سیزدهمین کنفرانس ملی انجمن کامپیوتر ایران
جزیره کیش، خلیج فارس، ایران ۱۹ الی ۲۱ اسفند ۱۳۸۶
انجمن کامپیوتر ایران
Computer Society of Iran
دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر
مقاله کامل

روشی برای مقایسه معماری‌های نرم‌افزار

وجیه‌اله منتقمی
دانشجوی کارشناسی‌ارشد، گروه مهندسی نرم‌افزار
دانشکده مهندسی کامپیوتر
دانشگاه صنعتی شریف
montaghami@ce.sharif.edu

سیدحسن میریان حسین‌آبادی
استادیار، گروه مهندسی نرم‌افزار
دانشکده مهندسی کامپیوتر
دانشگاه صنعتی شریف
hmirian@sharif.ir

چکیده: هدف اصلی روش پیشنهادشده در این مقاله، مقایسه معماری سیستم‌های نرم‌افزاری می‌باشد. تاکنون روش‌های بسیاری برای ارزیابی معماری نرم‌افزار پیشنهاد و بکار گرفته شده است. اما بیشتر این روش‌ها امکان واضح و مستقیمی برای مقایسه دو معماری ارائه نمی‌دهند. روش پیشنهادی امکان مقایسه دو معماری را در تمام دوره‌ی چرخه حیات نرم‌افزار تضمین می‌کند. این روش بر سه مفهوم اهداف کسب‌وکار، مدل کیفی استاندارد و سرویس‌های در سطح معماری استوار است. تمام مراحل این روش برمبنای اهداف کسب‌وکار انتخاب شده می‌باشد و تمام ویژگی‌های کیفی و اولویت‌ها از این اهداف استخراج می‌شوند. لذا با تغییر اهداف کسب‌وکار، بستر فراهم شده برای مقایسه تغییر چندانی نخواهد داشت و به سرعت مراحل انجام مقایسه بازسازی می‌شوند. با استفاده از مدل کیفی استاندارد، بیان، مستندسازی و اندازه‌گیری ویژگی‌های کیفی به صورت یکپارچه و ساده درخواهد آمد. مقایسه دو معماری برمبنای سرویس‌های در سطح معماری صورت می‌پذیرد که این امر باعث محدود شدن دامنه‌ی بررسی مولفه‌ها و اندازه‌گیری ویژگی‌های کیفی می‌شود و از سوی دیگر امکان مقایسه هر دو معماری موجود در یک دامنه را، مستقل از موارد کاربرد خاص آن‌ها فراهم می‌سازد. از این روش می‌توان برای تعیین معماری مرجع برای خط توسعه نرم‌افزار، مرتب‌سازی معماری‌های پیشنهادی باتوجه به هدف کسب‌وکار خاص، نظارت بر میزان پیشرفت پروژه در یک فرآیند مبتنی بر معماری نرم‌افزار و اثبات بهبود حاصل از انجام تغییرات کلی یا جزئی بر معماری پیشین استفاده نمود.

کلمات کلیدی: معماری نرم‌افزار، مقایسه معماری، ارزیابی معماری نرم‌افزار، هدف کسب و کار، سرویس معماری، مدل کیفی، ویژگی کیفی، معیار کیفی

۱- مقدمه

امروزه با افزایش اندازه و پیچیده‌تر شدن سیستم‌های نرم‌افزاری، تامین کیفیت مورد نیاز این سیستم‌ها خود بحث جدیدی را در مهندسی نرم‌افزار پدید آورده است. برخی از اوقات، اندازه‌ی این سیستم‌ها بیش از ده برابر سیستم‌های مشابه دهه‌ی اخیر افزایش یافته است. به همین لحاظ توضیحی سطح بالا از طراحی نرم‌افزار، نقشی مهم و اساسی را در فهم و مدیریت سیستم‌های بزرگ و پیچیده نرم‌افزاری ایفا می‌کند. در واقع ویژگی‌های کیفی مطروحه در سیستم‌های بزرگ نرم‌افزاری (مانند قابلیت نگهداشت، قابلیت‌اطمینان، قابلیت‌استفاده، کارایی، قابلیت‌انعطاف و ...) را بوسیله معماری نرم‌افزار مشخص ساخته و محدودیت‌های مورد نیاز را بوسیله آن ذکر می‌کنند. معماری نرم‌افزار نقش مهمی در دستیابی به ویژگی‌های کیفی سیستم دارد و در این حین ارزیابی معماری در خصوص میزان دستیابی به نیازهای کیفی مطلوب در مراحل اولیه حائز اهمیت است. در واقع هدف اصلی ارزیابی معماری نرم‌افزار، درک میزان پتانسیل معماری انتخاب شده، جهت دستیابی به استعداد برآورده نمودن نیازهای کیفی و شناخت ریسک‌های بالقوه می‌باشد.

تا به امروز روش‌هایی جهت ارزیابی موارد کیفی در سطح معماری نرم‌افزار مطرح شده است، ولی توافق جامعی بر سر روشی که بتواند تمام مسائل تکنیکی و غیر تکنیکی را پوشش دهد وجود ندارد. در بیشتر این روش‌ها، معماری نرم‌افزار نسبت به خواسته‌های کاربران و تعامل بین این درخواست‌ها ارزیابی می‌شود. درخواست‌های کاربران از یک سیستم نرم‌افزار به دو بخش وظیفه‌مند و غیروظیفه‌مند تقسیم می‌شود و معماری نرم‌افزار سعی در بیان هر دو قسم این درخواست‌ها را دارد. نیازهای وظیفه‌مند، صورتی بدیهی دارند و در پایان مرحله‌ی تولید میزان دستیابی به آن‌ها کاملاً واضح و مشخص است، لذا ارزیابی این دسته از نیازها در مرحله اولیه‌ی توسعه نرم‌افزار با استفاده از روش‌های دیگری انجام می‌شود. اما پاسخگویی به درخواست‌های غیروظیفه‌مند، صورت کیفی محصول را طی چرخه حیات آن تضمین می‌کند. با توجه به اهمیت حفظ کیفیت نرم‌افزار و لزوم پی‌ریزی درخواست‌های غیروظیفه‌مند در مراحل ابتدایی چرخه حیات نرم‌افزار، ارزیابی معماری نرم‌افزاری امری مفید و مقرون به‌صرفه است. طی فرآیند توسعه نرم-

افزار، علاوه بر تحلیل و ارزیابی معماری، مقایسه دو یا چند معماری سیستم نرم‌افزاری نیز حائز اهمیت می‌باشد و کاربردهای زیادی را در روش‌های مختلف توسعه نرم‌افزار دارد. به طور مثال، در برپاسازی خط تولید نرم‌افزار، انتخاب معماری مرجع از بین معماری‌های موجود و یا پیشنهاد شده از اهمیت بالایی برخوردار است. این انتخاب از آن جهت مهم است که انتخاب نادرست این معماری موجب نشر خطاها و کمبودها در تمام معماری‌ها خواهد شد.

