که جنبه‌های ایستای سیستم را مدل‌سازی می‌نمایند، نمودار کلاس یکی از پرکاربردترین نمودارها در مدل‌سازی سیستم‌های مبتنی بر عامل است. از این نمودار می‌توان برای مدل‌سازی عامل، ساختار سلسله مراتبی نقش‌ها، مدل سرویس عامل، ساختار سازمانی، واژگان شناختی، ساختار اجتماعی دانش عامل استفاده کرد. علاوه بر این نمودار شی را می‌توان برای مدل‌سازی عامل‌ها در زمان اجرا مورد استفاده قرار داد. نمودار ساختار ترکیبی در UML2 نشان می‌دهد که چگونه اجزای مختلف معماری با هم ترکیب می‌شوند. بدین ترتیب می‌توان از این نمودار برای نمایش ساختار اجتماعی استفاده کرد. از نمودار اجزای ارائه شده در این استاندارد می‌توان برای تعریف رفتارهای ورودی و خروجی هر وظیفه و برای نمایش شکستن معماری سیستم نیز استفاده نمود. برای تعیین توزیع عامل‌ها به صورت فیزیکی می‌توان از نمودار آرایش ساختاری استفاده کرد. جنبه‌های پویای سیستم با استفاده از نمودارهای رفتای قابل مدل سازی است. با توجه به اصلاحاتی که در تعریف بازیگران سیستم در نمودار مورد کاربرد ارائه شده است، نسخه جدید این نمودار، برای مدل سازی سیستم‌های مبتنی بر عامل مناسبتر شده است به طوری که می‌توان بازیگران داخلی و خارجی سیستم را مشخص نمود. همچنین برای تعیین برنامه‌هایی که یک عامل از آن استفاده می‌کند، می‌توان از نمودار فعالیت استفاده نمود. ولی برای این کار، مدل‌های UML2 نیاز به توسعه دارند. همچنین برای اینکه امکان انتخاب میان برنامه‌های مختلف وجود داشته باشد، توسعه‌هایی برای این نمودار لازم است. نمودارهای ترتیب و حالت را می‌توان برای مدل‌سازی پروتکل‌های تعامل میان عامل‌ها مورد استفاده قرار داد. توسعه‌های ارائه شده در UML2 برای نمودارهای ترتیب، دربرگیرنده کلیه توسعه‌های پیشنهاد شده توسط FIPA است [۶۴]. اما در آن نقش به عنوان یک مفهوم درنظر گرفته نشده است. بنابراین برای مدل‌سازی سیستم‌های مبتنی بر عامل، در نظر گرفتن نقش به عنوان توسعه‌ای برای نمودار ترتیب (و همچنین نمودار ارتباط) لازم است.

یکی دیگر از توسعه‌هایی که برای UML ارائه شده است، امکان مدل‌سازی قابلیت حرکت یک عامل است. در این زمینه مسایلی همچون ایجاد عامل جدید، مسیر حرکت و وضعیت فعلی عامل درنظر گرفته شده‌اند و برای مدل سازی این ویژگیها چهار مدل مختلف (شامل Sterotype, Swimlaned, State representation, framefragement) ارائه شده است. این مدل‌ها را می‌توان توسعه‌هایی برای نمودارهای ترتیب UML درنظر گرفت. بنابر پیشنهاد ارائه شده در این تحقیق، هر یک از این مدلها امکان بازنمایی ویژگیهای منحصر به فرد یک عامل متحرک در محیط‌های خاص را دارا هستند. بنابراین لازم است که به نوع کاربرد هر یک از نمودارها با توجه به شرایط خاص سیستم توجه شود[۶۵].

مدل‌سازی رفتارهای عامل از قبیل همزمانی، انطباق پذیری و حرکت نیز با استفاده از توسعه نمودارهای پویای UML موردتوجه قرار گرفته است. بدین منظور با توسعه نمودارهای ترتیب و نمودار فعالیت، راهنمایی جهت مدل‌سازی ویژگی‌های مختلف عامل ارائه شده است. به عنوان نمونه پیشنهاد شده است که برای تعامل میان عامل‌ها از نمودار ترتیب برای مدل‌سازی استفاده شود. درحالی که استفاده از نمودار فعالیت در مدل‌سازی هدف‌گرایی، ارسال و دریافت پیام‌ها و انطباق اهداف مناسب‌تر است. در مواردی مانند رفتارهای اجتماعی، انطباق هدف، توزیعی بودن، قابلیت حرکت و اجرای همزمان می‌توان از هر دو نمودار استفاده کرد [۶۶].

۲-۵ i*

استفاده از مفاهیم و ویژگی‌های عامل برای مدل‌سازی ویژگی‌های موجودیت‌های سیستم (شامل افراد، نرم‌افزارها و سخت‌افزارها)، مبنای بسیاری از روش‌های مبتنی بر عامل است [۶۷]. در مدل i* که شاخص‌ترین مدل در این روش‌هاست، از مفاهیمی مانند هدف، تمایلات، ارتباط‌ها و توانایی‌های عامل استفاده می‌شود، تا آنچه برای هر یک از ذینفعان سیستم حائز اهمیت است، مشخص شود. بدین ترتیب، هر یک از موجودیت‌های سیستم (از قبیل ذینفعان، موجودیت‌های سخت افزاری، نرم‌افزاری، سیستم) در قالب تعدادی عامل دیده می‌شوند که ویژگی‌هایی از قبیل هدف، تمایلات، ساختارهای سازمانی که در آنها شرکت دارند، براساس این دیدگاه مدل می‌شوند و میزان انطباق، سازگاری و تضادهای آنها با یکدیگر مشخص می‌شوند [۶۸]. در چارچوب i* دو نوع مدل وجود دارد. مدل ارتباطات استراتژیک که بازیگران سیستم، ارتباط میان آنها و وابستگی‌های آنها با یکدیگر را نشان می دهد. در مدل‌های عقلانیت استراتژیک، اهداف اصلی بازیگران و شکستن آنها به اهداف کوچکتر و نحوه ارتباط و وابستگی‌های این اهداف مدل می شود. با استفاده از این دو مدل در کنار هم، یک دیدگاه جامع از سیستم ارائه می‌گردد.

روش i* مبنای متدولوژی‌های مختلفی قرار گرفته که مهمترین آنها Tropos است. مدل‌سازی و تحلیل سیستم در این متدولوژی‌ها براساس مفاهیم و نمودارهای ارائه شده در i* انجام می‌شود. در [۶۹]، شش روش که i* را مبنای مدل‌سازی و تعریف نیازمندی‌های خود قرار داده‌اند انتخاب و براساس معیارهای مختلفی مقایسه شده‌اند. این معیارها عبارتند از: معیارهای فرآیند، میزان توضیحات و راهنماهای ارائه شده برای آنها، منابع معرفی شده مانند محصولات میانی و فرآورده‌های تولید شده، مباحث مطرح در i* مانند Constructorهای مورد استفاده و میزان گسترش‌پذیری.

