معماری مدل رانه، راه‌حل چالش‌های یکپارچه‌سازی در سامانه‌های فوق مقیاس وسیع

ابراهیم علایی[1]
گروه کامپیوتر، دانشکده علوم پایه دانشگاه شهرکرد
ebrahimalaee@gmail.com

چکیده

با گسترش دنیای ارتباطات و نیاز روزافزون به سیستم های نرم افزاری، مشکلات تولید نرم افزار اعم از طول مدت توسعهٔ نرم افزار با وجود سکوهای مختلف سخت افزاری و نرم افزاری و نبود معماری منسجم نرم افزاری برای تولید و به روزرسانی در سیستم های نرم افزاری فوق مقیاس وسیع چالش هایی جدی است که فرا روی ترسیم نرم افزار می باشد. معماری مدل رانه یک راهکار برای چالش های یکپارچه سازی در سامانه های فوق مقیاس وسیع است که توانسته تا حدی این چالش ها را برطرف نماید. در این مقاله سامانه های مقیاس وسیع را همراه با ذکر ویژگی های آنها و چالش های موجود در این گونه سیستم ها معرفی کرده و مشکلات بالقوه را شناسایی کرده و همچنین معماری مدل رانه را معرفی و به بررسی خصوصیات این معماری برای فائق آمدن بر چالش ها و راه حل سیستم های فوق مقیاس وسیع می پردازیم.

واژه‌های کلیدی

سامانه های فوق مقیاس وسیع (ULS)، معماری مدل‌رانه (MDA)، یکپارچه سازی، چالش ها، معماری نرم‌افزار.

۱-معرفی سامانه‌های فوق وسیع (ULS)[1]

در توصیف سامانه های فوق مقیاس وسیع از لحاظ گستردگی و مقیاس می توان چنین گفت که: اندازهٔ چنین سامانه‌هایی از هر نظر بسیار فراتر از مقیاس سامانه های امروزی است، از نظر تعداد خطوط کد برنامه، افراد درگیر در سامانه، داده های ذخیره شده و دستکاری شده و پالایش شده؛ میزان اتصالات و وابستگی بین واحدی مؤلفه های نرم افزاری، عناصر سخت افزاری و ...[1].

مقیاس در این سامانه ها خود باعث بوجود آمدن مواردی از قبیل: تغییرات سریع در کارکرد سامانه و نیازهای کاربران و همچنین بوجود آمدن خرابی های سخت افزای و نرم افزاری متعدد و مستمر می شود. روش های مهندسی نرم افزار امروزه به هیچ وجه جوابگوی چنین سیستم های مقیاس وسیع نمی باشد. یکی از مؤسساتی که توانسته در این راه گام های بزرگی از جمله شناسایی چالش ها و ارائه راهکارها بردارد، مؤسسهٔ تحقیقاتی مهندسی نرم‌افزار واقع در دانشگاه کارنیگ ملون (SEI)[2] است که به سفارش وزارت دفاع آمریکا تحقیق ۱۲ ماهه خود را در قالب یک گزارش و کتاب ارائه داده است [8]. گروه مدیریت شیء (OMG)[3] با ارائه آخرین محصول خود که معماری مدل رانه (MDA)[4] می باشد، توانسته است راهکارهایی را مبتنی بر این استاندارد برای کمک به یکپارچه سازی سامانه های ULS ارائه کند.

۲- ویژگی‌های سامانه‌های فوق وسیع

سامانه های مقیاس وسیع علاوه بر وجود اندازه و مقیاس بزرگ دارای ویژگی های دیگری می باشند که باعث شده روش های مهندسی نرم‌افزار امروزه نتواند راه حل مناسبی برای این سیستم ها باشد. در این بخش چند ویژگی مهم این سامانه‌ها را ذکر می‌کنیم[1].

۲-۱- کنترل نامتمرکز: عدم وجود کنترل متمرکز روی کلیه ویژگی های این سامانه ها، یکی از مشکلات عمده می باشد. از جمله ویژگی هایی که می توان بیان نمود این است که تمام تضادها نمی توانند کامل و سازگار رفع شوند. یعنی برطرف نمودن برخی از تضادها خود در بوجود آوردن مشکلات و تضادهای دیگر نقش دارند.

۲-۲- متضاد بودن نیازمندی ها با همدیگر: سامانه های USL به دلیل ماهیتشان باید طیف وسیعی از نیازمندی ها را ارضاء

کنند و هر چقدر دامنهٔ این نیازمندی ها وسیع تر باشد، تنوع و تضاد بین آنها افزایش می‌یابد. همچنین، یکپارچگی راه حل‌ها نیاز به دانش در حوزه های مختلف و در نتیجه یک دانش بین دامنه‌ای دارد، که به دست آوردن آن چندان ساده نیست[1]. نیازمندی ها در سامانه های مقیاس وسیع از قبل به طور کامل مشخص و معین نیستند و همچنین با مقیاس اندازه موجود در این گونه سیستم ها رسیدن به نقطهٔ مصالحه بین ذینفعان که البته زیاد و نامعین هستند مشکل می باشد. تکامل و بهینه سازی تغییرات در این گونه سیستم ها نیز به طور پیوسته در حال جریان می باشد.

