دوازدهمین کنفرانس بین‌المللی انجمن کامپیوتر ایران
دانشگاه شهید بهشتی، دانشکده مهندسی برق و کامپیوتر ،تهران، ایران، ۱ تا ۳ اسفند ۱۳۸۵

روش احتمالی سنجش ارزش تصمیم‌های معماری برای ارزیابی معماری نرم‌افزار

مجید وفایی جهان* فریدون شمس† سعید ستایشی‡

چکیده

در این مقاله کوشش شده است که روشی ریاضی مبتنی بر قوانین احتمال جهت انتخاب یک معماری مناسب از میان موارد پیشنهادی ارائه گردد. بطوری‌که با دسته‌بندی مجموعه صفات کیفی نرم‌افزار و راه‌حلهای (تصمیم‌های) معماری و محاسبه ارزش آنها، قالبی ریاضی– احتمالی جهت برازش معماری ایجاد شود. در این خصوص ابتدا صفات و راه‌حلهای معماری، ماتریس ارزش معماری را تشکیل می دهند. هر درایه ماتریس، نشان‌دهنده ارزش راه‌حل برای تامین یک صفت کیفی بوده و طبق روش فرآیند تحلیلی سلسله مراتبی (AHP) مقداردهی می‌شوند. با اعمال وزن صفات بر ماتریس فوق، بردار ارزش حاصل می‌گردد که نشان دهنده ارزش هر راه‌حل در تامین تمام صفات کیفی معماری می‌باشد. با توجه به اینکه در این روش، راه‌حل‌های معماری برای تامین صفات کیفی دارای توزیع احتمالی یکنواختی هستند، چگالی احتمال بردارهای ارزش، بیشتر حول میانگین تجمع کرده بنابراین راه‌حل‌های معماری‌های مختلف یک نرم‌افزار در قالب بردار ارزش از توزیع احتمال نرمال تبعیت می‌کنند. با توجه به این دستاورد، چگالی ارزش معماری یا برازش معماری تعریف و با استفاده از آن مناسب بودن معماری نسبت به مجموعه معماری‌های پیشنهادی برای نرم‌افزار قابل سنجش خواهد بود.

از مزایای عمده این روش می‌توان به داشتن معیار عددی طبق قوانین ریاضی- احتمالی اشاره کرد بطوری‌که قابلیت سنجش و برازش هر نوع معماری و انتخاب بهترین معماری از بین معماری‌های پیشنهادی را دارد و می تواند بهترین معماری را با تقریب خوبی در حدود ۱۰ درصد خطا انتخاب کند. همچنین در بعضی از روش‌های دیگر افزایش تعداد معماری‌های پیشنهادی باعث پیچیده‌تر شدن فرآیند تصمیم‌گیری می‌گردد در حالیکه در این روش نه تنها افزایش پیچیدگی تصمیم‌گیری دیده نمی‌شود بلکه دقت پاسخ افزایش می یابد.

واژه‌های کلیدی: ارزیابی معماری نرم‌افزار، صفات کیفی، تصمیم معماری، توزیع نرمال، فرآیند تحلیل سلسله مراتبی **AHP**.

A Probabilistic Method of Architecture Decisions Evaluation of Value for Software Architecture Trade-off Analysis

In this paper, a new approach based on probabilistic rules is presented to select the best software architecture based on architecture's decisions evaluation. In this method firstly, produce usefulness matrix by classifying software architecture to attributes (rows) and decisions (columns). Each item of matrix shows the value of decision for satisfying related attribute and initialized with Analytic Hierarchy Process (AHP) method.By multiplying weight of attributes to this matrix, usefulness vector is produced which shows the usefulness of each decision for satisfying all architecture's attributes. Because of the uniform distribution of decisions, vectors that are produced through this approach are integrated around the average of vectors and will have normal distribution. Hence,we can define architecture usefulness density to calculate value of architecture and select the best ones.

The benefit of this method is, owning new probabilistic method for software architecture evaluation to select best architecture (with approximation of ۱۰ percent). Also in some of the other methods increasing number of architecture make the decision complicate but in this purposed method the precision of the architecture selection is increased because of the property of normal distribution.

Keywords: Architecture trade off analysis, Quality attributes, Architecture decision, AHP, Normal distribution

* - مربی، گروه کامپیوتر، دانشکده فنی و مهندسی دانشگاه آزاد اسلامی مشهد، VafaeiJahan@mshdiau.ac.ir.
† - استادیار، دانشکده مهندسی برق و کامپیوتر دانشگاه شهید بهشتی، F_Shams@sbu.ac.ir.
‡ - استادیار، گروه مهندسی هسته‌ای (پرتوپزشکی)، دانشکده فیزیک و علوم هسته‌ای دانشگاه صنعتی امیرکبیر، Setayesh@aut.ac.ir.

۱. مقدمه

در یک دهه اخیر معماری نرم‌افزار و ارزیابی معماری توجه بسیاری از محققان را به‌خود جلب کرده است. با توجه به رشد روزافزون صنعت، نرم‌افزارها نیز با پیچیده گیهای زیادی روبرو شده اند که از آن‌جمله می‌توان به تقاضای روزافزون کیفیت نرم‌افزار اشاره کرد. از آنجایی که صفات کیفی تاثیر بسزایی در انتخاب معماری مناسب نرم‌افزار دارند انواع روش‌های ارزیابی معماری با کاربردهای همه منظوره یا خاص منظوره ارائه شده است. اساس اکثر روشها بر مبنای سناریوهای کیفی است و معمار نرم‌افزار با تجربه شخصی و با کمک دیگر متخصصان میزان همپوشانی یک معماری و مجموعه نیازهای کیفی درخواستی را بررسی کرده و در خصوص مناسب بودن معماری تصمیم گیری می‌کند. اکثر روش‌های پیشنهادی، چارچوب‌هایی را مشخص کرده اند که برازش راه‌حل‌های معماری طبق معیارهای عددی و روش‌های ریاضی مبتنی بر بهینه سازی یا تصمیم یار بنا شده است. با توجه به سطح تجریدی بالای معماری، تعریف صفات از قوانین خاصی پیروی می‌کند و معمولا با مشکلات عدم رعایت سطوح تجریدی مواجه است. کوشش شده است تا علاوه بر مرور کارها و فعالیت‌های انجام شده دو هدف برآورده شود: ۱) بیان قالبی مشخص برای توصیف نیازهای کیفی هم‌سطح، دسته‌بندی و ارزش دهی به آنها. ۲) کمک به معمار نرم‌افزار برای انتخاب معماری مناسبتر از بین معماری‌های پیشنهادی از طریق برازش عددی معماری.