همچنین در تحقیق دیگری که در زمینه این چارچوب مدل‌سازی صورت گرفته است، مفهوم و ویژگی ارثبری در مدل‌های i* معرفی شده است. براساس نتایج این تحقیق، سه ایده اصلی توسعه، اصلاح و تعریف مجدد معرفی و نحوه

اثرگذاری هر یک از آنها بر اجزای اصلی سیستم و مدل‌سازی مشخص شده است [۷۰].

۵-۳- AML

زبان مدل‌سازی عامل (AML) یک زبان گرافیکی برای توصیف، مدل‌سازی و مستندسازی سیستم‌هایی است که در آنها مفاهیم موردنیاز براساس تئوری سیستم‌های چندعامله درنظر گرفته شده است. این زبان توسعه‌ای بر UML2 منطبق بر چارچوب مدل‌سازی OMG است. AML مفاهیم موردنیاز برای مدل‌سازی ویژگی‌های مختلف سیستم‌های چندعامله مانند واژگان شناختی، موجودیت‌های سیستم چندعامله، ویژگی‌های اجتماعی، تجرید در رفتار و تعامل میان عاملها، جنبه‌ها و مفاهیم ذهنی برای مدل‌سازی ویژگی‌های ذهنی عامل را ارائه می‌نماید [۷۱].

برای مدل‌سازی ویژگی‌های اجتماعی سیستم‌های مبتنی بر عامل، مدل‌سازی ساختار اجتماعی با معرفی موجودیتهای اجتماعی، نقش‌های موجودیت‌ها، روابط اجتماعی، ارتباطات و ویژگی‌های نقش صورت می‌گیرد. علاوه بر این رفتار اجتماعی با در نظر گرفتن جنبه‌های پویای اجتماعی بودن، تعاملات اجتماعی و فعالیت‌های اجتماعی مدل می‌شوند. بدین منظور توسعه‌هایی برای UML ارائه شده است [۷۲].

۵-۴- سایر روش‌ها

علاوه بر زبان‌های مدل سازی که در بخش‌های قبل عنوان گردید، روش‌های دیگری نیز در مدل‌سازی سیستم‌های مبتنی بر عامل وجود دارد. این روش‌ها با استفاده از ایده‌های موجود در شی‌گرایی یا با استفاده از زبان‌های خاص به ارائه چارچوب‌هایی پرداخته‌اند که براساس آنها امکان مدل‌سازی عامل و ویژگی‌های مبتنی بر عامل وجود دارد. نکته قابل توجه درباره کلیه این روش‌ها آن است که معمولا کاربردهای خاص دارند و به اندازه زبان‌هایی که در بخش‌های قبل معرفی شده‌اند، عمومیت ندارند.

از آنجایی که ممکن است موجودیت‌هایی در سطوح مختلف تجرید در یک سیستم مورد استفاده قرار گیرند، یکی از مسایل موجود نحوه تعامل میان این موجودیت‌هاست. برای مدل‌سازی نحوه ارتباط عامل و شی در یک سیستم اطلاعاتی، روش مدل‌سازی AOR پیشنهاد شده است. هدف از این مدل‌سازی تسهیل انتقال اطلاعات میان موجودیت‌های مختلفی است که در طراحی سیستم‌های اطلاعاتی به کار گرفته می‌شوند. در این روش، عامل‌ها از دو دیدگاه داخلی و خارجی مدل می‌شوند. دیدگاه خارجی، دیدگاه یک ناظر خارجی است که در آن تعاملات سیستم میان موجودیت‌های مختلف سیستم مدل می‌شوند. در حالی که دیدگاه داخلی، دیدگاه یک عامل است که در آن محیط براساس وضعیت ذهنی عامل مدل می‌شود. برای هر یک از این دیدگاه‌ها مدل‌های قالب، ترتیب و الگو ارائه شده است [۷۳، ۷۴].

یکی دیگر از روش‌هایی که از روش‌های مدل‌سازی موجود در شی‌گرایی ایده گرفته است، OPM/MAS نامیده می‌شود. این روش توسعه‌ای بر متدولوژی شی- فرآیند است که برای سیستم‌های مبتنی بر عامل براساس مفاهیم معرفی شده در متدولوژی Gaia به روزرسانی شده است. ایده اصلی در این روش آن است که مدل‌سازی سیستم می‌تواند در یک نمودار واحد با درنظر گرفتن موجودیت‌ها و فرآیندهای موجود در سیستم صورت گیرد. موجودیت‌ها شامل سازمان، اجتماع، Platform ، قانون، نقش، کاربر، پروتکل، باور، تمایل، واقعیت، هدف، قصد، و سرویس است. درحالی که فرآیند شامل عامل، وظیفه و پیام‌رسانی است [۷۵].

روش دیگری که از متدولوژی شی- فرآیند ایده گرفته است، امکان مدل‌سازی سیستم در سطوح مختلف تجرید را فراهم می‌کند و این روش در متدولوژی SODA اعمال شده است [۷۶].

در تحقیق دیگری که در آن ارتباط میان عامل‌ها از طریق تعریف نقش صورت می‌گیرد، چارچوبی با عنوان BRAIN (Behavioral Role for Agent Interactions) ارائه شده است. در این مدل لایه‌ای، مدل نقش، مدل‌سازی تعامل میان عامل‌های مختلف در سطح قابلیت‌های اصلی عامل، نمادهایی برای مدل‌سازی نقش براساس XML و زیرساخت تعامل ارائه شده است [۷۷].

مدل‌سازی دانش موردنیاز عامل در زمان اجرا موضوع تحقیقی است که در [۷۸] گزارش شده است. در این تحقیق، یک مدل سلسله مراتبی از دانش که لایه‌های پایین آن در داخل سیستم و لایه‌های بالاتر در اختیار عامل‌ها قرار میگیرد، ارائه شده است. بدین منظور دانش کسب و کار به سه دسته دانش فرآیند، دانش قوانین و دانش مفاهیم تقسیم شده است. بر این اساس، دو مدل مفهومی برای تعریف کلمات عامل و مدل واقعیت‌ها با استفاده از XML بازنمایی شده است.

برای بهبود مدل‌سازی سیستم‌های مبتنی بر عامل، چارچوب مدل‌سازی دیگری با عنوان Conceptual Graph Model (CG Model) پیشنهاد شده است. تمرکز این چارچوب بیشتر بر روی مرحله بدست آوردن نیازمندی‌های اولیه با استفاده از به کارگیری مفاهیم مدل ارائه شده می‌باشد [۷۹]. در فرامدل ارائه شده در این تحقیق، مدل مورد کاربرد، مدل تراکنش، مدل‌سازی سناریوهای توصیف سیستم، صحت سنجی و ارزیابی مدل موجود و انتقال به مدل طراحی مطرح شده است.