۲-۳- همگن نبودن اجزاء سیستم‌ها: داشتن کاربران و محیط های کاری و نیازمندی های مختلف و ناهمگن یکی دیگر از ویژگی های سامانه های مقیاس وسیع می باشد. همچنین تأثیرات تغییرات نمی تواند به قدر کفایت در سیستم ناهمگن مشخص شود و این تغییرات باید در پیکربندی اعمال و مدیریت شوند که این پیکر بندی نامعلوم و کنترل ناپذیر می باشد.

۲-۴- نبودن مرز بین افراد و سامانه: یک فرد ممکن است در این گونه از سیستم ها دارای نقش های متفاوتی از جمله کاربر بودن ، توسعه دهنده و یا نگهدارنده از سیستم را داشته باشد و کاربران سیستم کاملاً مشخص نیستند. همچنین افراد تنها کاربران سیستم نیستند و مرز مشخصی بین کاربران و سازندگان این گونه از سیستم ها معین نیست. رفتار جمعی افراد و تعاملات اجتماعی موضوعات مهم دیگری است که بیشتر باید مد نظر قرار گیرد و این خود باعث به وجود آوردن یک دیدگاه فنی اجتماعی[5] در مورد این سیستم ها می شود. درسامانه های ULS چگونگی استفادهٔ گروه کاربران از فناوری و نیز چگونگی پشتیبانی فناوری از نیاز های گروه کار محسوب می شود[1].

۲-۵- طبیعی بودن خرابی در این سیستم‌ها: در سامانه های مقیاس وسیع وجود خرابی در هر زمان طبیعی می باشد، یعنی به صورت پیوسته در حال رخ دادن خطا بوده و بر طرف نمودن خطاها نیز یک امر پیوسته می باشد. ولی هرگز نمی توان خطاها را به طور کامل در این سامانه ها برطرف نمود. بستگی به شرایط سیستم می توان از روش های خطایابی مختلف استفاده کرد.

۲-۶- سیاست گذاری مختلف و گاهی متضاد در توسعهٔ این سیستم ها: چون تنها یک گروه یا افراد خاصی نقش توسعه دهندهٔ این سیستم ها را ندارند و گروه های مختلف و بعضاً از ملل گوناگون با سیاست گذاری مختلف و گاهاً متضاد در توسعه این گونه از سیستم ها نقش دارند باعث به وجود آمدن این گونه از پارادیم ها می‌شود.

۳- چالش های سامانه های مقیاس وسیع:

می توان چالش های فرا روی توسعهٔ این سامانه ها را در سه حوزه کلی مورد بررسی قرار داد.

(۱) طرح و تکامل، (۲) هم نواسازی[6] و کنترل و (۳) نظارت و ارزیابی. [8]

۳-۱- چالش های حوزهٔ طراحی و تکامل: مواردی که به عنوان چالش و مشکلات مخصوص طراحی و تکامل برای توسعه دهندگان سامانه های مقیاس وسیع می تواند به عنوان سوأل در اذهان مطرح شود عبارتند از: چگونه می‌توان به شکل سیستماتیک ویژگی های اکوسیستمی فنی اجتماعی سامانه های ULS را برآورده کرد؟ چگونه می‌توان زیرساخت های اکوسیستمی را طراحی کرد؟ (که شامل: سرویس های فراهم شده و به اشتراک گذاشته شدهٔ یک سامانه، قواعد (رسمی و اجتماعی) هدایت رفتار آن سامانه؛ موارد با اهمیت اقتصادی و ... هستند.) چگونه می توان فرآیند های سازمانی مربوط به تولید و به روزرسانی مستمر مؤلفه های سامانه های ULS و یکپارچه سازی آنها را طراحی کرد؟[1]

در مورد این گونه سوألات، روش های طراحی و توسعه نمی تواند جوابگو باشد و باید به متد های مبتنی بر روش های جامع و فراگیر برای چالش های موجود در این حوزه دست یافت. همچنین نگهداشت یکپارچگی مفهومی که در طراحی چنین سامانه هایی مطرح می شود فراتر از فعالیت های طراحی امروزی است.

