استفاده از محدودیت های معماری برای به دست آوردن استفاده مجدد اجزا نرم افزار در اضافه کردن و بهبود ویژگی ها

چکیده

ما سیستم های یکپارچه نرم افزار و سخت افزاری قطعات با طول عمر اعم از 10-15 سال را توسعه می دهیم. در طول عمر این سیستم ها، نیازهای بازار به طور قابل توجهی با توجه به پیشرفت های تکنولوژیکی، نیازهای زیست محیطی، و اولویتهای فرهنگی تغییر کرده است. هزینه تغییر نرم افزار و سخت افزار یک محرک بسیار بزرگ است که اغلب منجر به معرفی تغییری می شود که در نتیجه منجر به معرفی تغییر در نرم افزار برای روبه رو شدن با تکامل انتظارات بازار خواهد شد. بزرگترین مزیت نرم افزار-، سازگاری آسان – است و همچنین بزرگترین نقطه ضعف آن، متعد بودن برای تغییر است. از این رو، طراحی نرم افزار به ویژه در توسعه نرم افزار در سطح جهانی (GDSD) بسیار چالش برانگیز است. در این مقاله تمرینی، ما رویکردمان را از اعمال نفوذ محدودیت های معماری نرم افزار و چالش مواجه شده به همراه درس های آموخته شده را به اشتراک میگذاریم، استفاده مجدد نرم افزار در هنگام اضافه کردن و بهبود ویژگی ها هم باعث کاهش هزینه های کلی و هم باعث کاهش زمان برای رسیدن محصول به بازار برای یک تیم گسترش نرم افزار توزیع شده خواهد بود.

کلمات کلیدی: قابلیت استفاده مجدد نرم افزار، محدودیت های معماری؛ چالش معماری؛ مبادلات؛ گسترش نرم افزار توزیع شده

1. سابقه

سیستم های یکپارچه ای که ما به طور مستمر در حال توسعه آن ها هستیم در حال فشرده سازی نرم افزار بیشتر هستند، در حالی که در گذشته آنها فشرده تر بود. علاوه بر این، پس از چند سال، سیستم ما در حال تحول است و همچنین در حال رشد در هر دو بعد اندازه و پیچیدگی می باشد.

بنابراین معماری نرم افزار نقش رهبر را بازی می کند و آن را تبدیل به یک محصول مرکزی در چرخه زندگی سیستم های ما می نماید، چرا که آنها یک مرور کلی از سازمان و از این سیستم ها برای ارائه به ذینفعان مختلف را انجام می دهند. معماری نرم افزار به صورت زیر تعریف شده است "مجموعه ای از ساختارهای یک سیستم نرم افزاری، که برای استدلال در مورد آن ... لازم است (و) از بخش های نرم افزاری تشکیل شده است و همچنین به صورت روابط میان آنها و خواص این اشخاص و روابط" تعریف شده است[1]. معماری نرم افزار باعث می شود تا هر دو اجزای یک سیستم نرم افزاری و وابستگی بین این اجزا به صورت صریح باشد [2]. در سال 1990 اصطلاح "معماری نرم افزار" شروع به جلب توجه قابل توجهی هم از جامعه پژوهش و از صنعت نمود [3]. چالش ایجاد ارزیابی و حفظ این سیستم بزرگ تا حد زیادی باعث تحریک و رشد در زمینه معماری شده است. اهمیت معماری نرم افزار برای سیستم های نرم افزاری بزرگ و پیچیده را می توان با دلایل زیر توضیح داد[4]:

- **ارتباطات متقابل:** اکثر ذینفعان سیستم ها می توانند از معماری نرم افزار به عنوان پایه برای درک سیستم، به صورت اجماع، و برقراری ارتباط با یکدیگر استفاده کنند.
- **تصمیم گیری اولیه طراحی:** معماری نرم افزار اولین مصنوعی است که اولویت ها را در میان نگرانی های رقیب برای تجزیه و تحلیل قادر می سازد. چنین نگرانی هایی شامل مبادلات بین جنبه های عملکردی و غیر عملکردی مانند عملکرد، امنیت، نگهداری و اصلاح است. نگرانی های رقابتی اضافی را می توان هزینه توسعه در حال حاضر در مقابل هزینه های آینده نگهداری و یا تکمیل کاربردی و یا مبادلات بین جنبه های فنی و غیر فنی مانند بازار و یا بودجه در نظر گرفت.