در تحقیق دیگری که برای مدل‌سازی عامل‌ها گزارش شده است، از I/O Automata که یک زبان رسمی برای توصیف سیستم‌های توزیع شده و واکنشی است، برای توصیف عامل‌ها استفاده شده است. بدین منظور یک معماری شامل حسگر، کنترل کننده و اثرگذار برای عامل در نظر گرفته شده است. همچنین قابلیت حرکت به عامل اضافه شده و بخش‌های مختلف معماری با استفاده از این زبان مدل‌سازی شده است [۸۰].

چارچوب OperA یک چارچوب برای مدل‌سازی ساختارهای سازمانی برای سیستم‌های چندعامله است که دیدگاه‌های مختلف از جامعه عامل‌ها ارائه می‌نماید. در این چارچوب، مدل سازمانی، مدل اجتماعی و مدل تعامل برای عامل‌ها درنظر گرفته شده است که براساس آن تعاملات، روابط اجتماعی، قوانین و قابلیت‌های مذاکره مورد نیاز عامل‌هایی که به یک سازمان ملحق می‌شوند، مورد بررسی قرار می‌گیرد [۸۱]. براساس این چارچوب، متدولوژی ارائه شده است که سناریوهای موردنیاز یک سازمان برای مدل‌سازی شرایط سازمانی (از قبیل اهداف و استراتژی سازمان و اهداف و نیازهای کاربران) را پوشش دهد [۸۲].

۵-۵- سایر تحقیقات مرتبط با مدل‌سازی

یکی از تحقیقاتی که در زمینه مدل‌سازی سیستم‌های مبتنی بر عامل، صورت گرفته است، به ترکیب روش‌های رسمی و غیررسمی توجه نموده است [۸۳]. در این تحقیق، روش غیررسمی *i با روش رسمی Cognitive Agent Specification Language (CASL) ترکیب شده است تا امکان توصیف سیستم‌های مبتنی بر عامل فراهم شود. برای این کار مجموعه‌ای از نمادهای میانی در قالب نمودارهای Intentional Annotation Strategic Rational (iASR) پیشنهاد شده است که فاصله میان این دو روش توصیف رسمی و غیررسمی را بپوشاند. هدف از انجام این کار ارائه افزایش دقت توصیف نیازمندی‌ها و افزودن

امکان بررسی و استنتاج اتوماتیک درباره اهداف تعیین شده برای عامل و دانش تخصیص یافته به آن (به عنوان حالت‌های ذهنی عامل) بوده است.

یکی دیگر از تحقیقاتی که در زمینه مدل‌سازی سیستم‌های مبتنی بر عامل صورت گرفته است، بررسی ویژگی‌های عامل مانند یادگیری، قابلیت حرکت، امنیت با دیدگاه جنبه‌گرایی است [۸۴]. چون این ویژگی‌ها بر قابلیت‌های یک عامل از قبیل عملیات و اهداف آن بسیار تاثیرگذار است، در این تحقیق با معرفی مدل عامل، مدل جنبه و مدل ترکیب، یک چارچوب فرا- مدل برای مدل‌سازی جنبه‌های مختلف عامل ارائه نموده است.

در مدل عامل ویژگی‌های عامل از قبیل هدف و اعمال عامل مدل‌می شوند. مدل جنبه، مفهومی است که مفاهیم، ارتباط‌ها و ویژگی‌های زبان‌های مدل‌سازی جنبه‌گرا را داراست. در مدل ترکیب، آنچه از جنبه‌های سیستم که ممکن است بر عامل، هدف یا اعمال اثرگذار باشد، مدل می‌شوند.

۵-۶- تحلیل تحقیقات انجام شده در زمینه مدل‌سازی

در زمینه مدل‌سازی سیستم‌های مبتنی بر عامل، می‌توان گفت که UML و توسعه‌هایی که برای آن ارائه شده‌اند، مبنای اصلی زبان‌های مدل‌سازی سیستم‌های مبتنی بر عامل هستند. به طوری که AUML و AML هر دو براساس این زبان بنا شده‌اند. در سایر روش‌ها و چارچوب های مدل‌سازی نیز معمولا از UML و توسعه‌های آن استفاده شده است تا امکان توصیف سیستم براساس دیدگاه‌های موردنظر فراهم شود. تنها چارچوب مدل‌سازی i* دارای ساختاری منحصر به فرد است که براساس مهندسی نیازمندی‌ها با استفاده از دیدگاه عامل بنا شده است و نمودارهای وابستگی استراتژیک و عقلانیت استراتژیک را ارائه نموده است.

با بررسی سیرتکاملی مدل‌سازی نیز می‌توان دید که تقریبا توسعه های UML و استفاده از i* برای مدل‌سازی در سال‌های اولیه معرفی مهندسی نرم‌افزار مبتنی بر عامل مورد استفاده قرار گرفته‌اند اما AML زبان جدیدی است که در سال‌های اخیر معرفی شده است.

تحقیقات انجام شده در سال‌های اخیر و سال‌های گذشته دارای درصد نسبتا یکسانی است. (در حدود ۴۵٪ از مجموعه مقالات جمع‌آوری شده در این زمینه مربوط به سال‌های قبل از ۲۰۰۵ و ۵۵٪ مربوط به سال‌های بعد از ۲۰۰۵) است. اگر چه بیشتر تحقیقات اولیه بر روی توسعه‌های UML متمرکز بوده است و در سال‌های اخیر معرفی زبان‌های مدل‌سازی خاص برای سیستم‌های مبتنی بر عامل بیشتر مورد توجه قرار گرفته است.

علاوه بر اینکه چارچوب‌های جدید با دیدگاه‌های خاص نیز در سال‌های اخیر موضوع تحقیقات بسیاری قرار گرفته است. به نظر می‌رسد با توجه به جدید بودن زبانی مانند AML، این زبان زمینه تحقیقاتی مناسبی برای سال‌های آینده باشد. هرچند که توسعه‌های UML در سالهای اخیر نیز همچنان مورد توجه هستند و امکان تحقیق بر روی این زبان مدل‌سازی هنوز وجود دارد.

۶- ابزارهای مهندسی نرم‌افزار

ابزارهای تولید نرم‌افزارهای مبتنی بر عامل را می‌توان به دو دسته اصلی تقسیم کرد.

۱) ابزارهایی که برای پیاده‌سازی سیستم‌های مبتنی بر عامل به کار می‌روند. این ابزارها هم به صورت تجاری توسط گروه‌های صنعتی ارائه می‌شوند و هم به صورت نتایج فعالیت‌های تحقیقاتی توسط پژوهشگران معرفی می‌شوند. نمونه‌هایی از این ابزارها در [۸۵] معرفی شده‌اند.

۲) ابزارهایی که به ابزارهای مهندسی نرم‌افزار معروفند و پشتیبانی تمام خودکار یا نیمه خودکار را برای فرآیند و متدولوژی‌های تولید نرم‌افزار مهیا می‌سازند. برای سیستم‌های مبتنی بر عامل، ابزارهای مهندسی نرم‌افزار یکی از زمینه‌هایی است که کمترین نتایج تحقیقات در آن گزارش شده است.