۳-۲- چالش های حوزهٔ هم نواسازی وکنترل: هم نواسازی یعنی مجموعه ای از فعالیت ها که مؤلفه های یک سامانهٔ ULS را درجهت ارضاء اهداف مأموریتی به شکل معقولی با یکدیگر هماهنگ کنند. هم نواسازی با مدیریت وکنترل سروکار دارد، اما در مقیاس های فراتر از کنترل سنتی، متمرکز و یک دست می باشد. هم نواسازی به ترکیبی از طراحی پیش رو، ترویج و اجرا، سیاست کلی و

تنظیم بلادرنگ پارامتر های عملیاتی نیازمند است[1] . هم نواسازی در همهٔ سطوح سیستم فوق مقیاس وسیع لازم و ضروری می باشد.

۳-۳- چالش های حوزه نظارت و ارزیابی: در این حوزه باید میزان اثر بخشی حوزه های قبلی را برآورد و اندازه گیری نمود. که روش های اندازه گیری مخصوص به خودی را در این سامانه ها لازم می باشد و روش های امروزی نمی تواند جوابگو باشد. در حین اندازه گیری باید فاکتور های زیادی از قبیل نیروی انسانی و فنی و اجتماعی بودن را در نظر داشته باشیم. همچنین معیار های موفقیت و سلامت کلی درسامانه های ULS بسیار متفاوت از سامانه های کوچک است. از جمله عواملی که در این حوزه چالش زا می باشند می توان به بزرگی مقیاس، عدم تمرکز، توزیع شدگی، و ناهمگنی سامانه های ULS اشاره کرد که اینها چالش های زیادی را پیش روی نظارت و ارزیابی مؤثر ایجاد می کنند.

۴- معماری مدل رانه (MDA)

معماری مدل رانه آخرین استاندارد گروه مدیریت شئ است. این معماری یک روش مدل سازی است که از آن می توان برای توسعه معماری سازمانی مختلفی در سیستم های با مقیاس هایی گوناگون استفاده نمود. MDA یک قدم بلند در راستای رسیدن به یک سازمان بلادرنگ است که در آن می توانند تغییرات مورد نیاز در معماری سازمان را انجام دهند و این تغییرات متعاقباً در کد لحاظ می گردد.[9]

۴-۱- دیدگاه در معماری مدل‌رانه

دیدگاه، یک سیستم روشنی برای تجرید می باشد. یک دیدگاه مجموعهٔ برگزیده ای از مفاهیم معماری و قوانین است و سطحی از تجرید را به خدمت می گیرد تا بروی مفاهیم خاصی از سیستم تمرکز کند. معماری مدل رانه دارای سه دیدگاه زیر است.

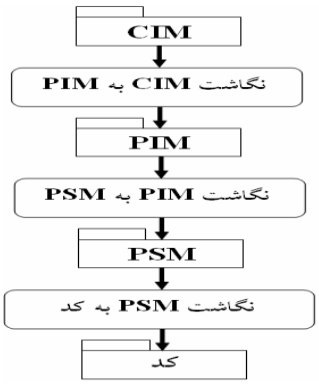
۴-۱-۱- دیدگاه مستقل از محاسبه: این دیدگاه کاری به محاسبات و کامپیوتر ندارد و بر روی خودِ حرفه و نیازمند های آن تمرکز دارد. همچنین جزییات ساختارها و فرایندهای سیستمی نادیده گرفته می‌شود[7].

۴-۱-۲- دیدگاه مستقل از سکو: این دیدگاه از دیدگاه قبلی نشأت گرفته است. یعنی این دیدگاه بر روی عملکرد یک سیستم تأکید می کند و در آن جزئیات لازم برای پیاده سازی روی یک سکوی خاص پنهان شده است. در واقع، این دیدگاه بخشی از مشخصهٔ کامل سیستم را نشان می دهد که از یک سکو به سکوی دیگر بدون تغییر باقی می ماند.[3]

۴-۱-۳ دیدگاه خاص سکو: این دیدگاه، دیدگاه قبلی را به همراه جزئیات پیاده سازی بر روی یک سکوی خاص نشان می دهد. همچنین می توان گفت که دید، مدلی از یک دیدگاه است که سیستم را از زاویهٔ یک دیدگاه خاص نشان می دهد[1]. برای هر کدام از دیدگاه های متفاوت از این معماری مدل هایی ایجاد می شود که به ترتیب می توان به آنها اشاره نمود: ۱- مدل مستقل از محاسبه[۷] (CIM) دیدی از سیستم که دیدگاه مستقل از محاسبه است و۲- مدل مستقل از سکو[۸] (PIM)دیدی از سیستم بر پایه دیدگاه مستقل از سکو است و ۳- یک مدل خاص سکو[۹] (PSM) دیدی از سیستم که بر پایه دیدگاه خاص سکو است.

۴-۲- نگاشت: یعنی تبدیل از یک نوع به نوع دیگر. مثلاً نگاشتی از یک CIM به یک PIM و یا از یک PIM به PSM. در زیر انواع نگاشت را بررسی می کنیم[2]. در شکل ۱ می توان تبدیلات بین سطوح مختلف مدل ها را در معماری مدل رانه مشاهده نمود.