در بیشتر روش‌های ارزیابی، معماری نرم‌افزار در یک برهه زمانی خاص ارزیابی می‌شود و در نتیجه، نتایج آن ارزیابی منحصر به همان مقطع از زمان خواهد بود. برای مثال در روش SAAM[1] لیستی از نقاط ضعف و قوت و در روش ATAM[2] نقاط مصالحه[3] در طراحی شناسایی می‌شوند. به طور کلی، روش‌های کنونی برای مرور و بررسی معماری یک سیستم و یا انتخاب شیوه طراحی معین و مشخصی که جنبه خاصی از معماری را پوشش می‌دهد، مناسب است. اما این روش‌ها، امکانی را به طور صریح و مشخص برای مقایسه و اولویت‌دهی به معماری‌های مختلف پیشنهاد نمی‌کنند. البته باید توجه داشت پرسشی که مقایسه جزئیات واضحی از معماری ۱ را با معماری ۲ مطرح می‌کند، اغلب پاسخی صریح دارد و به طور حتم، نمی‌توان این رویه را به تمام روش و مقایسه‌ها گسترش داد.

در ادامه چند روش تحلیل و ارزیابی مورد بررسی قرار می‌گیرد و نقاط ضعف آن‌ها نسبت به مقایسه دو یا چند معماری مشخص می‌شود.در بخش بعد روش پیشنهادی توضیح داده می‌شود و دست آخر یک مورد مطالعاتی، جهت وضوح بیشتر روش ارائه شده است.

۲- بررسی چند روش ارزیابی و ضعف آن‌ها در مقایسه چند معماری

اگرچه تاثیر ارزیابی معماری در کاهش هزینه‌ها و افزایش کیفیت نرم‌افزار امری پذیرفته شده به حساب می‌آید، اما روش‌های بررسی و تحلیل معماری، نسبتا نو و ابتدایی می‌باشند. در [1]، روش‌های معروف ارزیابی معماری نرم‌افزار به تفصیل آمده است.

SAAM: این روش ارزیابی معماری نرم‌افزار بر مبنای سناریوها، ویژگی‌های کیفی و اهداف کیفی انجام می‌شود. سناریوها شرحی مختصر از تعامل منفرد ذی‌نفع با سیستم می‌باشند. ویژگی‌ها و اهداف کیفی جنبه‌های یک نیاز را بیان می‌کنند. طی فرآیند ارزیابی، رویکرد معماری شناسایی می‌شود و با توجه به اهداف شناخته شده مورد ارزیابی قرار می‌گیرد. در نتیجه، هر تصمیمی که در سطح معماری اتخاذ شود به یک یا چند خطر بالقوه اشاره دارد و دلیلی بر وجود مشکل می‌باشد.

ATAM: رویکرد این روش در مقایسه با SAAM بر یافتن تصمیمات بحرانی استوار است و سعی در شناسایی خطرات دارد. بنابراین نتایج بدست آمده از فرآیند ATAM بیان‌گر درکی مناسب از نظرات موافق و مخالف هر نقطه مصالحه و تاثیرات راه‌حل‌های گوناگون در حوزه‌ی کل معماری می‌باشد.

CBAM[4]: این روش گسترشی بر روش ATAM است که رویکردی اقتصادی به آن می‌بخشد. این روش با کمی نمودن مزایای اقتصادی و هزینه‌ای که هر تصمیم در پی دارد، به ارزیاب در برآورد هزینه‌ی هر گونه تغییر در سطح معماری کمک می‌کند.

ALMA[5]: این روش ارزیابی معماری نرم‌افزار مبتنی بر سناریو، تمرکز خاصی بر جنبه تغییرپذیری سیستم دارد. پس از انتخاب اهداف ارزیابی، سناریوها و مستندات مرتبط با اهداف ارزیابی استخراج می‌شوند و طی فرآیندی تکرارپذیر، سناریوها بر معماری اعمال شده و نتایج ارزیابی مشخص می‌شود[3].

ARID[6]: این روش با رویکردی موضوع‌گرا، معماری زیرسیستم‌ها را محور اصلی منابع خود در نظر می‌گیرد. به دلیل تمرکز این روش در ارزیابی پیشنهادات طراحی‌های مطرح شده در فازهای ابتدایی ، تکنیک‌های مرور و بازبینی به شکل ساده اما کارا به کار گرفته می‌شوند.

SACAM[7]: این روش [4] ، مقایسه مجموعه‌ای از معماری‌های با دامنه کاربردهای متفاوت را امکان‌پذیر می‌سازد. هدف اصلی این روش، انتخاب بهترین معماری نرم‌افزاری از بین معماری‌های پیشنهادی و یا موجود است تا بتوان از آن به عنوان معماری مرجع در خطوط تولید نرم‌افزار استفاده نمود. ابتدا با انتخاب اهداف کسب و کار سازمان، معیارهای کیفی مربوطه استخراج شده و سپس بوسیله سناریوها بیان می‌شود. همانند روش ATAM، ارزیاب بر مبنای معیارهای و دیدهای از پیش تعریف شده نمراتی را به معماری‌ها می‌دهد.