تحقیقات مرتبط با ابزارهای مهندسی نرم‌افزار مبتنی بر عامل معمولا به صورت جانبی در کنار مفاهیم ارائه شده گزارش می‌شوند و تحقیقات مستقل در زمینه تولید ابزارهای یکپارچه‌ای که فرآیندها، متدولوژی‌ها و روش‌های تولید نرم‌افزار را پشتیبانی نماید، بسیار اندک است.

در این بخش در مجموع دو مقاله به صورت مستقل تحقیقات مرتبط با تولید ابزارهای مهندسی نرم‌افزار را گزارش نموده‌اند. در [۸۶] به تبدیل اتوماتیک نمودار برنامه‌ریزی متدولوژی Tropos به نمودارهای فعالیت در UML2 پرداخته شده است. در این تحقیق، نحوه استفاده از تکنیک‌های تبدیل برای نگاشت میان مدل‌ها تشریح شده است و ابزاری براساس این دیدگاه معرفی شده است.

تحقیق دیگری به موضوع افزایش قابلیت استفاده از سیستم‌های مبتنی بر عامل برای افراد خبره در دامنه مساله، پرداخته است. در این تحقیق امکان تغییر و اصلاح کاربران در سیستم مبتنی بر عامل با استفاده از یک چارچوب مبتنی بر اجزای نرم‌افزاری ارائه و ابزاری برای پشتیبانی آن معرفی شده است و کارایی ابزار در برآوردن نیازهای کاربران ارزیابی شده است [۸۷].

از میان متدولوژی‌های مهندسی نرم‌افزار معرفی شده در بخش ۴، متدولوژی MaSE با ابزار agentTool یکی از نخستین متدولوژی‌هایی بوده است که توسط ابزارهای یکپارچه در کلیه مراحل تحلیل و طراحی پشتیبانی می‌شوند. متدولوژی Tropos با ابزار TAOM4E پشتیبانی شده است. برای تحلیل هدف در این مرحله ابزار GR-Tool و برای ملاحظات امنیتی ابزار ST-Tool ارائه شده است.

متدولوژی Prometheus از دیگر متدولوژی‌هایی است که با ارائه ابزار PDT مراحل طراحی را پشتیبانی نموده است. متدولوژی PASSI متدولوژی دیگری است که با ارائه یک ابزار با عنوان PTK (Passi Toolkit) امکان پشتیبانی مراحل مختلف در آن فراهم شده است.

در تحقیقات مرتبط با نیازمندی‌های سیستم‌های مبتنی بر عامل، ابزارهای اتوماتیک برای نیازمندی‌ها در متدولوژی Tropos با عنوان OME ارائه شده است. علاوه بر اینکه GR-Tool نیز برای پشتیبانی تحلیل هدف در این مرحله می‌تواند مورد استفاده قرار گیرد. علاوه بر این از میان تحقیقات مرتبط با تحلیل نیازمندی‌ها، قابلیت استدلال عامل با ارائه یک ابزار که امکان بررسی ویژگی‌های پویای سیستم در برابر Trace های تعریف شده را به صورت اتوماتیک فراهم می‌کند، پشتیبانی شده است [۸۸]. همچنین مفاهیم مرتبط با تحمل‌پذیری خطا به عنوان یکی از ویژگی‌های کیفی سیستم مبتنی بر عامل با استفاده از ابزار PLFaultCAT فراهم شده است [۸۹].

در تحقیقات مرتبط با تست نرم‌افزار، ابزار تست اتوماتیک برای فراهم نمودن امکان تست در متدولوژی Tropos گزارش شده است. این ابزار که eCAT نامیده شده است، مخفف Continouse Agent Testing on Eclipse می‌باشد [۹۰]. همچنین توسعه ابزار JADE برای پشتیبانی تولید عامل‌های ساختگی که امکان تست واحد را فراهم می‌نمایند، صورت گرفته است [۹۱].

در تحقیقات دیگری که در [۹۲] گزارش شده، برای تست واحد با استفاده از چارچوب junit چارچوب جدیدی با عنوان Sunit ارائه شده است. همچنین برای فراهم نمودن امکان تست در متدولوژی PASSI چارچوب و ابزاری برای اجرای آن ارائه شده است [۹۳]. بدین ترتیب می‌توان نتیجه گرفت که در تحقیقات مرتبط با تست سیستم‌های مبتنی بر عامل، پشتیبانی با استفاده از ابزار بسیار حائز اهمیت و مورد توجه محققان می‌باشد.

۷-تحلیل فضای تحقیق و نتیجه‌گیری

با گذشت بیش از یک دهه از معرفی مهندسی نرم‌افزار برای سیستم‌های مبتنی بر عامل، در این مقاله تحقیقات مرتبط با مهندسی نرم‌افزار مبتنی بر عامل- به عنوان رویکرد بعدی تولید نرم‌افزار- معرفی شده است. بدین منظور روش تحقیق برای جمع‌آوری مجموعه تحقیقات انجام شده در این زمینه تشریح شده است. با پیروی از روش تحقیق بیان شده، مجموعه‌ای از مجلات و کنفرانس‌های معتبر در این زمینه (در سال‌های ۱۹۹۹-۲۰۰۸) انتخاب و مقالات ارائه شده در آنها به عنوان منابع تحقیق گردآوری شده است. با دسته‌بندی تحقیقات در این زمینه، موضوعات مرتبط با متدولوژی‌ها و فرآیندهای مهندسی نرم‌افزار مبتنی بر عامل، مدل‌سازی و ابزارهای ارائه شده در این حوزه، در این مقاله مورد بررسی و تحلیل قرار گرفته است.

براساس بررسی‌های انجام شده می‌توان نتیجه گرفت که موضوعات مرتبط با متدولوژی‌های مهندسی نرم‌افزار، مدل‌سازی و معماری سیستم‌های مبتنی بر عامل در این حوزه بیشترین تحقیقات را به خود اختصاص داده‌اند. این درحالی است که موضوعات مرتبط با تست سیستم‌های مبتنی بر عامل که در سال‌های اخیر مورد توجه قرار گرفته است و ابزارهای پشتیبانی فعالیت‌های مهندسی نرم‌افزار به عنوان تحقیقات مستقل، کمترین سهم را از تحقیقات انجام شده داشته‌اند.

با بررسی تحقیقات انجام شده در زمینه متدولوژی‌ها و فرآیندهای مهندسی نرم‌افزار مبتنی بر عامل می‌توان نتیجه گرفت:

- تحقیقات جاری در این زمینه را می‌توان به دسته‌های زیر تقسیم کرد:
  - بهبود و توسعه متدولوژی‌های موجود مانند Gaia, MaSE, Tropos, PASSI
  - ارائه متدولوژی‌های خاص منظوره
  - ارزیابی متدولوژی‌های موجود
  - حرکت به سمت تلفیق و استفاده از متدولوژی‌ها به صورت شخصی با استفاده از روش‌های method Fragment
- تحقیقات از تنوع و تکثر روش‌ها در سال‌ها اولیه به سمت استانداردسازی و تلفیق متدولوژی‌ها در حرکت است.
- تحقیقات در زمینه فرآیندهای مهندسی نرم‌افزار مبتنی بر عامل به ندرت انجام شده است و نیاز به تحقیقات بیشتری در این زمینه وجود دارد.