بنابراین ابتدا در قسمت ۲ به بررسی فعالیت‌های مرتبط پرداخته شده است. در قسمت ۳ سلسله مراتب صفات کیفی و راه‌حلهای معماری بررسی شده، در قسمت ۴ نمایش ماتریسی آنها تعریف و در قسمت ۵ دستاوردهای مقاله توصیف شده اند. در قسمت ۶ یک مسئله موردی با روش پیشنهادی بررسی و در قسمت ۷ و ۸ بحث و نتیجه گیری و پیشنهادات جهت ادامه فعالیت مطرح گردیده است.

۲. فعالیت‌های مرتبط

در زمینه معماری نرم‌افزار و ارزیابی معماری نرم‌افزار فعالیت‌های ارزشمند زیادی صورت گرفته است. در برخی از مقالات، نحوه یافتن صفات کیفی مناسب و تاثیرگذار در سیستم‌های نرم‌افزاری مورد بحث قرار گرفته است. در برخی دیگر روش‌های ارزیابی معماری و انواع تکنیک‌هایی که در ارزیابی معماری استفاده می شود مورد تحقیق قرار گرفته اند. از فعالیت‌های صورت گرفته دسته اول، در [۱] مجموعه تمام نیازهای غیروظیفه‌مندی و یا نیازهای کیفی نرم‌افزار مورد بررسی گرفته است و ضمن بیان نحوه یافتن آنها مکانیزم‌های انتخاب و اولویت دهی صفات کیفی بررسی شده است. در [۲،۳] روش‌هایی جهت اعمال ملاحظات کیفی ارائه شده است که طبق آن معمار نرم‌افزار می تواند صفات کیفی را بر نرم‌افزار اعمال کند. در [۴،۵،۶] نیز علاوه بر برشمردن صفات کیفی مهم و تاثیر آن بر معماری نرم‌افزار، تعامل صفات و نحوه تاثیر آنها بر یکدیگر توصیف شده‌اند. بعنوان مثال افزایش کارایی نرم‌افزار باعث کاهش قابلیت اصلاح پذیری نرم‌افزار یا افزایش امنیت باعث کاهش قابلیت حمل نرم‌افزار می‌گردد. موارد دیگری از این دست در [۶،۲۲] طبقه‌بندی شده‌اند. از سویی دیگر بسیاری از مقالات در خصوص روش‌های ارزیابی معماری پیشنهاداتی را مطرح کرده اند که طبق مروری که در [۱،۷] صورت گرفته تکنیکهای ارزیابی به دو دسته پرسشی و اندازه‌گیری تقسیم می‌شوند. در تکنیکهای پرسشی یک مجموعه از سئوالات و سناریوهای کیفی تعریف می‌گردند که پاسخ به آنها میزان مقبولیت معماری را مشخص می‌کند که خود به سه دسته تکنیکهای مبتنی بر پرسشنامه، لیست مرجع و سناریو تقسیم می‌گردند. که از این بین تکنیکهای مبتنی بر سناریو بدلیل پوشاندن تمام جوانب سیستمی و ارائه ارزیابی کاملی از معماری از عمومیت بیشتری برخوردارند بعنوان نمونه روش‌های زیر از این نوع هستند:

- *روش تحلیل ارزیابی معماری*[۱] *(ATAM)*: در این روش بعد از جمع آوری سناریوها و نیازهای کیفی نگرش‌های مختلف ذی‌نفعان نرم‌افزار اعمال شده و با استفاده از درختی موسوم به درخت سودمندی مصالحه ای بر درج سناریوها صورت می گیرد[۵،۷،۸،۲۱]. البته روش تحلیل هزینه - فایده[۲] (CBAM) که طبق عملکرد روش ATAM شکل گرفته است [۵،۸] و برای سنجش میزان بهره وری و ارزشمندی معماری بکار می رود نیز از همین نوع می‌باشد.

- *روش تحلیل معماری مبتنی بر سناریو*[۳] *(SAAM)*: در این روش بعد از جمع آوری و دسته بندی و اولویت دهی سناریوهای کیفی انواع معماری‌های کاندید برای نرم‌افزار مشخص می گردند سپس برای هر معماری درج سناریوها بررسی شده و نتایج ارزیابی می گردد در این بین اگر تعاملی نیز بین سناریوها وجود داشته باشد بررسی می گردد[۷].

مدل ارزیابی معماری نرم‌افزار[۴] *(SAEM)*: در این روش بعد از جمع‌آوری نیازها با استفاده از روش سئوال هدف[۵] (GQM) با ارائه سئوالهای هدفدار راه‌حلهای مناسب برای درج صفات انتخاب می گردند[۷].

- *تحلیل قابلیت اصلاح در سطح معماری (ALMA)*: این روش که بر ارزیابی صفات کیفی نرم‌افزار و میزان اصلاح پذیری نرم‌افزار تمرکز دارد پس از جمع‌آوری سناریوها و تعیین معماری‌های منتخب، سناریوهای ثابت و متغیر مشخص

می‌گردند و مورد ارزیابی قرار می گیرند و طبق نظر ذینفعان، سناریوهای منتخب اعمال می‌گردند[۷].