روشی که در این مقاله پیشنهاد شده است، بر ایده‌هایی عمومی موجود در روش‌های ارزیابی مبتنی بر سناریو استوار است. با این حال، علاوه بر تکنیک‌های موجود در این روش‌ها، مفاهیم جدیدی را برای پوشش دادن به مسئله مقایسه‌ی معماری نرم‌افزاری مطرح می‌سازد. از جمله تفاوت‌های این روش با روش‌های فوق می‌توان به موارد زیر اشاره نمود:

- به‌جز روش SACAM، هیچ‌یک از دیگر روش‌ها اجازه‌ی مقایسه‌ی معماری‌های مختلف را در یک مورد خاص نمی‌دهند. دراین روش‌ها، رویه ارزیابی به دانش ارزیاب و توصیفی که در اختیار وی قرار می‌گیرد بستگی دارد. اما روش ارائه شده برای هنگامی که نیازها به تناوب تغییر نموده و ارزیابی مجدد مورد نیاز است، کاربرد خواهد داشت.
- طی فرآیند ارزیابی SACAM، ارزیاب می‌کوشد با شناخت چند دید مشترک، سطوح انتزاعی همگنی را برای مقایسه فرآهم آورد. علی‌رغم یکپارچگی بوجود آمده در مراحل بعدی فرآیند، مقایسه معماری‌های کاملاً متفاوت از پیچیدگی بسیاری برخوردار است. اما در روش پیشنهادی فضای بستر، به معماری‌های هم‌راستا و هم-

دامنه محدود می‌شود. این هم‌راستایی براساس کارکردهای وظیفه‌مندی که در طرح کلی آمده مشخص می‌شود.

- علاوه بر موارد ذکر شده، در روش پیشنهادی، برای بیان وظیفه-مندی‌های اساسی و خواسته‌های کاربردی انتزاعی، از مفهوم "سرویس‌های مبتنی بر معماری" استفاده می‌شود. سناریوهای مطرح شده در روش‌های فوق بسیار خاص می‌باشند و احتمال تغییر آن‌ها طی چرخه حیات نرم‌افزار زیاد است. اما سرویس‌های مبتنی بر معماری توصیفی سطح بالاتر و دیدی پایدارتر را در مقایسه با سناریوهای مبتنی بر معماری مطرح می‌کنند.

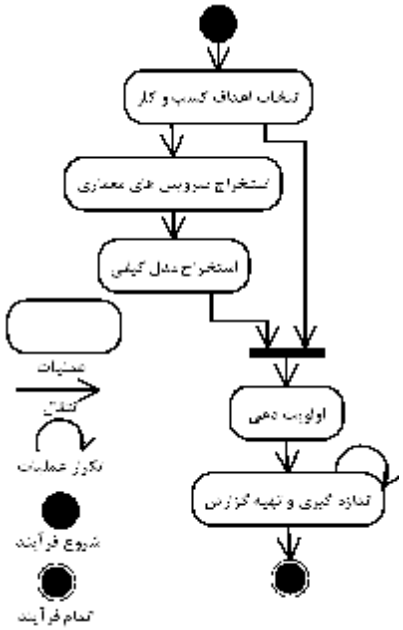
امروزه بیشتر روش‌های ارزیابی معماری بر ارزیابی یک معماری منفرد متمرکز می‌باشند و راه حلی استاندارد و تکرارپذیر برای مقایسه معماری‌های گوناگون ارائه نمی‌دهند. براساس بررسی‌ها انجام شده تنها استثناء شناخته شده روش SACAM می‌باشد. در مقایسه با SACAM باید گفت که روش پیشنهادی از سادگی بیشتر و ساختار پایدارتری برخوردار است.

۳- روش ارزیابی و مقایسه معماری نرم‌افزار

روش پیشنهادی برای ارزیابی و مقایسه معماری دو یا چند سیستم نرم-افزاری، بر سه مفهوم *اهداف کسب و کار*، *سرویس‌های معماری* و *مدل ویژگی‌های کیفی* استوار است. اهداف کسب و کار هر سیستم نرم-افزاری، دلایل وجود، استفاده و توسعه آن را مشخص می‌کنند. شناخت این اهداف و انتخاب معماری مناسب برای پاسخگویی به آن‌ها، موفقیت سیستم را در تمام مراحل چرخه حیات نرم‌افزار تضمین می‌کند. سرویس‌های معماری، بیان پایه‌ای از عامل‌های وظیفه‌مندی یک سیستم‌اند که به تمام نیازهای کمی آن پاسخ می‌دهند. مدل ویژگی‌های کیفی، ارتباط بیان نیازهای غیروظیفه‌مندی و روش‌های اندازه‌گیری آن-ها را مشخص می‌سازد. روش پیشنهادی، این سه مفهوم را در پنج مرحله به کار می‌گیرد تا در پایان مقایسه دقیقی بین معماری‌های مختلف انجام دهد. این پنج مرحله عبارتند از:

۱. **شناخت اهداف کسب و کار:** جمع‌آوری و پالایش اهداف کسب و کار و نیازمندی‌های وظیفه‌مندی اصلی،
۲. **استخراج سرویس‌های در سطح معماری:** استخراج سرویس-های معماری از اهداف کسب و کار، مدل معماری و شیوه‌های معماری،
۳. **استخراج مدل کیفی:** تعریف و تعیین مدل کیفی براساس یک مدل کیفی استاندارد،
۴. **اولویت‌دهی به ویژگی‌های کیفی:** اولویت‌دهی به ویژگی‌های کیفی بر مبنای سرویس‌های معماری و اهداف کسب و کار،
۵. **اندازه‌گیری و تهیه گزارش:** اندازه‌گیری ویژگی‌های کیفی بر اساس مدل کیفی و ساخت گزارش.

این پنج مرحله در قسمت‌های بعد به تفصیل بیان می‌شوند. باید توجه داشت، برای مقایسه معماری‌های مختلف مراحل یک تا چهار تنها یک بار انجام می‌شوند و مرحله پنجم به ازای هر معماری تکرار می‌شود. شکل (۱)، فرآیند ارزیابی و مقایسه معماری را به‌طور واضح بیان می-کند.



شکل (۱): فرآیند ارزیابی و مقایسه معماری

۳-۱- شناخت اهداف کسب و کار

با توجه به اهداف کسب و کار مختلف، هر سیستم را می‌توان از دیدگاه-های متفاوتی مورد ارزیابی قرار داد. با در نظر گرفتن دیدگاه انتخاب شده، می‌توان میزان پاسخ‌گویی به نیاز وظیفه‌مندی ویژه‌ای - به معنای کیفیت سیستم یا ویژگی‌های کیفی، را بررسی نمود. در فرآیند ارزیابی و مقایسه معماری می‌بایست بر جنبه‌هایی از معماری تاکید نمود که برای دستیابی به اهداف کسب و کار سیستم مهم می‌باشند. به‌طورکلی، هر سیستم اهدف کسب و کار مشخصی را برای پاسخگویی به نیاز ذی-نفعان برآورده می‌سازد. اهداف کسب و کار را می‌توان از محیط کسب و کار، نیاز ذی‌نفعان، محدودیت‌های کسب‌وکار، محدودیت‌های تکنیکی و نیازهای کیفی استخراج نمود [2].