۴-۲-۱- نگاشت های نوع مدل: "نگاشت نوع مدل" نگاشتی است از هر مدل ساخته شده با استفاده از انواع مشخص شده در زبان PIM (مثل کلاس و روابط بین آنها و سایر عناصر مدل) به مدل هایی که با استفاده از انواع مشخص شده در یک زبان PSM بیان می شود. یک PIM با به کارگیری یک زبان مدل سازی مستقل از سکو بیان می شود. معمار از عناصر این زبان مدل سازی برای ایجاد PIM با توجه به نیازهای سیستم استفاده می کند[2].



شکل ۱-فرآیند MDA [10]

۵- دلیل استفاده از معماری مدل رانــه در ســامانه های فوق وسیع

امروزه، استانداردهای زیادی برای مدل سازی ایجاد شده اند. این استانداردها تا حد زیادی از جنبهٔ علمی و هنری، دید مدل سازی ما را گسترش داده اند. برای یکپارچگی چنین استانداردهایی نیاز است قدرت این استانداردها از یک تعداد مدل ساده به سمت یک چرخه حیات کامل مبتنی بر مدل (MD)[10] افزایش یابد. هدف معماری مدل‌رانه رسیدن به چنین قدرتی برای استانداردهای مدل‌سازی است. راهکار MDAبرای انجام این کار شامل موارد زیر است:

- پذیرفتن فناوری‌های .Net، J2EE، XML،CORBA و ...
- بهبود قابلیت حمل برنامه ها به کمک امکان عینیت بخشی به یک مدل بر روی سکوهای مختلف. این کار توسط استانداردهای مختلف نگاشت انجام می‌شود.
- بهبود یکپارچه سازی بر اساس مدل های وابستگی بین حوزه های مختلف برنامه و واسط های مؤلفه ها. این کار باعث ایجاد تعامل پذیری بر پایهٔ وابستگی معنایی می شود.

معماری مدل رانه روشی را برای مشخصهٔ سامانه های اطلاعاتی معرفی می کند که در آن ، مشخصهٔ وظایف سامانه از مشخصهٔ پیاده سازی آن وظایف بر روی یک سکوی خاص فناوری، جدا شده است. برای انجام این کار MDA یک معماری را برای مدل ها تعریف می کند که در آن مجموعه ای از راهبردها برای سازماندهی مشخصه های بیان شده توسط مدل های مختلف فراهم شده است.

۵-۱- ویژگی های سامانه‌ها و راهکارهای معماری مدل رانه:

راهکارهای معماری مدل‌رانه می‌تواند در یکپارچه‌سازی سیستم‌های فوق وسیع با ویژگی‌هایی ذکر شده استفاده شود. در این سیستم‌ها نیازمندی‌ها در طول زمان مشخص شده و در حین توسعه، استقرار و نگهداری نیز به شدت متغیرند، لذا نیازمند معماری ای هستیم که انعطاف‌پذیر بوده و در کم‌ترین زمان با تغییرات جدید منطبق شود. با توجه به ویژگی‌های MDA که در بالا بیان شد، این جنبه از ULS ها در این نوع معماری پوشش داده می‌شود.

۱- **در سیستم‌های فوق وسیع در مسائل مختلفی عدم تمرکز وجود خواهد داشت**: دادهٔ نامتمرکز، تولید و توسعهٔ نامتمرکز، سیر تکاملی و کنترل عملیات غیرمتمرکز. همان‌طور که در بالا ذکر شد یکی از ویژگی‌های MDA فراهم ساختن استقلال در سطوح مختلف از فناوری تا کسب‌وکار و امنیت است. در MDA با استفاده از مدل‌های مجزایی که برای هر یک از موارد فوق درنظر گرفته می‌شود، تأثیر تغییرات کم‌ترین اثرات منتشر شونده را دربرخواهد داشت و کنترل غیرمتمرکز را پشتیبانی خواهد کرد.

۲- **تکامل و توسعهٔ پیوسته**: سیستم‌های فوق وسیع از جنبه‌های مختلف در حال تکامل هستند: اندازه، داده، کاربرد، کاربران و ... . و تغییرات عناصر بر عناصر دیگر تأثیرگذار است که بسیاری از تغییرات همروند با اجرای سیستم‌ها انجام می‌شود. یکی از مزایایی که MDA برای سازمان‌ها به وجود می‌آورد امکان ایجاد سازمان هایی است که به صورت بلادرنگ به تغییرات پاسخ دهند، این ویژگی در ULS ها نیز موردنیاز است و MDA می تواند پاسخ‌گوی آن باشد.