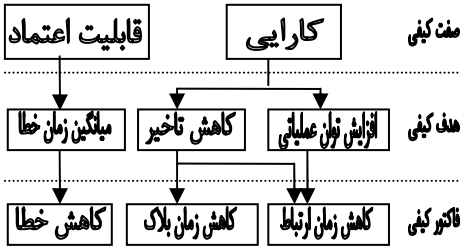
در تکنیکهای اندازه‌گیری، بعد از جمع آوری صفات و نیازهای کیفی، راه‌حل‌های معماری مطرح می‌گردند و طبق یکسری روش‌های وزن‌دهی مانند AHP[۶]، Trade-off و SMART[۷] وزن‌های مناسب به صفات کیفی نسبت داده می شوند. (البته در [۹] با شبیه سازی یک مسئله نمونه، ثابت کرده است که تفاوت خاصی در روش وزن‌دهی به صفات وجود ندارد و از هر روش استفاده شود تقریبا نتایج یکسانی در بردارد) سپس طبق روش‌های ریاضی و شبیه‌سازی سعی می‌شود معیارهای عددی حاصل گردد که نشان‌دهنده معماری مناسبتر باشد. مثلا در [۲] روشی بالا به پائین طبق سطوح مختلف تجرید از صفات ارائه شده است بطوری‌که تصمیم گیری در خصوص انتخاب راه‌حل‌های درج صفات کیفی برای یک معماری راحت‌تر و بواسطه معیارهای عددی سنجیده می شود. در این روش جدولی از صفات کیفی (سطرها) و راه‌حل‌ها (ستون‌ها) ایجاد می گردد و طبق مدلی تعریف شده با نام GAM[۸] وزن‌دهی می‌گردند. نتایج حاصل از جمع سطرهای هر ستون در وزن صفت، معیاری جهت تصمیم‌گیری اینکه کدام راه حل مناسب‌تر است ارائه می‌دهد. همچنین در [۱۰] روشی جهت انتخاب مناسبترین معماری از بین معماری‌ها ارائه شده سیستم نرم‌افزاری ارائه شده است که با دسته بندی و اولویت‌دهی به صفات طبق روش AHP و اعمال آنها بر معماری‌های ارائه شده نتایجی عددی جهت تصمیم گیری ارائه می‌گردد. در [۱۱] نیز روشی با نام ArchDesigner ارائه شده است که ابتدا انواع راه‌حل‌های معماری را درجه‌بندی و وزن‌دهی می‌کند و سپس مقادیر محاسبه شده را نرمال کرده و با اعمال وزن صفات به معیارهای عددی جهت انتخاب معماری مناسب می‌رسد. در این معیارها به فاکتورهای هزینه و زمان تولید نرم‌افزار نیز بعنوان صفات کیفی توجه خاص شده است. در [۱۲] با استفاده از روش AHP و مصالحه[۹] توانسته است روشی جهت ارزیابی معماری و خصوصا آزمایش حساسیت تصمیم در معماری نرم‌افزار ارائه کند. در [۷] روش مهندسی مجدد معماری سناریوگرا (SBAR) معرفی شده است که با استفاده از روش‌های مبتنی بر سناریو و ریاضی و متخصص انسانی، اسلوب وزن‌دهی به سناریوها در قالب جدول بیان شده است و با توجه به توانایی و یا عدم توانایی درج سناریو در یک معماری، به یک معیار عددی دست یافته است. در [۱۳] بر پایه یک روش احتمالی (توزیع مثلثی) روشی جهت برازش معماری نرم‌افزار ارائه داده و طبق نظر خبرگی انسانی و مقایسه نتایج حاصل هر معماری نتیجه گیری کرده است. در [۱۴] روشی آماری طبق خطوط رگرسیون ارائه شده است که میزان ارزشمندی راه‌حل‌های معماری برای تامین صفات کیفی را بررسی کرده و با ارائه یکسری فرمولهای حاصل از بررسی چندین معماری، روش پیشنهادی را عمومیت بخشیده است.

در روش های مبتنی بر شبیه سازی از انواع روش های شبیه سازی و شبکه های پتری [۱۵،۱۶] جهت ارزیابی معماری استفاده می گردد. در این‌گونه روش‌ها از شبکه‌های پتری برای به تصویر کشیدن ساختار و رفتار معماری نرم‌افزار استفاده می‌گردد و سپس با استفاده از روش‌های رسمی و یا شبیه سازی، عملکرد معماری بررسی شده و با تکرار این عمل بین معماری‌های مختلف معماری مناسب انتخاب می‌گردد بعنوان مثال مدل معماری نرم‌افزار[۱۰] (SAM) از این نوع است [۱۷].

علاوه بر موارد فوق یکسری روش‌های مبتنی بر هوش مصنوعی نیز برای ارزیابی معماری ارائه شده است که در این روش‌ها بر ایجاد سیستم‌های خبره برای تصمیم گیری در خصوص انتخاب معماری مناسب تاکید شده است. در این خصوص می توان به سیستم خبره ارزیابی نرم‌افزار(ESSE) اشاره کرد[۱۸].

۳- سلسله مراتب نیازهای کیفی

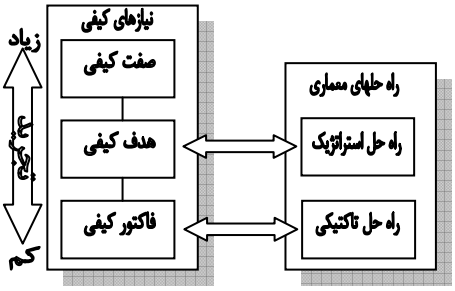
در [۱،۲] نیازهای کیفی نرم‌افزار بصورت سلسله مراتبی و در سه سطح تعریف شده است. بالاترین سطح نیازهای کیفی نرم‌افزار، صفات کیفی می‌باشد. مثلا کارایی صفتی کیفی است. هر صفت کیفی به تعدادی هدف کیفی تقسیم می گردد مثلا افزایش توان عملیاتی یا کاهش تاخیر یک هدف کیفی از صفت کیفی کارایی است. هر هدف کیفی نیز مرتبط با یک یا چند فاکتور کیفی است که پائین‌ترین سطح بیان صفات کیفی را معرفی می‌کند. مثلا کاهش زمان ارتباط، فاکتور کیفی است که با هدف کاهش تاخیر ارتباط مطرح می گردد و زیرمجموعه ای از صفت کارایی می‌باشد. شکل (۱) مراتبی از بیان نیازهای کیفی نرم‌افزار را نشان می‌دهد.



شکل ۱: سلسله مراتب بیان صفت، هدف و فاکتورکیفی از [۲]

در این مقاله بیان صفات کیفی در سطح متوسط یا در سطح هدف کیفی تعریف شده است و هرجا از صفت کیفی نام برده شده است منظور هدف کیفی بوده است.

برای رسیدن به صفت کیفی نیاز است که یک‌سری راه حل معماری منظور گردد که برای اهداف کیفی راه‌حل استراتژیک (SD) و برای فاکتورهای کیفی نیز راه‌حل تاکتیکی (TD) معرفی شده‌اند. شکل (۲) ارتباط راه‌حل‌ها را در سطوح تجریدی با صفات کیفی نشان می دهد.