در فرآیند پیشنهادی، شناسایی درست اهداف کسب و کار دستیابی به مقایسه درست معماری‌ها را تضمین می‌کند. شناسایی اهداف کسب و کار، پایه‌ای برای یافتن ویژگی‌های کیفی مهم، تعیین اولویت ویژگی‌های کیفی و تعیین ارتباط بین ویژگی‌های کیفی و سرویس‌های معماری به حساب می‌آید. برای ساخت مدل کیفی درست، می‌بایست ویژگی‌های کیفی اصلی را شناسایی نمود. اهداف کسب و کار شناسایی شده، بهترین منبع برای استخراج این ویژگی‌های می‌باشد. در

پایان این مرحله لیستی از اهداف کسب و کار و ویژگی‌های کیفی مهم مشخص می‌شود- البته الزامی بر بیان رسمی و دقیق این ویژگی‌ها وجود ندارد.

۳-۲- استخراج سرویس‌های در سطح معماری

توصیفی انتزاعی که هدف یک سیستم و نیازهای وظیفه‌مندی اولیه آن را نشان می‌دهد، سرویس معماری گوییم. در مقایسه با سناریوهای کاربردی، که بسیار خاص می‌باشند و ممکن است طی چرخه حیات معماری تغییر کنند، سرویس‌های معماری از بیانی سطح بالاتر و دیدگاهی پایدارتر برخوردارند. سرویس‌های معماری، به مانند انواع دیدهای معماری، [۳] فرآیند بررسی را بر جنبه‌هایی خاصی از معماری متمرکز می‌سازند. اما برخلاف آن‌ها، سرویس‌های معماری از نیازهای وظیفه‌مندی مشتق شده‌اند و مسائل فنی موجود در مستندات معماری را شامل نمی‌شوند.

با این توصیف، سرویس‌های معماری بر مبنای نیازهای‌های وظیفه‌مندی و شیوه‌های معماری شناسایی می‌شوند. در حالت اول، معمار سیستم بر مبنای دامنه کاربرد و نیازهای وظیفه‌مندی عمومی سرویس‌های معماری را تشخیص می‌دهد. در حالت دوم، هر شیوه‌ی معماری که در طراحی سیستم به کار گرفته شده، موید پاسخگویی به تعدادی نیاز وظیفه‌مندی و غیروظیفه‌مندی است. پس با شناخت این شیوه‌ها می‌توان سرویس‌های مورد نظر را استخراج نمود.

با یافتن سرویس‌های در سطح معماری، مولفه‌های مربوط به هر سرویس نیز شناسایی می‌شوند و هنگام تحلیل ویژگی‌های کیفی، مولفه‌های کمتری مورد بررسی قرار می‌گیرند. در نتیجه، از پیچیدگی مستندات معماری کاسته و فرآیند ارزیابی و مقایسه با دقت بالاتری پی‌گیری می‌شود. برای مثال، در سرویس انتقال اطلاعات یک سیستم فرضی، تنها مولفه‌هایی چون مسیریاب‌ها، پشته‌پروتکل، نرم‌افزار نهان‌ساز مورد مدل‌سازی قرار خواهند گرفت و بقیه مولفه‌ها در مستندات مربوط به ارزیابی این سرویس لحاظ نخواهند شد.

۳-۳- استخراج مدل کیفی

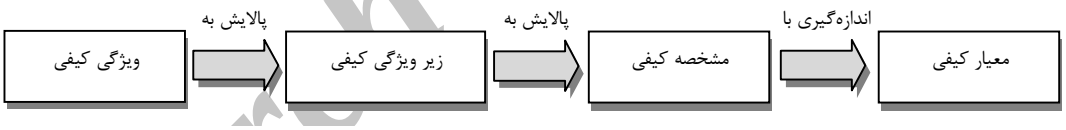
همیشه نیازهای وظیفه‌مندی هرسیستم نرم‌افزاری به همراه تعدادی نیازهای غیروظیفه‌مندی مطرح می‌شود و سیستم می‌بایست سرویس‌های خود را با توجه به ملاحظاتی ارائه نماید. این ملاحظات براساس نیازهای غیروظیفه‌مندی تعریف و با استفاده از مدل کیفی مدل می‌شوند. مدل کیفی، ارتباطی یکپارچه را بین اهداف کسب و کار، ویژگی‌های کیفی شناخته شده و معیارهای اندازه‌گیری ویژگی‌های کیفی برقرار می‌سازد.

شکل (۲): مدل کیفی ISO9126-1 [7]

در سیستم‌های مختلف، از واژگان و گزاره‌های گوناگونی برای بیان ویژگی‌ها و معیارهای کیفی استفاده می‌شود. اما هنگام ارزیابی و مقایسه ویژگی‌های کیفی معماری‌های مختلف، باید از مدلی یکسان و استاندارد استفاده نمود. مدل کیفی ISO9126-1 از جمله مدل‌های استانداردی است که ویژگی‌های مختلف کیفی را بیان نموده و ارتباط بین آن‌ها را مشخص می‌کند. همان‌گونه که در شکل (۲) مشخص است، ویژگی‌های کیفی به شش دسته اصلی تقسیم می‌شوند:

- **وظیفه‌مندی**[۸]: قابلیت یک محصول نرم‌افزاری در برآورده‌سازی نیازهای وظیفه‌مندی طی یک شرایط خاص را ویژگی وظیفه‌مندی سیستم گویند.(این ویژگی را نباید با نیازمندی‌های وظیفه‌مندی اشتباه در نظر گرفت.)
- **قابلیت‌اطمینان**[۹]: قابلیت یک محصول نرم‌افزاری در حفظ سطح مشخصی از سرویس‌دهی طی یک شرایط خاص و در بازه‌ی مشخص زمانی را ویژگی قابلیت‌اطمینان گویند.
- **قابلیت‌استفاده**[۱۰]: قابلیت یک محصول نرم‌افزاری در فهم، آموزش، استفاده و جذب توجه کاربر طی یک شرایط خاص را ویژگی قابلیت‌استفاده سیستم گویند.
- **کارایی**[۱۱]: قابلیت یک محصول نرم‌افزاری در برآورده نمودن کارایی مناسب، با در نظر گرفتن منابع استفاده شده طی یک شرایط خاص را ویژگی کارایی سیستم گویند.
- **قابلیت‌نگهداشت**[۱۲]: قابلیت تغییر یک محصول نرم‌افزاری ویژگی قابلیت‌نگهداشت نام دارد. تغییر در سیستم‌های نرم‌افزاری به معنای تصحیح، بهبود یا انطباق با تغییرات محیطی، نیازها و توصیفات وظیفه‌مندی می‌باشد.
- **قابلیت‌حمل**[۱۳]: قابلیت انتقال یک محصول نرم‌افزاری از یک محیط به محیط دیگر، ویژگی قابلیت‌حمل نام دارد. مفهوم محیط، انواع محیط‌های سازمانی، سخت‌افزار یا نرم‌افزار را در بر می‌گیرد.

هر ویژگی کیفی، دامنه خاصی از نیازهای غیروظیفه‌مندی را پوشش می‌دهد. لذا می‌توان هر ویژگی کیفی را به تعدادی زیرویژگی دیگر تقسیم نمود. با انجام این پالایش، اهداف کسب‌وکار و نیازهای غیروظیفه‌مندی به‌طور واضح و شفاف بیان می‌شوند. استاندارد ISO9126-1 پیشنهادی را برای تقسیم هر ویژگی ارائه داده است که در شکل (۲) آمده است. پس از بیان دقیق اهداف کسب‌وکار و نیازهای غیروظیفه‌مندی، می‌بایست معیاری را برای اندازه‌گیری و سنجش هر زیر ویژگی کیفی ارائه داد. برای این منظور می‌بایست زیرویژگی‌های مورد نظر برای هر سرویس را به اعدادی قابل مقایسه تبدیل نمود تا با اولویت‌دهی به آن‌ها بتوان مقایسه دقیقی بین پیاده-سازی‌ها مختلف از یک سرویس بدست آورد. همان‌گونه که در شکل (۳) مشخص است، هرویژگی کیفی به تعدادی زیرویژگی کیفی پالایش می-شود تا بتوان مشخصه‌های قابل شمارشی از آن‌ها بدست آورد. در این میان، از معیار به عنوان روش اندازه‌گیری و ابزاری برای اختصاص یک عدد به ویژگی کیفی مربوط به آن، استفاده می‌شود. برای مثال، درباره میزان اعتبار یک محصول نرم‌افزاری، ویژگی قابلیت‌نگهداشت را می‌توان به زیرویژگی قابلیت‌تست پالایش کرد و از آن پیچیدگی کد را به عنوان مشخصه کیفی مرتبط با قابلیت‌تست در نظر گرفت و آن را با معیارهای مربوطه، به ویژه تعداد خطوط مربوط به هر مولفه، اندازه گرفت.



شکل (3): رابطه بین المان‌های مدل کیفی

مشخصه‌های کیفی تمام ویژگی‌های غیروظیفه‌مندی قابل اشاره (مانند کارایی) و غیرقابل شمارش (مانند امنیت) را در بر می‌گیرد. ویژگی‌های غیر قابل شمارش را با توجه به سطح آن مورد اشاره قرار می‌دهند. برای مثال امنیت را می‌توان با توجه به وجود مکانیزم‌های امنیتی، مانند شناخت کاربران، مدل سطح دسترسی و کدگذاری بر داده شمارش نمود.

۳-۴- اولویت‌دهی به ویژگی‌های کیفی

برای آن‌که بتوان ارزیابی و مقایسه دقیقی از معماری‌های پیشنهادی داشت، می‌بایست برای مشخصه‌های مختلف مورد ارزیابی، اولویت‌بندی دقیقی تعیین کرد. با توجه به نتایج مراحل پیشین، مشخصه‌های سیستم به سرویس‌های در سطح معماری و ویژگی‌های کیفی پالایش می‌شوند. هر ویژگی کیفی اولویت به‌خصوصی را برای مجموعه‌ی سیستم و هریک از سرویس‌ها دارد که با لحاظ نمودن آن‌ها، دقت مقایسه افزایش یافته و روش پیشنهادی به نیاز ذی‌نفعان نزدیک‌تر خواهد بود. اولویت‌دهی در سه مرحله انجام می‌شود. در مرحله اول، ویژگی‌های کیفی نسبت به هم اولویت‌دهی می‌شوند. در مرحله بعد زیرویژگی‌هایی که در قالب یک ویژگی دسته‌بندی شده‌اند، نسبت به هم اولویت‌دهی می‌شوند. اولویت‌ها به طور مستقیم از اهداف کسب‌وکار بدست می‌آیند. این اولویت‌ها بین دو سطح متغیر خواهند بود. به ویژگی‌هایی که عدم تامین آن‌ها دستیابی به اهداف کسب‌وکار را با مشکل مواجه کند اولویت بالا و به ویژگی‌هایی که اهداف کسب‌وکار اهمیت کمتری برای آن‌ها قائل‌اند، اولویت کمتری داده می‌شود. این

سرویس‌های معماری	ویژگی کیفی ۱						ویژگی کیفی ۲				ویژگی کیفی ۳			
	زیرویژگی۱		زیرویژگی۲		زیرویژگی۳		زیرویژگی۱		زیرویژگی۲		زیرویژگی۱		زیرویژگی۲	
	وزن	مقدار	وزن	مقدار	وزن	مقدار	وزن	مقدار	وزن	مقدار	وزن	مقدار	وزن	مقدار
سرویس ۱														
سرویس ۲														
سرویس ۳														
اولویت زیرویژگی														
اولویت ویژگی														

جدول (۱): ماتریس مقایسه و ارزیابی سرویس‌های معماری

اولویت‌ها بین اعداد ۰ تا ۱۰۰ متغیر می‌باشد و پس از هر مرحله اولویت‌دهی به صورت نرمال درآورده می‌شوند.