۳- **ناهمگونی، ناسازگاری و تغییر مؤلفه‌ها**: کنترل محدودهٔ مؤلفه‌های سخت‌افزاری و نرم‌افزاری امکان‌پذیر نیست. در این سیستم‌ها، محیط عملیاتی و مؤلفه‌ها تغییر می‌یابند. تأثیر تغییرات در این سیستم‌ها گاهی سریع و گاهی بسیار زمان‌بر است و تأثیر تغییرات قابل پیش‌بینی نیست. یکی از بهبودهایی که MDA به دنبال دارد بهبود یکپارچه‌سازی بر اساس مدل‌های وابستگی بین حوزه‌های مختلف برنامه و واسط‌های مؤلفه‌هاست. این کار باعث ایجاد تعامل پذیری بر پایهٔ وابستگی معنایی می‌شود که در ULS ها به شدت موردنیاز است.

۶- اهمیت معماری در سامانه‌های مقیاس وسیع

هدف اصلی از ایجاد معماری در سامانه‌های مقیاس وسیع، حل مسئلهٔ پیچیدگی می‌باشد. همچنین معماری در چنین سیستم‌هایی اصلی‌ترین دغدغه می‌باشد. سودمندی این عمل را می‌توان در چند مورد ذکر کرد :[14]

۶-۱ سهولت عمل طراحی هنگام تصمیم‌گیری:

یعنی امکان طراحی آسان سیستم مقیاس وسیع قبل از پیاده‌سازی آن، هنگام طراحی منطقی آن، در مرحله معماری سیستم. اگر چه طراحی و معماری کامل همراه با همهٔ ابعاد آن در زمان معماری اولیه امکان پذیر نیست اما می‌توان طرحی نزدیک به واقعیت (پیاده سازی عملی) را در درک چالش‌ها و مسائل ممکن پیش‌بینی نمود.

۶-۲- معماری در ULS یک انتزاع از آن سیستم است:

بوسیله این مزیت می‌توان گفت که ما چطور می‌توانیم از این سامانه‌های مقیاس وسیع تصور ذهنی دقیقی داشته باشیم[14]. که این سؤال یکی از چالش‌ها در این زمینه می‌باشد. داشتن انتزاع از این سامانه‌ها به ذینفعان آن در مورد درک مسائل و چالش‌های آن کمک

شایانی می‌نماید. همچنین داشتن تصور صحیح از سیستم ما را در مورد توسعهٔ آن با ابزارهای موجود و نحوهٔ توسعهٔ سریع بوسیلهٔ سطوح مختلف انتزاع کمک شایانی می‌نماید. مانند ایجاد مدل مستقل از سکو و تبدیل آن به مدل‌های خاص آن سکو در هنگام طراحی معماری قبل از پیاده‌سازی کامل آن سامانه مقیاس وسیع.

۷- نقش MDA در یکپارچه‌سازی سامانه‌های ULS:

در این بخش به بررسی مسئلهٔ اصلی این مقاله می‌پردازیم؛ بررسی چالش‌های یکپارچه‌سازی سامانه‌های فوق وسیع و ارائه راهکارهای مبتنی بر MDA برای آنها. همان‌طور که پیش ازاین گفتیم ،چالش‌های مختلفی فراروی توسعه سامانه‌های ULS وجود دارد. ابتدا چالش‌های یکپارچه‌سازی سامانه‌های فوق وسیع را ذکر کرده و در ادامه نشان می دهیم MDA چگونه می‌تواند در حل این چالش‌ها نقش داشته باشد. [1]همچنین به ارائه راهکارهایی که دیگران در این زمینه طرح نموده اند خواهیم پرداخت.

۷-۱- چالش اول: چگونگی داشتن درک صحیح از چالش‌ها و مشکلات سامانه‌های مقیاس وسیع برای همه ذینفعان آن سیستم، قبل از پیاده‌سازی آن به چه صورت می‌تواند انجام شود؟[16]

راهکار: داشتن درک صحیح از این سامانه‌ها ما را در طراحی هر چه بهتر و اندیشیدن برای رفع مشکلات آن سامانه‌ها برای پیاده‌سازی با دغدغهٔ کمتر کمک شایانی می‌نماید. این راهکار می‌تواند با داشتن یک معماری منسجم و جنبه‌های مستقل از پیاده‌سازی و تکنولوژی خاص (مستقل از سکو و محاسبات) ما را در حل این مشکل کمک کند [16]. البته معماری که برای همه ذینفعان آن سیستم گویا باشد و همگان بتوانند از آن در درک چالش‌ها و مشکلات فعلی و آیندهٔ سیستم مطلع شوند، به صورت صددرصد عملی نیست چون مقیاس و تغییرات مدام این اجازه را نمی‌دهد.