با بررسی تحقیقات انجام شده در زمینه مدل‌سازی برای سیستم‌های مبتنی بر عامل می‌توان نتیجه گرفت:

- تحقیقات جاری در این زمینه را می‌توان به دسته‌های زیر تقسیم کرد:
  - استفاده از زبان مدل‌سازی UML و ارائه توسعه‌هایی بر آن جهت مدل‌سازی ویژگی‌های عامل و معرفی AUML
  - استفاده از زبان مدل‌سازی *i به عنوان یکی از روش‌های مدل‌سازی عامل‌ها و روابط اجتماعی آنها
  - AML به عنوان یک زبان مدل سازی برمبنای UML
  - ارائه چارچوب‌های خاص برای مدل‌سازی عامل
- UML و توسعه‌هایی که برای آن ارائه شده‌اند، مبنای اصلی زبان‌های مدل‌سازی سیستم‌های مبتنی بر عامل (مانند AUML و AML) هستند.
- اگرچه بیشتر تحقیقات اولیه بر روی توسعه‌های UML متمرکز بوده است و در سال‌های اخیر معرفی زبان‌های مدل‌سازی خاص برای سیستم‌های مبتنی بر عامل بیشتر مورد توجه قرار گرفته است.
- به نظر می‌رسد با توجه به جدید بودن زبانی مانند AML، این زبان زمینه تحقیقاتی مناسبی برای سال‌های آینده باشد.

با بررسی تحقیقات انجام شده در زمینه ابزارهای اتوماتیک و نیمه اتوماتیک مهندسی نرم‌افزار مبتنی بر عامل می‌توان نتیجه گرفت که:

- تحقیقات مستقل درباره ابزارهای تولید نرم‌افزارهای مبتنی بر عامل، سهم بسیار کمی در مقالات گزارش شده در این حوزه تحقیقاتی دارند.
- تقریبا تمامی متدولوژی‌های مهم در زمینه مهندسی نرم‌افزار مبتنی بر عامل، توسط یک ابزار اتوماتیک یا نیمه اتوماتیک پشتیبانی می‌شوند که فعالیت‌های مراحل مختلف پشتیبانی را پوشش می‌دهد.
- یکی از نقاط قوت تحقیقات مرتبط با متدولوژی‌ها، پشتیانی توسعه‌های ارائه شده برای متدولوژی به کمک ابزار (در قالب توسعه ابزار) می‌باشد.
- تست سیستم‌های مبتنی بر عامل، یکی از حوزه‌های بسیار قابل توجه در پشتیبانی توسط ابزار است. به طوری که تحقیقات مرتبط با تست سیستم‌های مبتنی بر عامل، اغلب همراه با یک ابزار تست ارائه شده است.

مراجع

[1] N. R. Jennings, "On Agent-based Software Engineering," *J. of Artificial Intelligence*, vol. 117, no. 2, pp. 277-296, 2000.

[2] F. Zambonelli, and A. Omicini, "Challenges and Research Directions in Agent-Oriented Software Engineering," *J. of Autonomous Agents and Multi-Agent Systems*, vol. 9, no. 3, pp. 253-283, 2004.

[3] A. Kay, "Computer Software," *J. of Scientific American*, vol. 251, no. 3, pp. 53–59, 1984.

[4] S. Russell, and P. Norving, Artificial Intelligence: A Modern Approach, Prentic Hall Series in Artificial Intelligence, Second Edition, 2003.

[5] H. V. D. Parunak, "A Practitioners' Review of Industrial Agent Applications," *J. of Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 4, pp. 389-407, 2000.

[6] R. Pressman, Software Engineering, a Practitioner's Approach, McGraw-Hill, Sixth Edition, 2005.

[7] I. Hadar, T. Kuflik, A. Perini, I. Reinhartz-Berger, F. Ricca, and A. Susi, "An empirical study of requirements model understanding: Use Case vs. Tropos models," In Proceedings of the 2010 ACM Symposium on Applied Computing (SAC '10). ACM, pp. 2324-2329, 2010.

[8] M. Cossentino, N. Gaud, V. Hilaire, S. Galland, and A. Koukam, "ASPECS: an agent-oriented software process for engineering complex systems," *J. of Autonomous Agents and Multiagent Systems Systems*, vol. 20, no. 2, pp. 260–304, 2009.

[9] S. A. DeLoach, and J. C. García-Ojeda "O-MaSE: a customisable approach to designing and building complex, adaptive multi-agent systems," *Int. J. Agent-Oriented Software Engineering*, vol. 4, no. 3, pp. 244–280, 2010.

[10] N. Spanoudakis, and P. Moraitis, "Using ASEME methodology for model-driven agent systems development," The 11th Int. Workshop on Agent Oriented Software Engineering (AOSE 2010), LNCS 6788, pp. 106-127, 2011.

[11] T. Miller, S. Pedell, L. Sterling, and B. Lu, "Engaging Stakeholders with Agent-Oriented Requirements Modelling," The 11th Intl Workshop on Agent Oriented Software Engineering (AOSE 2010), LNCS 6788, pp. 62-78, 2011.

[12] A. Sharpanskykh, and J. Treur, "Automated analysis of compositional multi-agent systems," *Int. Journal of Agent-Oriented Software Engineering (IJAOSE)*, vol.4, no.2, pp. 174 – 221, 2010.

[13] J. C. Garcia-Ojeda, and S. A. DeLoach Robby, "AgentTool Process Editor: Supporting the Design of Tailored Agent-based Processes," In Procedings of the 2009 ACM Symp on Applied Compting, ACM: New York, 2009.

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

[15] M. Wooldridge, and P. Ciancarini, "Agent-Oriented Software Engineering: The State of the Art," Agent-Oriented Software Engineering Workshop (AOSE00). Springer-Verlag LNCS1957, pp. 55-82, 2001.

[16] C. Bernon, M. Cossentiono, and J. Pavon, "An Overview of Current Trends in European AOSE Research," *Informatica*, vol. 29, pp. 379-390, 2005.

[17] S. Vafadar, and A. Abdollahzadeh Barfouroush, "Requirements Engineering for Agent Based Systems," Technical Report, CE/PR/88/01, Computer Engineering Faculty, Amirkabir University of Technology, 2008.

[18] M. Wooldridge, Reasoning about Rational Agents, The MIT Press: Cambridge, MA, 2000.

[19] F. Zambonelli, N. Jennings, A. Omicini, and M. Wooldridge, "Agent oriented Software Engineering for Internet Applications," Coordination of Internet Agents: Models, Technologies, and Applications, Chapter 13. Springer-Verlag.