شکل ۲: راه‌حل‌های معماری و نیازهای کیفی از [۲]

در این مقاله راه‌حل‌های استراتژیک برای اهداف کیفی در نظر گرفته شده است بطوری‌که هرجا از راه‌حل صحبت می‌شود راه‌حل در سطح استراتژیک مطرح است.

۴- نمایش ماتریسی صفات کیفی

در این مقاله رابطه بین صفات کیفی و راه‌حل‌های معماری بصورت ماتریس نشان داده شده است بطوری‌که سطرها، نشان دهنده صفات کیفی و ستون‌ها، نشان دهنده راه‌حل‌ها است. مقدار هر درایه ماتریس یک عدد بین ۹+ تا ۹- بوده که بر پایه روش AHP مقداردهی می گردد. در روش AHP میزان تاثیر راه‌حل در تامین صفت کیفی با یک عدد در بازه ۱+ تا ۹+ طبق جدول (۱) نمایش داده می‌شود عدد ۱ کمترین تاثیر مثبت و عدد ۹+ بیشترین تاثیر مثبت راه‌حل در تامین صفت کیفی را نشان می‌دهد. همان‌گونه که در [۶] به تناقض و تعامل بعضی از صفات کیفی اهمیت داده شده است بعنوان مثال افزایش کارایی باعث کاهش اصلاح‌پذیری می‌گردد بنابراین در این مقاله بازه ۱- تا ۹- برای معرفی تاثیر منفی راه‌حل بر صفت معرفی شده است، بطوری‌که ۱- کمترین تاثیر منفی و ۹- بیشترین تاثیر منفی یک راه‌حل بر یک صفت را نشان می‌دهد.

جدول ۱: تخصیص ارزش طبق روش AHP

میزان تاثیر عددی	میزان تاثیر حرفی
۱+ تا ۹+	تامین صفت از کم به زیاد با تاثیر مثبت
۱- تا ۹-	تامین صفت از کم به زیاد با تاثیر منفی

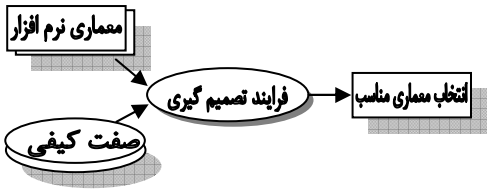
از طرفی هر صفت نرم‌افزار دارای یک وزن است که عددی بین ۰ تا ۱ بوده و نشان دهنده میزان اهمیت آن صفت در نرم‌افزار است. همچنین مجموع وزن تمام صفات نرم‌افزار برابر ۱ است. (وزن صفات نرمال شده‌اند)

جدول ۲: نمایش ماتریسی صفات کیفی و راه‌حل‌های معماری

	راه‌حل ۱	راه‌حل ۲	.....	راه‌حل m
صفت ۱	[-۹,+۹]	.....	.....	.....
صفت ۲	.....			
.....	.....			
صفت n	.....			

۵- شرح مسئله

اگر فرض شود که هر نرم‌افزار نیازمند n صفت کیفی باشد و هر معماری نرم‌افزار m راه‌حل (تصمیم) برای تامین صفات پیشنهاد کند بطوری‌که هر راه‌حل بتواند بر یک یا چند راه‌حل تاثیر مثبت داشته و بر یک یا چند صفت تاثیرمنفی بگذارد آنگاه می‌خواهیم معیاری جهت انتخاب یک معماری از میان p معماری پیشنهادی ارائه کنیم و به فرمولی جهت برازش معماری نرم‌افزار دست یابیم بطوری‌که معیارعددی تصمیم گیرنده مناسب بودن معماری نرم‌افزار باشد. شکل (۳)، الگوی کلی پیشنهادی را جهت رسیدن به این هدف نشان می دهد. همانطور که مشاهده می گردد انواع معماری‌های ممکن برای یک نرم‌افزار در یک فرآیند تصمیم گیری و بر اساس میزان مناسب بودن راه‌حل‌های معماری در تامین صفات کیفی نرم‌افزار بررسی شده و به یک عدد یا برازش که معیاری جهت انتخاب معماری مناسب باشد ختم می‌گردد. با مقایسه اعداد برازشی بین معماری‌های مختلف، معماری بهتر قابل انتخاب خواهد بود.



شکل۳ : الگوی کلی رسیدن به معماری مناسب

۵-۱- معرفی نمادهای استفاده شده

در این قسمت ابتدا نمادهای مورد استفاده در این مقاله معرفی شده اند.

جدول ۳: نمادهای استفاده شده در این تحقیق

n	تعداد صفات کیفی
m	تعداد راه‌حل‌های (تصمیم‌های) معماری
p	تعداد معماری‌های پیشنهادی
W	ماتریس $1*n$ وزن صفات کیفی نرم‌افزار

W^T	ماتریس ترانهاده W
W_k	وزن صفت k ام
X_i	ماتریس $n*m$ میزان تامین صفات کیفی با استفاده از راه‌حل‌های معماری I ام
X_{ij}	بردار j ام ماتریس X_i (ارزش راه‌حل j ام بر صفات کیفی بدون احتساب وزن صفات را نشان می دهد)
G_i	ماتریس $1*m$ که میزان اهمیت راه‌حل‌های معماری i ام بر نرم‌افزار را نشان می‌دهد.
G_{ix}	درایه x ماتریس G_i
μ	ماتریس میانگین G_i ها
μ_x	درایه x ماتریس μ
K	ماتریس کوواریانس راه‌حل‌های معماری‌های پیشنهادی
σ_{ij}	کواریانس دو بردار G_i و G_j
$f(G_i)$	چگالی ارزش اعمال معماری i ام بر نرم‌افزار

۲-۵- راه‌حل پیشنهادی

با توجه به اینکه هر راه‌حل دارای ارزشی در تامین صفت کیفی است حاصل‌ضرب ارزش راه‌حل‌ها در وزن صفات، ماتریس G_i را نتیجه می‌دهد.

$$G_i = W^T X_i \qquad (۱)$$

ماتریس G_i میزان اهمیت یا ارزش راه‌حل‌های معماری i ام را نشان می دهد و آنرا بردار ارزش می نامیم. هر درایه آن ارزش یک راه‌حل را در تامین کلیه صفات کیفی نشان می‌دهد و هر چه مقدار بیشتری داشته باشد چگال‌تر (ارزشمندتر) است. اگر ماتریس G_i برای چندین معماری مختلف (i های متفاوت) و برای یک نرم‌افزار محاسبه گردد با برازش آنها هدف این تحقیق که یافتن بهترین G_i می‌باشد محقق می‌گردد.