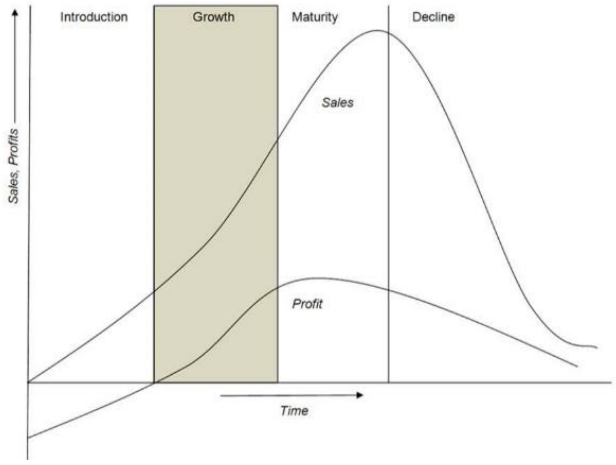
• **انتزاع قابل انتقال از یک سیستم:** مدل معماری نرم افزار انتقال در سراسر سیستم قابل انتقال است. به طور خاص، می توان آن را به سیستم های مشابه دیگر اعمال نیز اعمال نمود و استفاده مجدد از آن ها در مقیاس بزرگ را ترویج داد.

در حالی که سیستم های نرم افزاری در حال تحول هستند، تصمیم گیری ها و اصول معماری نیز در پی آن ها دنبال شوند و گسترش پیدا کنند. تغییر تصمیم گیری معماری دشوار است و نیاز به یک تجزیه و تحلیل و تاثیر دقیق و در نظر گرفتن دلایل تغییرات گذشته دارد. بنابراین مستندسازی تصمیمات معماری از جمله استدلال برای تصمیم گیری یک فعالیت مهم در فرآیندهای توسعه نرم افزار [5] بسیار مهم است. محدودیت های معماری در میان مهمترین توصیفات هستند و در میان اسناد مهم تصمیم گیری هستند. [7]

محدودیت معماری نشان دهنده مشخصات یک حالتی است که یک معمار باید به منظور جلب رضایت یک تصمیم در مورد آن بگیرد. بنابراین محدودیت معماری نقش مهمی در تصمیم گیری های طراحی و اعتبار سازی معماری بازی می کند. هنگام طراحی معماری نرم افزار، تصمیمات و محدودیت نیازها به اهداف کسب و کار متصل می شود. تصمیم گیری های طراحی اغلب به دلایل غیر فنی گرفته شده اند: نگرانی استراتژیک کسب و کار، رویارویی با محدودیت های هزینه و برنامه، استفاده از پرسنل موجود، و غیره [8].

2. مقدمه

به طور معمول طول عمر سیستم های یکپارچه ما در محدوده 10-15 سال است. در طول عمر خود یک سیستم از چهار مرحله مختلف (1) مقدمه (2) رشد، (3) بلوغ، و (4) کاهش عبور خواهد کرد.



شکل 1. طول عمر محصول(13)

نیازهای بازار برای سیستم های یکپارچه در حال تغییر قابل توجهی به دلیل پیشرفت های زیست محیطی، فرهنگی، و فن آوری است. هزینه تغییر یک سیستم یکپارچه بستگی به مرحله چرخه عمر آن دارد. فاز رشد برای محصولات ما مهم و چالش برانگیز است و دو دلیل دارد: 1) این محصول در حضور رقابت تشدید باید با نیاز بازار پیش برود و ویژگی های یک محصول جدید نیز به آن اضافه شود. 2) محصول تحت یک انتقال با توجه به جریان زیاد نیاز به این محصول مورد نیاز است.

ما بر روی پلت فرم خط تولید کار میکنیم که در فاز رشد و توسعه یافته در سراسر 3 قاره (اروپا، آسیا، و شمال امریکا) کار می کنند. ما با ارائه برنامه های متعدد بر اساس این پلتفرم می پردازیم و به طور مداوم دامنه خط تولید را گسترش می دهیم. از این رو مهم است که یک ساختار در محل که به عنوان یک راهنما برای اضافه کردن ویژگی های جدید برای تمام تیم ها در سراسر جغرافیا داشته باشیم.