[20] T. Juan, A. Pearce, and L. Sterling, "ROADMAP: Extending the Gaia Methodology for Complex Open Systems," The First International Joint Conference on Autonomous Agents & Multi-Agent Systems (AAMAS 2002), ACM Press, pp. 3-10, 2002.

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

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

[23] S. A. DeLoach, M. Wood, and C. Sparkman, "Multiagent Systems Engineering," *The Int. Journal of Software Engineering and Knowledge Engineering*, vol. 11 no. 3, 2001.

[24] J. DiLeo, T. Jacobs, and S. A. DeLoach, "Integrating Ontologies into Multiagent Systems Engineering," 4th international bi-conference workshop on agent- oriented Information systems (AOIS 2002), Italy, 2002.

[25] S. A. DeLoach, "Modeling Organizational Rules in the Multiagent Systems Engineering Methodology," Proceedings of the 15th Canadian Conference on Artificial Intelligence, USA, 2002.

[26] A. Self, and S. A. DeLoach, "Designing and Specifying Mobility within the Multiagent Systems Engineering Methodology," Proceedings of the Eighteenth Annual ACM Symposium on Applied Computing, (2003).

[27] S. Vafadar, A. Abdollahzadeh Barfouroush, and M. R. Ayatollahzadeh Shirazi, "Towards a More Expressive and refinable Multiagent System Engineering," 5th international bi-conference workshop on agent- oriented Information systems (AOIS 2003), LNCS 3030, pp. 126-141, 2003.

[28] S. A. DeLoach, "Multiagent Systems Engineering of Organization-based Multiagent Systems," 4th Int. Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'05), Springer LNCS vol. 3914, pp. 109 – 125, 2006.

[29] J. C. Garcia-Ojeda, S. A. DeLoach, Robby, W. H. Oyenan, and J. Valenzuela, "O-MaSE: A Customizable Approach to Developing Multiagent Development Processes," 8th Int. Workshop Agent-Oriented Software Engineering (AOSE 2007), LNCS 4951, Springer-Verlag, pp. 1-15, 2008.

[30] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini, "TROPOS: An Agent-Oriented Software Development Methodology," *J. of Autonomous Agents and Multi-Agent Systems*, vol. 8, no.3, pp. 203-236, 2004.

[31] P. Giorgini, J. Mylopoulous, and R. Sebastiani, "Goal-Oriented Requirements Analysis and Reasoning in the Tropos Methodology," *J. of Engineering Applications of Artificial Intelligence*, Elsevier, vol. 8, no.2, 2005.

[32] J. Jureta, S. Faulkner, and P. Schobbens, "Allocating Goals to Agent Roles During MAS Requirements Engineering," 7th Int. Workshop Agent-Oriented Software Engineering (AOSE 2006), LNCS 4405, Springer-Verlag, pp. 19-34, 2007.

[33] A. U. Mallya, and M. P. Singh, "Incorporating Commitment Protocols into Tropos," 6th Int. Workshop on Agent Oriented Software Engineering (AOSE 2005), LNCS 3950, pp. 69-80, 2005.

[34] J. Lind, Iterative Software Engineering for Multiagent Systems: The MASSIVE Method, LNCS 1994, Springer-Verlag, 2001.

[35] G. Caire, R. Evans, P. Massonet, W. Coulier, F. J. Garijo, J. Gomez, J. Pavón, F. Leal, P. Chainho, P. E.

[47] A. H. Elamy, and B. Far, "A Statistical Approach for Evaluating and Assembeling Agent Oriented Software Engineering Methodologies," 8th international bi-conference workshop on agent- oriented Information systems (AOIS 2006), pp. 105-122, 2006.

[48] L. Cernuzzi, M. Cossentino, and F. Zambonelli, "Process Models for Agent-based Development," *J. of Engineering Applications of Artificial Intelligence*, vol. 18, pp. 205–222, 2005.

[49] R. Basseda, A. Moallem, T. Alinaghi, and F. Taghiyare, "Estimation of Complexity in Agent Oriented Methodologies via Evaluation of Models and Artifacts," 12th computer society of Iran computer conference (CSICC2007), 2007.

[50] Q. N. Tran, B. Henderson-Sellers, and J. Debenham, "Incorporating the elements of the MASE methodology into Agent OPEN," In Proceedings of 6th Int. Conference on Enterprise Information Systems (ICEIS'2004), pp. 380-388, 2004.

[51] B. Henderson-Sellers, J. Debenham, and Q. N. Tran, "Adding Agent-Oriented Concepts Derived from GAIA to Agent OPEN," 16th International Conference of Advanced Information Systems Engineering., CAiSE 2004, LNCS 3084, Springer-Verlag, Berlin, pp. 98-111, 2004.

[52] B. Henderson-Sellers, Q. N. Tran, and J. Debenham, "Incorporating elements from the Prometheus agent oriented methodology in the OPEN Process Framework," Proceedings. Of AOIS@CAiSE04, pp. 370-385, 2004.

[53] B. Henderson-Sellers, J. Debenham, Q. N. Tran, M. Cossentino, and G. Low, "Identification of Reusable Method Fragments from the PASSI Agent-Oriented Methodology," 7th international bi-conference workshop on agent- oriented Information systems (AOIS 2005), pp. 90-97, 2005.

[54] B. Henderson-Sellers, "Creating a comprehensive agent-oriented methodology - using method engineering and the OPEN metamodel," in Agent-Oriented Methodologies (eds. B. Henderson-Sellers and P. Giorgini), Idea Group, pp. 368-397, 2005.

[55] G. Beydoun, C. Gonzalez-Perez, G. Low, and B. Henderson-Sellers, "Synthesis of a Generic MAS metamodel," 4th International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'05), Springer LNCS, vol. 3914, pp. 1-5, 2005.

[56] M. Cossentino, S. Gaglio, A. Garro, and V. Seidita, "Method fragments for agent design methodologies: from standardisation to research," *International Journal of Agent-Oriented Software Engineering*, vol. 1, no.1 pp. 91-121, 2007.

[57] F. Bergenti, and A. Poggi, "Exploiting UML in the Design of Multi-Agent Systems," Proceedings of the ECOOP - Workshop on Engineering Societies in the Agents' World 2000 (ESAW'00), pp. 96–103, 2000.

Kearney, and J. Stark, "Agent Oriented Analysis using MESSAGE/UML," LNCS 2222, Springer-Verlag, pp. 119-135, 2002.

[36] J. Pavón, and J. Gómez-Sanz, "Agent- Oriented Software Engineering with INGENIAS," 3rd International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS'03), LNCS 2691, Springer Verlag, pp. 394-403, 2003.

[37] C. Iglesias, M. Garijo, J. Gonzales, and J. R. Velasco, "Analysis and Design of Multi-agent Systems using MAS-CommonKADS," Proceedings of the Fourth International Workshop on Agent Theories, Architectures, and Languages (ATAL'97), LNCS 1365, Springer-Verlag, pp. 313–326, 1998.

[38] M. Cossentino, "From Requirements to Code with the PASSI Methodology," Agent-Oriented Methodologies, chapter IV, pp. 79—106, 2005.