اگر میانگین G_i ها را معماری میانگین μ بنامیم ماتریس ارزش میانگین طبق فرمول (۲) محاسبه می‌گردد. کوواریانس یا اختلاف از میانگین هر دو درایه G_i در ماتریس کوواریانس K قرار می گیرد و طبق فرمول (۳) و (۴) محاسبه می‌گردد.

$$\mu = \begin{bmatrix} E(G_{i1}) \\ \dots \\ E(G_{im}) \end{bmatrix} \qquad (۲)$$

$$K = \begin{bmatrix} \sigma_{11} & \dots & \sigma_{1m} \\ \dots & \dots & \dots \\ \sigma_{m1} & \dots & \sigma_{mm} \end{bmatrix} \qquad (۳)$$

$$\sigma_{ij} = Cov(G_i, G_j) \qquad (۴)$$

چگالی احتمال G_i را برازش معماری G_i تعریف کرده و با فرمول (۵) محاسبه شده است.

$$f(G_i) = \frac{1}{(2\pi)^{\frac{m}{2}} |\det K|^{\frac{1}{2}}} e^{[-\frac{1}{2}(G_i-\mu)^T K^{-1}(G_i-\mu)]} \qquad (۵)$$

در فرمول (۵)، G_i با فرمول (۱) محاسبه می‌شود، K^{-1} ماتریس معکوس K ، $|\det K|$ قدرمطلق دترمینان ماتریس K هستند. همچنین $f(G_i)$ یک عدد است که ارزش اعمال معماری i ام بر نرم‌افزار را نشان می‌دهد. بعبارت دیگر $f(G_i)$ میزان مناسب بودن معماری را نشان می‌دهد که آنرا برازش معماری می‌نامیم و هرچه بزرگتر باشد معماری چگال‌تر خواهد بود.

۳-۵- اثبات توزیع نرمال G_i ها

اگر فرض شود X_{ij} ها یا هر بردار j یک نمونه تصادفی باشد، هر درایه این بردار دارای توزیع یکنواخت در بازه (۹+ و ۹-) می‌باشد پس X_{ij} ها به‌عنوان متغیرهای تصادفی n متغیره دارای توزیع یکنواخت می‌باشند. بنا به قضیه حد مرکزی [۱۹]، هر درایه G_{ix} که طبق فرمول (۱) حاصل می‌گردد متغیر تصادفی است که از حاصل‌جمع متغیرهای مستقل و هم‌توزیع حاصل شده است. بنابراین دارای توزیع نرمال خواهد بود پس طبق تعریف بردار نرمال، G_i ها به ازاء تمام i ها، ماتریس‌های مستقل و دارای توزیع احتمالی نرمال و با میانگین μ_G بوده و از فرمول (۶) محاسبه می گردد:

$$\mu_G = \frac{1}{p}(G_1 + G_2 + \dots + G_p) \qquad (۶)$$

بنابراین می توان μ_G را یک برآوردگر μ درنظر گرفت یعنی از آنجا که μ مشخص نیست اگر به ازاء هر نرم‌افزار بردارهای ارزش $G_1, G_2, \dots, G_p$ محاسبه شود طبق فرمول (۶) معماری میانگینی حاصل خواهد شد که می تواند نقش μ را در فرمول (۵) ایفا کند. با محاسبه K بر پایه مقادیر σ_{ii} و σ_{ij} از فرمول‌های (۷) و (۸) و قرار دادن آن در فرمول (۵) برازش معماری i ام به ارزش $f(G_i)$ قابل دسترسی خواهد بود.

$$\sigma_{ii} = \frac{1}{p-1}\sum_{x=1}^{p}(G_{xi} - \mu_i)^2 \qquad (۷)$$

$$\sigma_{ij} = \frac{1}{p}\sum_{x=1}^{p}\left|(G_{xi}-\mu_i)(G_{xj}-\mu_j)\right| \qquad (۸)$$

توجه شود که شرط استفاده از فرمول (۵)، داشتن توزیع نرمال G_i هاست. طبق خاصیت توزیع نرمال [۱۹]، چگالی احتمال متغیرهای تصادفی که به میانگین نزدیک‌ترند، بیشتر است بنابراین می‌توان انتظار داشت که بردار ارزشی چگالی احتمال بیشتری دارد که به بردار میانگین نزدیک‌تر باشد. بنابراین برای برازش معماری، آن معماری که به معماری میانگین نزدیکتر است بیشترین چگالی را بعد از معماری میانگین خواهد داشت و جواب مسئله خواهد بود.

۶- مطالعه موردی یک مسئله

یک مسئله تجربی برای برازش معماری‌های پیشنهادی بررسی شده است تا جوانب مختلف روش ارائه شده بیشتر مورد ارزیابی قرار گیرد. این مسئله مربوط به یک شرکت تولیدکننده نرم‌افزار است که هم اکنون در حال تولید یک چارچوب برای تولید نرم‌افزاری تجاری – صنعتی است. این نرم‌افزار که خود برای تولید نرم‌افزار بکار برده می شود از امکانات مختلفی همچون چارچوب مولفه بنیاد و مجموعه ای غنی از مولفه‌های کاربردی استفاده می‌کند[۲۳]. از آنجایی که تجربه تولید این نرم‌افزار وجود دارد با مروری مجدد بر معماری آن سعی شده است که تحلیلی بر معماری‌های ممکن این نرم‌افزار صورت گیرد.