۷-۲- چالش دوم: صنعت نرم افزار (و بویژه توسعه سامانه‌های مقیاس‌پذیر) ویژگی‌های خاصی دارد که آن را از سایر صنایع جدا می کند. هرسال (حتی گاه سریعتر) فناوری‌های جدید ابداع می شوند و به سرعت رایج می شوند (برای مثال جاوا، لینوکس، J2EE, JSP ASP, UML, SOAP, HTML, XML, سرویس‌های وب و ...) در محیط های مقیاس وسیع بسیاری ازشرکت‌ها باید خود را با این تکنولوژی‌های جدید هماهنگ کنند و با آنها حرکت کنند. دربحث یکپارچه‌سازی، این وضعیت وخیم‌تر می شود. حتی اگر شما تنها روی نوع خاصی از تکنولوژی تمرکز کنید، برای تعامل با سایر نرم‌افزارها و یکپارچه‌سازی آنها نیازمند هماهنگی هستید. لازم به ذکر است که حتی یک تکنولوژی خاص نیز به سرعت تغییر می‌کند و نسخه های متعددی ایجاد می‌شوند که هیچ تضمینی برای سازگاری آنها با نسخه‌های قدیمی وجود ندارد و توسعه دهندگان معمولاً تنها دو یا سه نسخه قبلی را پشتیبانی می‌کنند. وجود مقیاس بسیار بالا در سامانه‌های ULS استفاده از طیف وسیعی از فناوری‌ها را اجتناب‌ناپذیر می‌سازد، چگونه می‌توان این فناوری‌ها رایکپارچه کرد؟[1]

راهکار: معماری مدل رانه با داشتن سطوح مختلف و مدل‌هایی که طی مراحلی به یکدیگر نگاشت می‌شوند و مستقل از همدیگر بوده و تغییراتی که در هر سطح از مدل‌های فرایند معماری مدل رانه رخ دهد به سطوح بالاتر تاثیر گذار نیست و مستقل از تغییرات می باشند. با وجود فناوری‌های جدید، در زمانی که یک تکنولوژی تغییر می کند مدل های PIM,CIM بدون تغییر باقی می‌ماند و تنها نیاز است مدل های PSM تغییرکنند. انجام این کار می‌تواند با ابزارهای خاص به طور خودکار انجام شود[1]. این کار در سطح تئوری بوده و عملیاتی شدن آن لازمهٔ پیشرفت این معماری می‌باشد و امید است که در آینده این راهکار عملی شود. بااین حال، قطعاً تغییر تنها مدل های PSM بسیار ساده‌تر وکم هزینه‌تر از تغییرکل نرم‌افزار است.

۷-۳- چالش سوم: فناوری‌های نوینی وجود دارند که هنوز به شکل کامل توسعه نیافته‌اند، اما در آیندهٔ نزدیک توسعه داده می‌شوند. سامانه‌های ULS چگونه می توانند با این فناوری‌ها یکپارچه شود؟[1]

راهکار:در مورد این چالش می‌توان گفت که معماری مدل رانه با داشتن ابزارهایی چون فرامدل‌ها و فراشئ که مستقل از فناوری هستند می‌تواند جوابگو باشد. به عنوان نمونه؛ ابزاری فرا شئ (MOF) است که در قسمت‌های قبلی توضیحاتی در مورد آن داده شده است. بنابر نقل از منبع شماره یک می‌توان این گونه بیان کرد: اگرتکنولوژی‌های جدید براساس استانداردهای MOF توسعه داده شوند، سایر نرم افزارهای پیرو MOF به راحتی می‌توانند با آنها تعامل برقرار نموده و به این شکل متحدکردن آنها بسیار ساده‌تر از زمانی است که هیچ فرامدل مشترکی بین آنها وجود نداشته باشد. همان‌طورکه اشاره کردیم، MOF هستهٔ استانداردهای MDA است و تمامی مدلها، فرامدلها، و استانداردهای MDA پیرو MOF هستند[1].

۷-۴- چالش چهارم: تولید و به‌روزرسانی مستندات در بهبود و نگهداری و یکپارچه‌سازی نرم‌افزارها مفید است. در توسعهٔ سامانه‌های

ULS با توجه به مقیاس بسیار بالایی که دارند، آیامستندات تولید می شود؟چگونه می‌توان مستندات را تولید و به‌روزرسانی کرد؟[1]

راهکار: مستندسازی یکی از کارهای اصلی در فرایند تولید نرم‌افزارها می‌باشد، ولی در اکثر تیم‌های پروژهٔ نرم افزاری دیدگاهی مختلف و ضعیف نسبت به این عمل وجود دارد و همواره به آن به عنوان یک کار جانبی نگاه می‌شود. چون تولید کد در نرم‌افزارها کاری مهم‌تر می‌باشد، بیشتر سعی در تولید کد می شود تا تولید مستندات پروژه. MDA می‌تواند این وضعیت را بهبود بخشد، در معماری مدل رانه تولید مدل ها و مستندات جزء وظیفه اصلی محسوب می‌شود و تولید کد تا حدامکان به طور خودکار انجام می شود. ابزارهای MDA این توانایی را دارندکه مدل‌های مختلف را به همراه مستندات آنها به خوبی نگهداری و یکپارچه سازند که نتیجه آن وجود اطلاعات کافی در زمان نگهداری، توسعه و یکپارچه سازی نرم افزاراست.[6] همچنین وجود ابزارهای نسل چهارم برای مستندسازی نیز می تواند راهکار دیگری باشد.