[39] M. Winikoff, and L. Padgham, "The Prometheus Methodology," Federico Bergenti, Marie-pierre Gleizes and Franco Zambonelli, editors, Methodologies and Software Engineering for Agent Systems, Chapter 11. Kluwer Academic Publishing (New York), 2004.

[40] J. M. Serrano, and S. Ossowski, "On the Impact of Agent Communication Languages on the Implementation of Agent Systems," 8th International Workshop of Cooperative Information Agents (CIA 2004), LNCS 3191, Springer-Verlag, pp. 92-106, 2004.

[41] M. Gervais, "ODAC: An Agent-Oriented Methodology based on ODP," *J. of Autonomous Agents and Multi-Agent Systems*, vol. 7, no.3, pp. 199–228, 2003.

[42] ISO/IEC X.900 (1995). IS 10746-x ITU-T Rec. X90x, ODP Reference Model Part x.

[43] G. B. Roest, and N. B. Szirbik, "Intervention and Escape Mode: Involving Stakeholders in the agent development process," 7th Int. Workshop Agent-Oriented Software Engineering (AOSE 2006), LNCS 4405, Springer-Verlag pp. 109-120, 2007.

[44] M. Morandini, L. Penserini, A. Perini, and A. Susi, "Refining Goal Models by Evaluating System Behaviour," 8th Int. Workshop Agent-Oriented Software Engineering (AOSE 2007), LNCS 4951, Springer-Verlag, pp. 44-57, 2008.

[45] A. Bogdanovych, M. Esteva, S. Simoff, C. Sierra, and H. Berger, "A Methodology for Developing Multiagent Systems as 3D Electronic Institutions," 8th International Workshop Agent-Oriented Software Engineering (AOSE 2007), LNCS 4951, Springer-Verlag, pp. 103-118, 2008.

[46] G. Jayaputera, A. Zaslavsky, and S. Loke, "Approaches to Dynamically Generated User-Specified MAS," 6th Int. Workshop Agent-Oriented Software Engineering (AOSE 2005), LNCS 3950, Springer-Verlag, pp. 139-153, 2006.

[58] J. Odell, H. V. D. Parunak, and B. Bauer, "Extending UML for Agents," Agent-Oriented Information Systems (AOIS) Workshop at the 17th National conference on Artificial Intelligence (AAAI), 2000.

[59] J. Odell, and C. Bock, OMG document ad/99-12-01, "Suggested UML Extensions for Agents," Submitted to the OMG's Analysis and Design Task Force (ADTF) in response to the Request of Information (RFI) entitled "UML2.0 RFI," 1999.

[60] B. Bauer, J. Müller, and J. Odell, "Agent UML: A Formalism for Specifying Multiagent Interaction," The First International Agent-Oriented Software Engineering, AOSE 2000. Lecture Notes in Computer Science 1957, Springer-Verlag, pp. 91-103, 2001.

[61] B. Bauer, "UML Class Diagrams Revisited in the Context of Agent-Based Systems," Second International Workshop of Agent- Oriented Software Engineering, AOSE 2001, Lecture Notes in Computer Science 2222, Springer-Verlag, pp. 101-118, 2002.

[62] H. V. D. Parunak, and J. Odell, "Representing Social Structures in UML," In Proceedings of the fifth international conference on Autonomous agents, 2001.

[63] B. Bauer, and J. Odell, "UML 2.0 and agents: how to build agent-based systems with the new UML standard," *J. of Engineering Applications of Artificial Intelligence* vol. 18, pp.141–157, 2005.

[64] FIPA: Foundation for Intelligent Physical Agents, http://www.fipa.org/specs, 2005.

[65] M. Kusek, and G. Jezic, "Extending UML Sequence Diagrams to Model Agent Mobility," 7th International Workshop Agent-Oriented Software Engineering (AOSE 2006), Lecture Notes in Computer Science 4405, Springer-Verlag, pp. 51-63, 2007.

[66] V. Silva, R. Choren, and C. Lucena, "Using MAS-ML Dynamic Diagrams to Model MAS Behavioral Properties," 8th international bi-conference workshop on agent- oriented Information systems (AOIS 2006), pp. 9-16, 2006.

[67] E. Yu, "Why Agent Oriented Requirements Engineering," Proceedings of 3rd International Workshop on Requirements Engineering: Foundations for Software Quality (June 16-17, 1997, Barcelona, Catalonia). E. Dubois, A.L. Opdahl, K. Pohl, eds. Presses Universitaires de Namur, 1997.

[68] E. Yu, "Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering", Proceedings of the 3rd IEEE Int. Symp. on Requirements Engineering (RE'97) pp. 226-235, 1997.

[69] G. Grau, C. Cares, X. Franch, and F. J. Navarrete, "A Comparative Analysis of i* Agent-Oriented Modelling Techniques," Eighteenth International Conference on Software Engineering and Knowledge Engineering (SEKE'06), 2006.

[70] R. Clote, X. Franch, L. López, J. Marco, N. Seyff, and P. Grünbacher, "On the Meaning of Inheritance in i*," In Proceedings of the 17th Internacional Workshop on Agent-Oriented Information Systems (AOIS'07), published as part of the CAiSE'07 Proceedings of Workshops and Doctoral Symposium, vol. 2, pp. 651-665, 2007.

[71] R. Cervenka, I. Trencansky, M. Calisti, and D. Greenwood, "AML: Agent Modeling Language. Toward Industry-Grade Agent-Based Modeling," In Odell, J., Giorgini, P., Muller, J., eds.: Agent-Oriented Software Engineering V: 5th International Workshop, AOSE 2004, Springer-Verlag, 2005.

[72] R. Cervenka, I. Trencansky, and M. Calisti, "Modeling Social Aspects of Multi-Agent Systems -The AML Approach," International 6th International Workshop Agent-Oriented Software Engineering (AOSE 2005), Lecture Notes in Computer Science 3950, Springer-Verlag, pp. 40-53, 2006.

[73] G. Wagner, "The Agent-Object- Relationship Metamodel: Towards a Unified View of State and Behavior," *J. of Information Systems* 28 (5), pp. 475-504, 2003.

[74] G. Wagner, "Agent-Object-Relationship Modeling," Second International Symposium – from Agent Theory to Agent Implementation together with EMCRS, 2000.

[75] A. Sturm, D. Dori, and O. Shehory, "Single-Model Method for Specifying Multi-Agent Systems," Second International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2003), ACM Press, pp. 121-128, 2003.

[76] A. Omicini, "SODA: Societies and Infrastructures in the Analysis and Design of Agent-Based Systems," Agent-Oriented Software Engineering: First International Workshop, AOSE 2000. Lecture Notes in Computer Science 1957, Springer-Verlag, pp. 185-193, 2001.

[77] G. Cabri, L. Ferrari, and L. Leonardi, "Supporting the Development of Multi-Agent Interactions via Roles," 6th International Workshop Agent-Oriented Software Engineering (AOSE 2005), Lecture Notes in Computer Science 3950, Springer-Verlag, pp. 40-53, 2006.