۶-۱- صفات کیفی نرم‌افزار

از آنجایی که نرم‌افزار [۲۳] خود برای تولید نرم‌افزارهای دیگر بکار می رود یکسری صفات کیفی به شرح ذیل مدنظر می‌باشد که سعی شده است معماری‌های پیشنهادی، بر اساس امکان تحت پوشش قرار دادن صفات کیفی ارائه شوند:

قابلیت انعطاف از حیث عملکرد: از آنجایی که نرم‌افزار بایستی بتواند برای عملکردهای مختلف انعطاف داشته باشد و تغییرات در حوزه نیازهای عملکردی را بپذیرد. همچنین قابلیت افزودن نیازهای جدید و یا جایگزینی با نیازهای قدیمی را داشته باشد نیازمند استفاده از معماری‌هایی است که بتواند این هدف را برآورده کند. این صفت کیفی را A۱ می‌نامیم، راه‌حل معماری که می توان برای این صفت در نظر گرفت، جداسازی ارائه سرویس از داده می‌باشد و آنرا D۱ می‌نامیم.

کارایی از حیث سرعت اجراء: از آنجایی که این نرم‌افزار برای تولید نرم‌افزارهای تجاری که عموما در محیط شبکه فعالیت می‌کنند مورد استفاد قرار می گیرند و برای این نرم‌افزارها سرعت اجرای پردازه‌ها و زمان پاسخ پرس‌وجوها از اهمیت خاصی برخوردار است در این راستا پشتیبانی از معماری‌هایی که روش‌های ارتباطی مناسبی برای مولفه‌های شبکه معرفی می‌کنند و یا حتی از الگوریتم‌ها و نرم‌افزارهای کنترلی خاصی تبعیت می‌کنند از ارجحیت برخوردارند. این صفت را A۲ می نامیم. راه‌حلی که می‌توان برای تامین این صفت پیشنهاد داد تکرار مولفه‌های دسترسی به داده در مکان‌های مختلف است که آنرا با D۲ می شناسیم یا استفاده از ریسمان برای پردازش موازی دستورات است، که D۳ می نامیم.

امنیت دسترسی کاربران: در سیستم‌های نرم‌افزاری تجاری امنیت فاکتوری اساسی در اطمینان کاربران محسوب می‌گردد و مولفه‌هایی که سعی در ارائه دسترسی به کاربران و یا مخفی‌سازی برخی از داده‌ها دارند بایستی در سطح کلان معماری مورد بررسی دقیق قرار گیرند. این صفت را A۳ می‌نامیم و ایجاد مولفه‌هایی برای کنترل دسترسی به بانک اطلاعاتی و سیستم عامل را به‌عنوان راه‌حل پیشنهادی D۴ می‌نامیم.

هزینه زمانی: در تولید نرم‌افزار تجاری، هزینه تولید از حیث زمان تولید نرم‌افزار بسیار مورد توجه است بطوری‌که زمان عرضه به بازار برای بسیاری از سیستم‌های بزرگ در حد محدودی قابل تعریف است بنابراین معماری‌هایی که جمع‌آوری و مونتاژ مولفه‌های آماده را توصیه می‌کنند، محبوب‌ترند. این صفت کیفی را A۴ می‌نامیم و استفاده از بسته‌ها و مجموعه کدهای آماده را بعنوان راه‌حل پیشنهاد کرده و D۵ می نامیم.

۶-۲- ارائه معماری‌های پیشنهادی

برپایه صفات کیفی بیان شده، ۳ نوع معماری پیشنهاد شده است که طبق تجربه نویسندگان و مشورت با افراد خبره برای هر معماری، ماتریس ارزش مطابق جدول (۳) برای هر معماری ارائه شده است. معماری ۱، بر پایه قابلیت‌هایی است که معماری J۲EE پیشنهاد می‌کند. معماری ۲ بر اساس قابلیت‌های VB.NET و معماری ۳ بر اساس معماری Delphi Enterprise می‌باشد.

همچنین برای صفات بیان شده، ماتریس W معرف وزن صفات می باشد. این مقادیر طبق صلاحدید و اهمیت صفات کیفی در نرم‌افزار انتخاب شده اند.

$$W^T = [0.3 \quad 0.2 \quad 0.2 \quad 0.3]$$

جدول۳: ماتریس ارزش معماری‌های کاندید

	معماری ۱					معماری ۲					معماری ۳				
	D۱	D۲	D۳	D۴	D۵	D۱	D۲	D۳	D۴	D۵	D۱	D۲	D۳	D۴	D۵
A۱	۷	۷	۵	۴	۵	۵	۳	۳	۴	۳	۵	۴	۴	۴	۵
A۲	۰	۵	۳	۲	-۳	-۶	۱	۳	۲	-۴	-۵	۳	۴	۲	-۲
A۳	-۱	-۲	۱	۵	-۲	-۵	-۳	-۱	۲	-۲	-۳	-۳	۰	۳	-۱
A۴	-۶	-۵	-۴	-۲	۶	-۵	-۲	-۳	-۴	۳	-۴	-۱	-۲	-۳	۴

۳-۶- حل مسئله موردی:

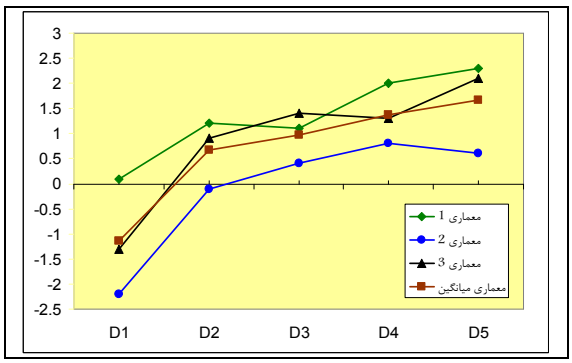
ابزارهایی که برای حل این مسئله و رسم نمودارهای لازم مورد استفاده قرار گرفته اند عبارتند از:

- نرم‌افزار Mathcad ۲۰۰۰ تولید شرکت MathSoft که برای محاسبه فرمول‌های ریاضی و ماتریسی مورد استفاده قرار گرفته است[۲۴].
- نرم‌افزار Excel ۲۰۰۳ تولید شرکت میکروسافت که برای رسم نمودارهای پراکندگی مورد استفاده قرار گرفته است[۲۵].