۷-۵- چالش پنجم: یکپارچه‌سازی نیاز به یک تغییر بهت عمده درسیاست‌های همکاری دارد. در سامانه‌های ULS به دلیل مقیاس بالا این ضرورت بیشتر احساس می شود. چگونه می‌توان سیاست‌های همکاری را تسهیل کرد؟[1]

راهکار: معماری مدل رانه در مدل‌های مستقل از سکو، خود با زبان های مشترک می‌تواند به تغییرات در سیاست‌ها و همکاری بین سیاست‌های مختلف فائق آید. همچنین در هنگام تولید کد بر طبق مدل خاص سکو می‌تواند برنامه را تولید نماید و برخی سیاست‌ها را در آن اعمال نماید. البته محدودیت‌های آن زبان خاص سکو نیز در این امر دخیل می‌باشد.

۷-۶ - چالش ششم: تلاش‌های یکپارچه‌سازی به دلیل وسعت حیطه عمل آنها (خصوصاً درسامانه های ULS) تأثیرات وسیعی روی کسب و کار می‌گذارند. هنگامی که مهمترین وظایف یک کسب‌وکار به شکل یک راه‌حل یکپارچه در آمدند، درست عمل کردن این راه‌حل برای کسب‌وکارحیاتی است. چگونه می توان قبل از پیاده‌سازی و یکپارچه‌سازی از صحت عملکرد آنها مطمئن شد؟[1]

راهکار: وجود ابزارهای شبیه‌سازی و مدل‌سازی در این معماری، خود می تواند یک راه‌حل باشد. همچنین به نقل از منبع شماره یک، این امکان وجود دارد که پس از تحلیل و طراحی مدل‌ها، آنها را اجرا کرد تا از عملکرد آنها مطمئن شد. تکنیک‌های مربوط به EXECUTABLE که در MDA مورد توجه قرار گرفته است این امکان را فراهم می‌کند که مدل‌ها مستقیماً اجرا و ارزیابی شوند به این شکل امکان تست تحلیل و طراحی به شکل مؤثری امکان‌پذیر است و می‌توان قبل از پیاده‌سازی و یکپارچه‌سازی از صحت آنها مطمئن شد.

۷-۷- چالش هفتم: علی‌رغم نیاز گسترده به راه‌حل‌های یکپارچه، تنها استانداردهای اندکی در این حوزه ایجاد شده است. این استانداردها نیز باهم هماهنگ نیستند و باعث آشفتگی درگسترش و تفسیر جدید استانداردها می‌شوند. چگونه می‌توان قابلیت همکاری و یکپارچه‌سازی را بین محصولات «پیرو استاندارد[11]» مختلف (محصولاتی که از استانداردهای خاصی تبعیت می کنند) افزایش داد؟

راهکار: وجود استانداردهای مختلف برای سکوهای متفاوت مانند AndroMDA که برای محیط‌های مبتنی بر جاوا فراهم شده، خود باعث این چالش شده است، چون در این معماری همهٔ محصولات باید از فرامدل‌ها پیروی کنند. می‌توان همهٔ مدل‌های استانداردها را توسط این معماری تعریف نمود البته با یک فرامدل که قبلاً باید در این مورد تعریف شده باشد تا بتوان از آنها در محیط‌های مختلف برای یکپارچه سازی استفاده نمود. همچنین وجود چهار سطح مختلف در استاندارد MOF (سطوحM0تاM3) باعث می‌شود تعریف استانداردها ساده شود. از آنجا که MOF دارای سطح تعریف فرافرامدل[12] است می توان استانداردهای جدیدی را تعریف کردکه خود آنها از یک فرااستاندارد بالادستی پیروی کنند.[1]

۸- نتیجه گیری

در این مقاله به بررسی ویژگی‌های سامانه‌های مقیاس وسیع و ساختار مدل رانه پرداختیم. دیدیم که این معماری با ابزارهایی چون مدل و تجرید از سطوح مختلف برای توسعه نرم افزار مدل محور می باشد . برای تولید و توسعه سریع و قابلیت تغییرپذیری بالا برای سیستم‌های اطلاعاتی و نرم‌افزاری بزرگ امروزه و سکوهای مختلف کاری می‌تواند بسیار مفید واقع شود. نرم‌افزار ایجاد شده بوسیله این معماری قابلیت‌های چون استفادهٔ مجدد در سکوهای مختلف و کاربرد های متفاوت را دارد. چون روش‌های مهندسی نرم‌افزار نمی‌تواند مشکلات سامانه‌های بزرگ امروزه را حل نماید،

پس با به وجود آمدن معماری مدل رانه که یکی از جدیدترین استانداردها است، قابلیت‌های نوینی در اختیار صنعت نرم‌افزار قرار می گیرد. سعی کردیم به کمک معماری مدل رانه راهکارهای اولیه برای چالش‌های یکپارچه‌سازی سامانه‌های ULS ارائه کنیم. لازم به ذکراست معماری مدل رانه قطعاً نمی‌تواند یک پاسخ کامل به چالش های توسعه سامانه‌های ULS باشد[1].