پس از اولویت‌دهی نسبی به ویژگی‌های کیفی و زیرویژگی‌های آن، در مرحله آخر نوبت به اولویت‌دهی به زوج‌مرتب زیرویژگی کیفی و سرویس در سطح معماری می‌رسد. همان‌گونه که در جدول (۱) آمده است، تمام سرویس‌های در سطح معماری نسبت به هم و در قالب زیرویژگی کیفی وابسته به آن اولویت دهی می‌شود. با اعمال این اولویت‌بندی‌ها می‌توان بستر مقایسه ارائه شده را برای دامنه‌های مختلف تنظیم نمود و نقاط ضعف و قوت معماری‌های مختلف را نسبت به هم و در سطوح مختلف کیفی سنجید

۳-۵- اندازه‌گیری و تهیه گزارش

پس از اولویت‌دهی به ویژگی‌های مختلف، نوبت به اندازه‌گیری معیارهایی است که در مرحله‌ی قبل شناسایی شده‌اند. به دلیل آن‌که هر معیار اندازه‌گیری شده برای سرویس در سطح معماری بازه‌ی تغییرات مربوط به خود را دارد، می‌بایست اعداد و ارقام بدست آمده را نرمال‌سازی و در ستون مربوط به مقدار هر معماری وارد نمود. این نرمال سازی با توجه به حداکثر اندازه هر معیار انجام می‌شود.

با توجه به جدول بدست آمده می‌توان انواع مقایسه‌ها را در سطوح مختلف ویژگی‌های کیفی و سرویس در سطح معماری انجام داد. با جمع نمودن مقادیر معیارهای مختلف اندازه‌گیری شده در سرویس‌های گوناگون و در نظر گرفتن اولویت‌های مختلف، می‌توان معماری‌ها را در سطح زیرویژگی و یا ویژگی کیفی مقایسه نمود. به‌طور مثال می‌توان گفت بر اساس مدل کیفی ISO9126-1 معماری ۱ از لحاظ قابلیت-دسترسی بهتر از معماری ۲ است. این روال را می‌توان با وارد نمودن ضرایب اولویتی بین ویژگی‌های کیفی مختلف در سطح دو معماری نیز مطرح نمود و در پایان به این نتیجه رسید که معماری ۲ از معماری ۱ بهتر است.

۴- بررسی یک مورد مطالعاتی

برای بررسی بیشتر روش پیشنهادی، معماری دو سیستم انتقال صوت بر پایه شبکه IP مورد مقایسه قرار می‌گیرند. این دو سیستم بر پایه پروتکل انتقال SIP[۱۴] کار می‌کنند. این دو معماری در یک حوزه قرار می‌گیرند و قصد دارند مسئله انتقال صوت بر پایه شبکه IP و سوییچینگ انتقال صوت را بهبود دهند. هر یک از این دو معماری جنبه‌های خاصی از پیاده‌سازی این پروتکل را در نظر دارند. معماری اول JAIN-SIP نام دارد و بر پایه تکنولوژی جاوا بنا نهاده شده است. معماری دوم SNOCER نام دارد. این دو معماری در [13,14] به تفصیل بیان شده‌اند.

۴-۱- شناخت اهداف کسب و کار

معماری‌های پیشنهادی سعی در ارائه راه‌حلی برای مسئله انتقال صوت در شبکه IP و بر پایه پروتکل SIP دارند. پروتکل SIP، فرآیند علامت-دهی را در انتقال صوت به صورت نظیر به نظیر حل می‌کند. لذا بدون در نظر گرفتن پیاده‌سازی خاصی از این شبکه می‌توان اهداف کسب و کار زیر را استخراج نمود:

- برقراری ارتباط بین هر دو نقطه اینترنت و در هر زمان
- حفظ کیفیت داده در حال انتقال به صورت یکنواخت
- اعمال تغییرات در سطح کسب و کار و پروتکل در سریع‌ترین زمان و کمترین هزینه
- ارسال و دریافت اطلاعات به صورت محرمانه و جلوگیری از شنود ناخواسته

با توجه به اهداف کسب و کار مذکور، نیازهای غیروظیفه‌مندی این سیستم عبارتند از قابلیت‌دسترسی بالا و کیفیت انتقال داده تضمین شده و امنیت داده‌ها هنگام علامت‌دهی و انتقال اطلاعات.

۴-۲- استخراج سرویس‌های در سطح معماری

شبکه انتقال صوت مبتنی بر IP از دو بخش سرویس و مشتری تشکیل شده است. این دو بخش ارتباطی نظیر به نظیر را با استفاده از پروتکل SIP برقرار می‌سازند. سرویس‌های در سطح معماری مهم و قابل توجه در سمت سرور به صورت زیر است:

سرویس انتقال اطلاعات: این سرویس امکان انتقال اطلاعات بین مشتری‌ها و سرویس‌دهندگان را فراهم می‌سازد. مشتری‌ها و سرویس‌دهندگان بر روی سکوها مختلف قرار دارند و در لحظه‌ی تماس انتظار سرویسی در لحظه، با فرمت صوتی، تصویری یا داده‌ای را دارند.

سرویس ذخیره و بازیابی اطلاعات کاربران: تمام اطلاعات مشتری‌ها در پایگاهی نگهداری می‌شود تا در هنگام تماس تصمیمات مناسبی جهت برقراری ارتباط اتخاذ شود. در این حین تمام داده‌های با ارزش جهت محاسبه صورت‌حساب و پیگیری‌های آتی در سرور ذخیره می‌شود.

سرویس شناخت کاربران و سطوح دسترسی: این سرویس وظیفه شناخت و کنترل دسترسی مشتریان به مولفه‌ها و سرویس‌های کاربردی سرور را در برقراری تماس و تنظیم اطلاعات بر عهده دارد.

سرویس مدیریت منابع: منابعی همچون نخ‌های پردازشی و اتصال به پایگاه اطلاعات توسط این سرویس مدیریت می‌شود.

سرویس نمایش اطلاعات کاربران: این سرویس اطلاعات مورد نیاز کاربران را در قالب گرافیکی برای مشتریان فراهم می‌آورد.