در بخش بعدی ما جزئیات چنین ساختاری را همراه با چالش هایی که ما با آن مواجه شده ایم و درس هایی که آموختیم را ارائه می کنیم.

3. چهار رکن

ما یک رویکرد با چهار رکن را برای استفاده از محدودیت های معماری برای نرم افزار استفاده مجدد طراحی کرده ایم در حالی که ویژگی هایی را افزودیم و یا اینکه آن ها را ارتقا دادیم. این چهار رکن عبارتند از: (A) تجزیه و تحلیل مهندسی مورد نیاز بر اساس همین دلیل در مقابل تجزیه و تحلیل (B) نوع، یک رویکرد کارآمد برای رسیدن بهترین گزینه مناسب طراحی (C) تصمیم گیری معماری های کلیدی و تجزیه و تحلیل موازنه برای استفاده مجدد موثر، (د) نرم افزار طرح استفاده مجدد از اجزاء.

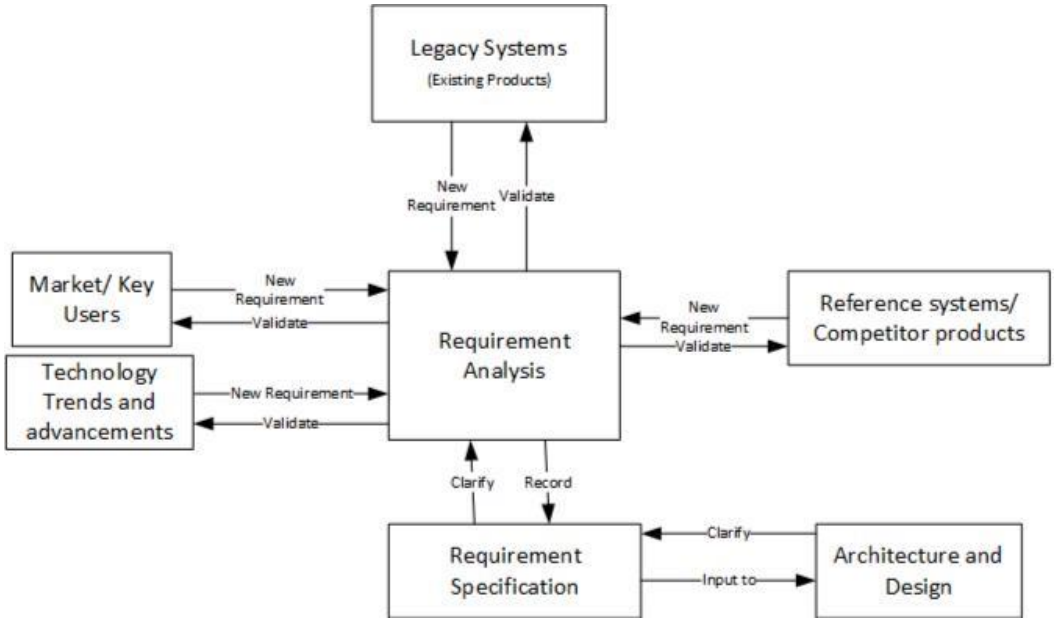
A. مهندسی مورد نیاز

برای خط تولید مورد نیاز ما، مهندسی چالش برانگیز است و نیاز برنامه های مختلف تجزیه و تحلیل و نیاز پلت فرم نیز باید مرتفع شود. از سوی دیگر نیازهای مورد نیاز روشن و معتبر هستند و نیازمند یک پیش نیاز برای طراحی نرم افزار منظم و توسعه و بهبود اشتباهات در مراحل اولیه انتشار از طریق روند توسعه و حل آنها مشکلات بعدی هستند. معاملات فرآیند استخراج مورد نیاز با ابهام، غیررسمی بودن، ناقص و دارای تناقض بودن روبه رو می شوند، که در آن "دانش" از الزامات این مسئله است که روشن و واضح نیست [9]. نیازها می توانند به عنوان الزامات عملکردی (FRS) و نیازهای غیر کارکردی (NFRs)، به عنوان مثال به صورت قابلیت اطمینان، نگهداری، و عملکرد طبقه بندی شوند. این الزامات ورودی کلیدی برای طراحی معماری نرم افزار [10] می باشد. ما فرآیندهای مهندسی مورد نیاز استاندارد را که شامل (1) استخراج مورد نیاز ، (2) تجزیه و تحلیل مورد نیاز، (3) مشخصات مورد نیاز و (4) اعتبار مورد نیاز است و آن ها را شامل می شود را دنبال می کنیم.