برای حل مسئله موردی ابتدا مقدار G_i ها با استفاده از فرمول (۱) محاسبه شده اند سپس مقدار معماری میانگین μ طبق فرمول (۲) محاسبه شده است.

$$G_1 = [0.1 \quad 1.2 \quad 1.1 \quad 2 \quad 2.3]$$
$$G_2 = [-2.2 \quad -0.1 \quad 0.4 \quad 0.8 \quad 0.6]$$
$$G_3 = [-1.3 \quad 0.9 \quad 1.4 \quad 1.3 \quad 2.1]$$
$$\mu = [-1.133 \quad 0.667 \quad 0.967 \quad 1.367 \quad 1.667]$$

برپایه مقدارهای محاسبه شده، نمودار مقادیر G_i ها یا ارزش راه‌حل‌های پیشنهادی هر معماری به همراه معماری میانگین μ در شکل (۴) رسم شده‌است. همانطور که مشاهده می‌گردد معماری ۱ بیشترین تفاضل مثبت نسبت به معماری میانگین دارد همچنین معماری ۳ دارای نزدیک‌ترین مقادیر به معماری میانگین می‌باشد و معماری ۲ بیشترین تفاضل منفی نسبت به معماری میانگین دارد.



شکل ۴: نمودار ارزش راه‌حل‌های پیشنهادی هر معماری

اگر مقادیر K و $f(G_i)$ ها طبق فرمول‌های (۵) و (۲) محاسبه گردند مقادیر زیر نتیجه می شوند.

$$K = \begin{bmatrix} 1.343 & 0.505 & 0.28 & 0.466 & 0.664 \\ 0.505 & 0.463 & 0.202 & 0.263 & 0.419 \\ 0.28 & 0.202 & 0.263 & 0.145 & 0.292 \\ 0.466 & 0.263 & 0.145 & 0.363 & 0.345 \\ 0.664 & 0.419 & 0.292 & 0.345 & 0.863 \end{bmatrix}$$

همچنین مقدار چگالی ارزش هر یک از معماری‌های فوق به شرح زیر خواهد بود:

جدول ۴: چگالی ارزش معماری‌های کاندید

$f(G_1) = 0.076$	$f(G_2) = 0.068$	$f(G_3) = 0.084$
$f(\mu) = 0.161$		

همانطور که مشاهده می‌گردد معماری ۳ دارای بالاترین چگالی ارزش و ۰/۰۸۴ و معماری ۲ دارای کمترین چگالی می‌باشد. بنابراین معماری ۳ بهترین انتخاب از بین ۳ معماری پیشنهادی است.

۷- بحث و بررسی

با توجه به راه‌حل ارائه شده و حل مسئله موردی، نتایج زیر قابل بررسی و موشکافی است:

۱- با توجه به جدول (۴)، معماریی دارای بالاترین چگالی است که راه‌حل‌های آن به راه‌حل‌های معماری میانگین نزدیکتر باشد و این مورد با توجه به خاصیت توزیع نرمال دور از انتظار نبوده و می‌تواند اثباتی بر نرمال بودن توزیع احتمال راه‌حل‌های معماری باشد.

۲- همان‌طور که در شکل (۴) مشاهده می‌گردد معماری ۱ دارای بیشترین تغییرات مثبت نسبت به میانگین است این موضوع باعث می‌شود که روش‌های بیان شده در [۲،۱۱،۲۰] معماری مناسب‌تر را معماری ۱ انتخاب کنند. زیرا روش‌های فوق میزان تفاوت مثبت هر معماری را نسبت به معماری

میانگین معیار قرار داده اند در صورتی که تکیه بر این ایده‌ها باعث می‌شود صفاتی که دارای وزن بیشتری هستند بر جواب تاثیر زیادی بگذارند و راه‌حل معماری هرچند دارای ارزش کمی باشد ولی وزن زیادی پیدا می‌کند و باعث می شود معماری را توجیه‌پذیر کند در حالی‌که در روش پیشنهادی تمام راه‌حل‌ها بر اساس میزان ارزش و تفاضل از میانگین برازش می‌شوند و تاثیر وزن صفات متعادل شده، معماری برتر انتخاب می‌گردند.

۳- مقادیر برازش معماری‌های پیشنهادی در جدول (۴)، دارای تفاوت‌های بسیار کوچکی می باشند درحدود ۰،۱، که نزدیک به‌نظر می‌رسند. اما با توجه به واریانس حاصل ($|K|^{\frac{1}{2}}$) که حدود ۰،۰۷ می‌باشد رنج تغییرات منطقی به‌نظر می‌رسد بنابراین با توجه به رنج محدود واریانس، مجموعه جواب‌های معماری در بازه کوچکی قرار می‌گیرند و در نتیجه نتایج قابل قبول‌تری را نیز تولید می‌کنند.

۴- در این روش افزایش تعداد معماری‌ها باعث بهتر شدن پاسخ می‌گردد زیرا افزایش تعداد معماری‌ها، نمودار پراکندگی معماری‌ها را به سمت توزیع نرمال دقیق‌تر پیش می‌برد درحالیکه در روش‌های ارائه شده در [۱۳،۱۴] افزایش تعداد معماری‌ها باعث پیچیدگی عملیات برازش شده و در نتیجه دقت کاهش می یابد.

۸- نتیجه گیری و کارهای آینده

با توجه به قسمت ۲ این روش بدلیل ماهیت احتمالی، در گروه روش‌های مبتنی بر اندازه گیری عددی قرار می‌گیرد بطوری‌که با دسته بندی ماتریس ارزش راه‌حل‌ها و تاثیر وزن صفات بر آنها و برازش بردارهای حاصل بر اساس توزیع احتمالی نرمال، معماری بهتر تشخیص داده می‌شود. بر این اساس دستاوردهای زیر قابل بیان است که می‌تواند سرلوحه کارهای آینده قرار گیرد:

۱- تهیه نرم‌افزاری هوشمند از انواع معماری‌های ممکن، بطوری‌که بیشترین صفات کیفی را شامل شود و براساس روش ارائه شده برازش راه‌حل‌ها و معماری‌های مورد نظر را برای تشخیص یک معماری مبنا محاسبه کند.

۲- با مبنا قراردادن این روش می توان میزان ماتریس ارزش صفات کیفی معماری مجهولی را محاسبه کرد برای این منظور میتوان تابع معکوس $f(G_i)$ را محاسبه کرد و ماتریس ارزش را با استفاده از روش‌های عددی بدست آورد و بر این اساس معماری را پیشنهاد داد که حداقل ارزش‌های معماری محاسبه شده را داشته باشد.

سپاس‌گزاری

جا دارد از کمک‌های ارزنده جناب آقای مهندس حسین کفاش سپاسگزاری کنیم.