[78] L. Xiao, and D. Greer, "A Hierarchical Agent-oriented Knowledge Model for Multi-Agent Systems," Eighteenth International Conference on Software Engineering and Knowledge Engineering (SEKE'06), 2006.

[79] R. Hill, S. Polovina, and M. D. Beer, "Improving AOSE with and Enriched Modelling Framework," 6th International Workshop Agent-Oriented Software Engineering (AOSE 2005), Lecture Notes in Computer Science 3950, Springer-Verlag, pp. 94-108, 2006.

[80] M. Bakhouya, and J. Gaber, "Autonomous Agent Modeling using I/O Automata," 7th International Workshop Agent-Oriented Software Engineering (AOSE 2006), Lecture Notes in Computer Science 4405, Springer-Verlag, pp. 157-168, 2007.

[81] V. Dignum, A Model for Organizational Interaction: based on Agents, founded in Logic. SIKS Dissertation Series 2004-1.PhD Thesis, Utrecht University, 2004.

[82] M. Mensonides, B. Huisman, and V. Dignum, "Towards Agent-Based Scenario Development for Strategic Decision Support," 8th international bi-conference workshop on agent-oriented Information systems (AOIS 2006), pp. 1-8, 2006.

[83] A. Lapouchnian, and Y. Lesperance, "Modelling Mental States in the Analysis of Multiagent Systems Requirements," 7th International Workshop Agent-Oriented Software Engineering (AOSE 2006), Lecture Notes in Computer Science 4405, Springer-Verlag, pp. 104-121, 2007.

[84] A. Garcia, C. Chavez, and R. Choren, "An Aspect Oriented Modelling Framework for Designing Multi-agent Systems," 7th International Workshop Agent-Oriented Software Engineering (AOSE 2006), Lecture Notes in Computer Science 4405, Springer-Verlag, pp. 35-50, 2007

[۸۵] ا. عبداله‌زاده بار فروش، ب. معصومی و م. ر. آیت‌اله‌زاده شیرازی، مقدمه‌ای بر هوش مصنوعی توزیع شده (معرفی عامل و سیستم‌های چندعامله)، انتشارات جلوه، تهران، ۱۳۸۴.

[86] A. Perini, and A. Susi, "Automating Model Transformations in Agent-Oriented modeling," 6th International Workshop Agent-Oriented Software Engineering (AOSE 2005), Lecture Notes in Computer Science 3950, Springer-Verlag, pp. 167-178, 2006.

[87] G. B. Jayatilleke, L. Padgham, and M. Winikoff, "Model Driven Development Toolkit for Domain Experts to Modify Agent Based Systems," 7th International Workshop Agent-Oriented Software Engineering (AOSE 2006), Lecture Notes in Computer Science 4405, Springer-Verlag, pp. 25-36, 2007.

[88] T. Bosse, C. M. Jonker, and J. Treur, "Requirements Analysis of an Agent's Reasoning Capability," Proceeding of the 7th Intenational Workshop on Agent-Oriented Information Systems, 7th international bi-conference workshop on agent- oriented Information systems (AOIS 2005), 2005.

[89] J. Dehlinger, and R. R. Lutz, "PLFaultCAT: A Product-Line Software Fault Tree Analysis Tool. Automated Software Engineerin," 13(1): 169-193, 2006.

[90] D. C. Nguyen, A. Perini, and P. Tonella, "A Goal-Oriented Software Testing Methodology," 8th International Workshop Agent-Oriented Software Engineering (AOSE 2007), Lecture Notes in Computer Science 4951, Springer-Verlag. pp 58-72, 2008

[91] R Coelho, U. Kulesza, A. von Staa, and C. Lucena "Unit testing in multi-agent systems using mock agents and aspects." Proceedings of the 2006, international workshop on Software engineering for large-scale multi-agent systems, pp. 83–90, ACM Press, 2006.

[92] A. M. Tiryaki, S. Oztuna, O. Dikenelli, R. Erdur, "Sunit: A unit testing framework for test driven development of multi-agent systems," In 7th International Workshop on Agent Oriented Software Engineering, 2006.

[93] G. Caire, M. Cossentino, A. Negri, A. Poggi, and P. Turci, "Multi-agent systems implementation and testing," International symposium – From agent theory to agent implementation, AT2AI, 2004.

احمد عبداله‌زاده بارفروش همکاری خود را با دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیرکبیر در سال ۱۳۷۰ و پس از اخذ مدرک دکتری مهندسی کامپیوتر از دانشگاه بریستول انگلستان، آغاز نموده است و هم‌اکنون استاد دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیرکبیر می‌باشد. ایشان از سال ۱۳۷۹ الی ۱۳۸۱ به عنوان استاد مدعو در دانشگاه‌های Maryland College Park آمریکا و ORASY (Paris 11) فرانسه و در سال ۱۳۸۸ تا ۱۳۹۰ به عنوان استاد مهمان در دانشگاه‌های Trento ایتالیا و Loughborough انگلستان مشغول به کار بوده است. دکتر عبداله‌زاده کتاب‌های "مقدمه‌ای بر هوش مصنوعی توزیع شده" و "کلیات متدولوژی تامین کیفیت" را نیز تالیف نمود. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: تکنیک‌های هوش مصنوعی، هوش مصنوعی توزیع شده، مذاکره خودکار، سیستم‌های خبره، پردازش زبان طبیعی، سیستم‌های تصمیم‌یار، هوش تجاری، پایگاه داده تحلیلی، داده‌کاوی، و مهندسی نرم افزار

آدرس پست‌الکترونیکی ایشان عبارت است از:

ahmadaku@aut.ac.ir

شیوا وفادار تحصیلات خود در رشته مهندسی کامپیوتر (نرم‌افزار) را در مقاطع کارشناسی و کارشناسی ارشد در دانشگاه‌های علم و صنعت ایران و صنعتی امیرکبیر به پایان رسانده و در حال حاضر دانشجوی دکتری این رشته در دانشگاه صنعتی امیرکبیر است. نامبرده در سال ۱۳۸۸ به عنوان دانشجوی مهمان در مرکز تحقیقاتFBK دانشگاه Trento ایتالیا مشغول به فعالیت بوده است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: مهندسی نرم‌افزار، مهندسی نرم‌افزار مبتنی بر عامل، مهندسی هوش برای سیستم‌های نرم‌افزاری، الگوهای تحلیل و نیازمندی در سیستم‌های نرم‌افزاری و سیستم‌های هوشمند.

آدرس پست‌الکترونیکی ایشان عبارت است از:

vafadar@aut.ac.ir

اطلاعات بررسی مقاله:

تاریخ ارسال: ۸۸/۵/۳۱

تاریخ اصلاح: ۹۱/۳/۱۳

تاریخ قبول شدن: ۹۱/۸/۱۵

نویسنده مرتبط: دکتر احمد عبداله‌زاده بارفروش، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، ایران.