با توجه به سرویس‌های در سطح معماری، می‌توان قابلیت‌های وظیفه‌مندی و مولفه‌های پیاده‌ساز آن‌ها را دسته‌بندی نمود و نیازهای وظیفه‌مندی را برای هر دسته به صورت مجزا انجام داد. این کار باعث کاهش پیچیدگی و افزایش دقت مقایسه خواهد شد.

۴-۳- استخراج مدل کیفی

برای یکپارچه‌سازی فرآیند مقایسه، از مدل کیفی استاندارد ISO9126-1 استفاده می‌شود. این مدل کلی می‌باشد، لذا برخی از ویژگی‌ها و زیرویژگی‌های غیر ضروری را حذف خواهند شد. حذف این ویژگی‌ها براساس اهداف کسب و کار شناسایی شده در مرحله اول می‌باشد. بنابر نتایج بدست آمده از مرحله اول، ویژگی‌های قابلیت‌اطمینان، کارایی، وظیفه‌مندی مهم‌ترین ویژگی‌های کیفی در این معماری‌ها می‌باشند. برای توصیف و اندازه‌گیری دقیق، هریک از این ویژگی‌ها به زیر ویژگی‌هایی پالایش می‌شوند.

هدف کسب‌وکار زیر به زیرویژگی قابلیت‌دسترسی اشاره دارد.

- حفظ برقراری ارتباط بین هر دو نقطه اینترنت و در هر زمان

به دلیل آن‌که قابلیت‌دسترسی به‌طور مستقیم در سطح معماری قابل شناسایی و اندازه‌گیری نمی‌باشد آن را به دو زیرویژگی قابلیت‌بازیافت و تحمل‌خطا پالایش می‌دهیم. قابلیت‌بازیافت به معنای ادامه سرویس‌دهی با کارایی قابل قبول، پس از رویداد وقفه‌ای می‌باشد. این زیرویژگی در سطح معماری با تشخیص مکانیزم‌های مربوط به این کار اندازه‌گیری می‌شوند. این مکانیزم‌ها یا بوسیله مولفه‌هایی مجزا و یا بوسیله سرویسی از یک مولفه، پیاده‌سازی می‌شوند. برای مثال برپاسازی هم‌زمان چند مولفه سرویس‌دهنده به مشتریان یا ذخیره‌سازی داده‌ها در چند مولفه موازی از جمله‌ی این مکانیزم‌ها می‌باشد. پس از تشخیص این مکانیزم‌ها، برای اندازه‌گیری دقیق، هریک از آن‌ها به زمان و کارایی لازم برای بازشروع سیستم، پالایش می‌شود. قابلیت‌تحمل خطا به توانایی حفظ سطح خاصی از سرویس‌دهی در هنگام بروز مشکل گفته می‌شود. این زیرویژگی در سطح معماری با تشخیص مکانیزم‌های مربوط به این کار اندازه‌گیری می‌شود. مکانیزم‌های لازم برای قابلیت-تحمل خطا در این نوع سیستم‌ها عبارتند از: مکانیزم توزیع بار، مکانیزم تشخیص خطا، مکانیزم ذخیره‌سازی چندگانه. در صورت وجود هر یک از این مکانیزم‌ها، یک سطح به مقدار این زیرویژگی افزوده می‌شود.

هدف زیر به زیرویژگی امنیت از ویژگی وظیفه‌مندی اشاره دارد.

- ارسال و دریافت اطلاعات به صورت محرمانه و جلوگیری از شنود ناخواسته

این زیرویژگی به دسترسی غیرمجاز داده‌ها اشاره دارد. اندازه‌گیری این زیرویژگی با شمارش مکانیزم‌های حفظ امنیت انجام می‌گیرد. این مکانیزم‌ها عباتند از: مکانیزم کنترل‌دسترسی، مکانیزم شناخت کاربران، مکانیزم حفظ جامعیت، مکانیزم عدم انکار و مکانیزم پیگیری تغییرات. وجود هر یک از این مکانیزم‌ها به معنای یک سطح اضافه می‌باشد.

هدف زیر به زیرویژگی رفتار مبتنی بر زمان از ویژگی کارایی اشاره دارد.

- کیفیت داده در حال انتقال به صورت یکنواخت

این زیر ویژگی به سرویس‌دهی سیستم در شرایط کاری مختلف می‌باشد. برای اندازه‌گیری این ویژگی، از شمارش زمان‌های پاسخگویی مولفه‌ها و متصل‌کننده‌ها به‌کاررفته در پاسخگویی به درخواست‌ها، استفاده می‌شود.

۴-۴- اولویت‌دهی به ویژگی‌های کیفی

برای اولویت‌دهی به ویژگی‌های کیفی و زیرویژگی‌های کیفی از اهداف کسب‌وکار استفاده می‌شود. با توجه به اهداف کار ذکر شده، هر سه ویژگی‌کیفی از اولویت یکسانی برخوردارند. این اولویت‌ها پس از نرمال سازی وارد جدول ۲ است. اولویت‌دهی به زیر ویژگی‌های کیفی نیز با توجه به نیاز کیفی سرویس‌ها در جدول (۲) آمده است.

۴-۵- اندازه‌گیری و تهیه گزارش

پس از تعیین هریک از معیارهای اندازه‌گیری برای زیر ویژگی‌های کیفی، بر مبنای سرویس‌های شناسایی شده اندازه‌گیری می‌شوند. نتایج این اندازه‌گیری‌ها در جدول (۲) و (۳) آمده است. مقادیر بین دو جدول

جدول (۲): نتایج بررسی معماری JAIN

سرویس‌های معماری	قابلیت‌اطمینان				وظیفه‌مندی		کارایی	
	قابلیت‌بازیافت		تحمل‌خطا		امنیت		رفتار مبتنی بر زمان	
	وزن	مقدار	وزن	مقدار	وزن	مقدار	وزن	مقدار
انتقال اطلاعات	۲۵	۶۱	۳۳	۶۲	۲۰	۶۰	۳۳	۷۰
ذخیره و بازیابی اطلاعات کاربران	۲۵	۳۴	۳۳	۸۵	۲۰	۴۰	۳۳	۶۰
شناخت کاربران و سطوح دسترسی	۲۵	۲۸	۳۳	۶۴	۲۰	۴۰	۰	۰
مدیریت منابع	۲۵	۳۴	۰	۰	۲۰	۲۰	۳۳	۶۰
نمایش اطلاعات کاربران	۰	۰	۰	۰	۲۰	۶۰	۰	۰
اولویت زیرویژگی	۵۰		۵۰		۱۰۰		۱۰۰	
اولویت ویژگی	۳۳				۳۳		۳۳	