مراجع

[۱] J.A. McCall, "*Quality factors*", in Encyclopedia of Software Engineering, J.L. Marciniak (ed.), John Wiley & Sons, New York, pp. ۹۵۸-۹۶۹, ۱۹۹٤.

[۲] K. LEE, "*A top-down approach to Quality driven architecture engineering of software systems*", IEICE TRANS.INF.& SYST, Vol.E۸۸-D, NO.۱۲, December ۲۰۰۵.

[۳] J.Bosch, "*Design & Use of Software Architectures: Adopting and evolving a product-line approach*", Addison-Wesley, ۲۰۰۰.

[٤] L. Lundberg and et al, *"Quality Attributes in Software Architecture Design"*, Proceedings of the IASTED ۳rd International Conference on Software Engineering and Applications, October, ۱۹۹۹.

[٥] Len Bass, Paul Clements, Rick Kazman, "*Software Architecture in Practice",* Second Edition, Addison Wesley, ۲۰۰۳

[٦] L. Hohmann, "*Beyond Software Architecture: Creating and Sustaining Winning Solutions*", Addison Wesley, ۲۰۰۳.

[۷] L. Dobrica, E. Niemela, "*A survey on software architecture analysis methods*", IEEE Transaction on software engineering, Vol. ۲۸, No. ۷, July ۲۰۰۲.

[۸] R. Kazman, J. Asundi, M. Klein, "*Quantifying the Costs and Benefits of Architectural Decisions*", ICSE, p۲۹۷, ۲۳rd International Conference on Software Engineering (ICSE'۰۱), ۲۰۰۱.

[۹] M. Poyhonen, R.P. Hamalainen, "*On the convergence of multiattribute weighting methods*", European Journal of Operational Research ۱۲۹, pp. ٥٦۹-٥۸٥, ۲۰۰۱.

[۱۰] M. Svahnberg, C. Wohlin, L. Lundberg, M. Matsson, "*A quality-driven decision-support method for identifying software architecture candidates*", International Journal of Software Engineering and Knowledge Engineering, Vol. ۱۳, No. ٥, pp. ٥٤۷-٥۷۳, ۲۰۰۳.

[۱۱] T. Al-Naeem۱, I. Gorton, M.A. Babar۱, F. Rabhi and B. Benatallah, "*A quality-driven systematic approach for architecting distributed software applications*", In Proceedings of the ۲۷th International Conference on Software Engineering (ICSE), St. Louis, USA, ۲۰۰٥.

[۱۲] L. Zhu, A. Aurum, I. Gorton, R. Jeffry, "*Tradeoff and sensitivity analysis in software architecture evaluation using analytic hierarchy process*", Software Quality Journal, ۱۳, pp. ۳٥۷–۳۷٥, ۲۰۰٥.

[۱۳] T. Rosqvist, M.Koskela, Hannuarju, "*Software Quality Evaluation Based on Expert Judgment*", Software Quality Journal, ۱۱, pp. ۳۹–٥٥, ۲۰۰۳.

[۱٤] F.Liu, K. Noguchi, A. Dhungana, V. Srirangam, A.P. Inuganti, "*A quantitative approach for setting technical targets based on impact analysis in software quality function deployment (SQFD)*", Software Quality Journal ۱٤:pp. ۱۱۳–۱۳٤, ۲۰۰٦.

[۱۵] W. Wenxin, M. Saeki, "*A Technique to Specify Software Architectures Based on Colored Petri Nets*", ۱۱th International Conference on Software Engineering & Knowledge Engineering , pp. ۱۵۲-۱۶۱, ۱۹۹۹.

[۱۶] K. Fukuzawa, M. Saeki, "*Evaluation Software Architecture by Coloured Petri Nets*", The ۱٤th international conference on Software engineering and knowledge Engineering, July ۱۵-۱۹, ۲۰۰۲, Ischia, Italy. ACM, pp. ۲٦۳-۲۷۰, ۲۰۰۲.

[۱۷] X. He, Y. Deng, "*A Framework for developing and analyzing software architecture specification in SAM*", biritish computer society, the computer journal vol.٤٥, NO.۱, ۲۰۰۲.

[۱۸]I. Vlahavas, I. Stamelos, I Refandis, A.Tsoukias, "*ESSE: An Expert system for software evaluation*", Knowledge-based system, ۱۲, pp. ۱۸۳-۱۹۷, ۱۹۹۹.

[۱۹] A. Papoulis, S. Pillai, *"Probability Random Variables and Stochastic Processes"*, ٤th Edition, McGraw- Hill, ۲۰۰۲.

[۲۰] M. Lindvall, R.T. Tvedt, P. Costa "*An Empirically-Based Process for Software Architecture Evaluation*", Empirical Software Engineering, ۸, ۸۳–۱۰۸, ۲۰۰۳.

[۲۱] R. Kazman, L. Bass, M. Klein, T. Lattanze, L. Northrop, "*A Basis for Analyzing Software Architecture Analysis Methods*", Software Quality Journal, ۱۳, pp. ۳۲۹–۳۵۵, ۲۰۰۵.

[۲۲] A. Andrews, E. Mancebo, Perruneson, R. France, "*A Framework for Design Tradeoffs*", Software Quality Journal, ۱۳, ۳۷۷–٤۰۵, ۲۰۰۵.

[۲۳] مجید وفایی جهان، "ارائه یک معماری مولفه بنیاد توزیع شده تجاری (BizMaster) برای تولید نرم‌افزارهای اطلاعات مدیریت (MIS)"، ۹ امین کنفرانس انجمن کامپیوتر ایران، تهران، ۱۳۸۲ ص ۳۷۳-۳۸۱.

[۲٤] The Microsoft Excel ۲۰۰۳ as statistical tool at: www.microsoft.com .

[۲۵] The Mathcad ۲۰۰ professional from MathSoft Inc, as Mathematical tool at www.Mathsoft.com .

زیرنویس‌ها

---

۱ - Architecture Trade-off Analysis Method

۲ - Cost Benefit Analysis Method

۳ - Scenario Based Architecture Analysis Method

٤ - Software Architecture evaluation model

۵ - Goal Question Metric

٦ - Analytic Hierarchy Process

۷ - Simple multi attribute rating technique

۸ - Goal Analysis Method

۹ - Trade-off

۱۰ - Software Architecture Model