مراجع

[1]استادزاده سیدشروین ,شمس فریدون, نقش معماری مدل رانه در یکپارچه‌سازی سامانه‌های فوق وسیع: راهکار ها وچالش ها, پنجمین کنفرانس فناوری اطلاعات و ارتباطات ایران

[2]خوشنویس صدیقه ارائه روشی برای تدوین معماری سازمانی سرویس گرا با استفاده از چار چوب مدل معماری رانه پایان نامه کارشناسی ارشد دانشگاه آزاداسلامی واحد علوم و تحقیقات تهران ۱۳۸۷

[3]طریحی علی , بررسی قابلیت بهر ه برداری از SOAدر چار چوب MDA , گزارش سمینار کارشناسی ارشد, دانشگاه شهید بهشتی , پاییز۱۳۸۶

[4]استاد زاده سید شروین , جایگاه MDA در معماری سیستمهای سازمانی ,گزارش سمینار کارشناسی ارشد , ارشد دانشگاه آزاداسلامی واحد علوم و تحقیقات تهران ۱۳۸۴تابستان

[5]استادزاده، سید شروین، ۱۳۸۵, یک روش مبتنی بر معماری مدل رانه برای مدل سازی یکپارچه سلولهای چارچوب زکمن, پایان نامه کارشناسی ارشد، دانشگاه آزاد اسلامی واحد علوم و تحقیقات، ، دانشکده فنی ، مهندسی، گروه مهندسی کامپیوتر

[6]استاد زاده سید شروین , استاد زاده سیدآرش , شمس فریدون , انطباق چارچوب معماری سازمانی توسعه یافته با مدل معماری مدل را نه , سیزدهمین کنفرانس انجمن کامپیوتر ملی ایران جزیره کیش,۱۳۸۶

[7]استاد زاده سید شروین و شمس فریدون, انطباق چهارچوب سازمانی زکمن با معماری مدل رانه, دوازدهمین انجمن بین المللی کامپیوتر , دانشگاه شهید بهشتی ,اسفند ۱۳۸۵

[8]SEI, Ultra-Large-Scale System :"The Software Callenge of the Future" , software Engineering Institute (SEI), Carnegie Mellon University ,June 2006

Brown, A., An Introduction to Model Driven Architecture, [9] IBM, 2004

Oject Management Group,MDA Guide,Version1.0.1, OMG [10] Document omg/2003-06-01,june2003

[11]Lúcia Abrunhosa Fernandes, Beatriz Helena Neto, Vladimir Fagundes, Geraldo Zimbrão, Jano Moreira de Souza, and"Model-driven Architecture Approach for DataWarehouse" Sixth International Conference on Autonomic and Autonomous Systems, 2010

[12]Chen Fei Cao, Wan-hua Huang, yongWuhan, "Research on Component-based Model Driven ArchitectureDevelopment and assembly"Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing,2009

[13]Oksana Nikiforova, Antons Cernickins, Natalja Pavlova"Discussing the Difference between Model Driven Architecture and Model DrivenDevelopment in the Context of Supporting Tools"Fourth International Conference on Software Engineering Advances,2009

[14] Amir Azim Sharifloo, Mehdi Mirakhorli, Fereidoon Shams:" How Could ULS Systems Achieve Architecture Benefits?"

[15] Mellor, S. J., Scott, K., Uhl, A. and Weise, D., 2004. “MDA distilled: Principles ofModel Driven Architecture”, Addison Wesley.

Mehdi Mirakhorli, Amir Azim Sharifloo, Fereidoon Shams:"Architectural Challenges of Ultra Large Scale Systems [16]

[17]Ostadzadeh, S.S., Shams, F., Ostadzadeh, S.A., "*An MDA-Based Generic Framework to Address Various Aspects of Enterprise Architecture*," Advances in Computer and Information Sciences and Engineering, Springer, August 2008

زیرنویس ها

۱- Ultra-Large-Scale systems

۲- Software Engineering Institute

۳- Object Mangment Group

۴- Model-Driven Architecture

۵- Socio- technical

۶- Orchestration

۷- Computation Independent Model

۸- Platform Independent Model

۹- Platform Specific Model

۱۰- Model-Driven

۱۱- Standard-Compliant

۱۲- Meta Meta Model