جدول (۳): نتایج بررسی معماری SNOCER

سرویس‌های معماری	قابلیت‌اطمینان				وظیفه‌مندی		کارایی	
	قابلیت‌بازیافت		تحمل‌خطا		امنیت		رفتار مبتنی بر زمان	
	وزن	مقدار	وزن	مقدار	وزن	مقدار	وزن	مقدار
انتقال اطلاعات	۲۵	۵۰	۳۳	۳۰	۲۰	۹۵	۳۳	۴۰
ذخیره و بازیابی اطلاعات کاربران	۲۵	۳۵	۳۳	۱۵	۲۰	۴۰	۳۳	۳۰
شناخت کاربران و سطوح دسترسی	۲۵	۲۵	۳۳	۲۵	۲۰	۶۰	۰	۰
مدیریت منابع	۲۵	۵	۰	۰	۲۰	۶۰	۳۳	۳۰
نمایش اطلاعات کاربران	۰	۰	۰	۰	۲۰	۸۰	۰	۰
اولویت زیرویژگی	۵۰		۵۰		۱۰۰		۱۰۰	
اولویت ویژگی	۳۳				۳۳		۳۳	

به صورت نرمال آورده شده‌اند. برای مثال، زیر ویژگی امنیت تعداد با توجه به تعداد مکانیزم‌های حفظ و برقراری امنیت اندازه‌گیری می‌شود. در معماری JAIN سه مکانیزم حفظ جامعیت، شناخت کاربران و کنترل دسترسی‌ها در سرویس انتقال اطلاعات وجود دارد. با توجه به پنج سطح تعریف شده در بخش۴-۳ این معیار پس از نرمال‌سازی نسبت به حداکثر اندازه معیار در جدول (۲) ثبت خواهد شد.

با توجه به این جداول می‌توان گفت معماری مربوط به JAIN نسبت به معماری SCONER از کارایی و قابلیت‌اطمینان بالاتری برخوردار است اما امنیت در معماری دوم نقش اساس دارد. این جداول را می‌توان برای معماری‌های هم‌دامنه دیگر نیز به‌کار برد و یا اگر تغییری در هریک از معماری انجام شود می‌توان میزان بهبود این تغییر را اندازه گرفت.

۵- نتیجه‌گیری و کارهای آتی

مقایسه دو معماری کاربردهای گسترده‌ای در توسعه و چرخه حیات نرم‌افزار دارد. از این روش می‌توان در انتخاب معماری مرجع برای خط تولید نرم‌افزار، سنجش میزان بهبود یک معماری پیشنهادی و فرآیند تولید نرم‌افزار مبتنی بر معماری نرم‌افزار استفاده نمود.

روش پیشنهادی از جهات زیر قابل گسترش و توسعه می‌باشد:

- بیان رسمی سرویس‌های معماری و ارتباط آن‌ها با زبان‌های توصیف معماری
- بهبود مراحل پالایش از اهداف کسب‌وکار تا معیارهای کیفی دقیق
- بهبود مراحل پالایش از اهداف کسب و کار تا اولویت‌دهی به ویژگی‌های کیفی و سرویس‌های معماری

مراجع

[1] Clements, P., Kazman, R. and Klien, M., *Evaluating Software Architecture Methods and Case Study*, SEI Series in Software Engineering, 2002.

[2] Stoermer, C., Bachmann, F. and Verhoef, C., *SACAM: The Software Architecture Comparison Analysis Model*, Carnegie Mellon University, Software Engineering Institute. Pittsburgh, 2003.

[3] Rosa, N. S., Justo, G. R., Cunha, P., *A Framework for Building Non-Functional Software Architectures*, ACM, 2001, ACM.

[4] Kruchten, P., *Architectural Blueprints-The "4+1" View Model of Software Architecture*, November 1995, IEEE Software, Vol. 12(6), pp. 42-50.

[5] Bass, L., Celements, P. and Kaszman, R., *Software Architecture in Practice. 2nd. s.l. : SEI Series in Software Engineering*, 2003.

[6] Clements, P., et al., *Documenting Software Architecture,* 2nd Review, SEI Series in Software Engineering, 2001.
[7] Losavio, F., et al. *Quality Characteristics for Software Architecture. Journal of Object Technology,* March-April 2003, Vol. Vol. 2, pp. 133-150.
[8] Chastek, G. and Ferguson, R., *Toward Measures for Software Architectures,* Cornegie Mellon University, Software Engineering Institute. Pittsburgh: s.n., 2006. CMU/SEI-20060TN-013.
[9] Boucke, N. and Holvoet, T., *Relating Architectural Views with Architectural Concerns,* ACM, 2006. ACM.
[10] Bengfsson, P., et al., *Architecture-level modifiability analysis (ALMA),* The Journal of System and Software. 2004, Vol. Vol. 69, pp. 129-147.
[11] Kazman, R., et al. *A Basic for Analyzing Software Architecture Analysis Methods,* Software Quality Journal. 2005, pp. 329-355.
[12] Losavio, Farncisca, et al. ISO quality standards for measuring architectures. The Journal of Systems and Software. 2004, Vol. Vol. 72, pp. 209-223.
[13] Jepsen T., Anjum F., Bhat R. R., Jain R., Sharma A., Tait D.. Java in Telecommunications Solutions for Next Generation Networks. Wiley Press, Chapter 4, 2001
[14] Dagiuklas, T., Geneiatakis, D., Kambourakis, G., Sisalem, D., Ehlert, S., *General Reliability and Security Framework for VoIP Infrastructures,* May 2005

زیرنویس‌ها

[1] Software Architecture Analysis Method
[2] Architectural Tradeoff Analysis Method
[3] Trade-off point
[4] Cost Benefit Analysis Method
[5] Architecture-Level Modifiability Analysis
[6] Active Review for Intermediate Design
[7] Software Architecture Comparison Analysis Method
[8] Functionality
[9] Reliability
[10] Usability
[11] Efficiency
[12] Maintainability
[13] Portability
[14] Session Initiation Protocol