1) استخراج مورد نیاز: ما تمام ورودی های ممکن جمع آوری شده از بازار / کاربران کلیدی و سهامداران را به صورت یکجا در می آوریم. از آنجا که خط تولید ما در حال جایگزین شدن با مجموعه ای از سیستم های قدیمی است، این ذینفعان شامل کاربران کلیدی و مدیران محصول این سیستم ها هستند. ماهیت جهانی این محصول نیز باعث می شود تا موارد مورد نیاز برای این محصولات از بخش های مختلف بازار به دست بیاید.

2) تجزیه و تحلیل مورد نیاز: در فاز تجزیه و تحلیل ما نیازمند سند تنها نخواهیم بود، بلکه استفاده از روش بازرسی چند بعدی در اینکار مورد نیاز است (شکل را ببینید 2). ما موارد مورد نیاز برای سیستم های رقیب را مطالعه می کنیم و پیشرفت های تکنولوژیکی، نیازهای بازار، امکان سنجی و هزینه تحقق آن ها را برآورده می کنیم. بر این اساس در تحلیل اولیه ما هم با سهامداران بررسی می کنیم که کدام محصولات به احتمال زیاد در نسخه بعدی مورد توجه خواهند بود و در نتیجه نیاز به بررسی بیشتر و تصمیم گیری دقیق تری دارند. این داوطلبان از طریق یک فرایند جداسازی که در آن ما مشترک هستیم و تنوع مطالعه در آن وجود دارد و دسته بندی مورد نیاز بر اساس اینکه آیا آنها برای ادامه تولید مفید هستند یا خیر بررسی می شود، همچنین این مورد که این محصولات وارد مقیاس پلتفرم بشوند یا اینکه دارای یک عملکرد منحصر به فرد و یا یک محصول خاص هستند. نیازها با تاثیر پلت فرم ممکن است بیشتر به یک نیاز خاص و یک نیاز پلت فرم های فنی مشتق شده تقسیم شده باشد. همه نیاز پلت فرم ها با توجه به اهداف کسب و کار دسته بندی و طبقه بندی می شوند. نیازهای نرم افزار برای تجزیه و تحلیل بیشتر به یک تیم نرم افزار جداگانه اختصاص داده می شود. نیازهای غیر کارکردی با استفاده از یک رویکرد مبتنی بر سناریو مشتق می شود.

3) مشخصات مورد نیاز: تمام الزاماتی که از طریق یک فرایند فیلتر چند لایه تجزیه و تحلیل شده اند، در همکاری با ذینفعان بیشتر مورد تجزیه و تحلیل قرار می گیرند و در حوزه انتشار پلت فرم فعلی مستندسازی می شوند.



شکل 2. بازرسی چند بعدی

4) اعتبار سنجی نیاز: در مرحله اعتبار سنجی ، ما بررسی ویژگی هایی مانند کامل بودن، ثبات، امکان سنجی و آزمون پذیری را انجام می دهیم.

B. تجزیه و تحلیل نوع:

می تواند گزینه های طراحی های متعدد برای یک راه حل وجود داشته باشد که نیاز خاصی در یک خط محصول را بر آورده خواهد کرد. تجزیه و تحلیل نوع روندی است که پس از رسیدن به مناسب ترین گزینه طراحی است که با جایگزین کردن و با جمع آوری کردن، مستند سازی و ارزیابی آنها همراه است. سنجش می تواند مفهومی انجام شود، اما اغلب نیاز به توسعه نمونه اولیه برای ارزیابی امکان سنجی، مزایا، معایب و جایگزین طرح های مختلف نیز وجود دارد. اسناد و مدارک روشن از نتایج حاصل از تجزیه و تحلیل نوع در موارد زیر برای ما مفید بوده است:

• چند جایگزین طراحی معتبر می تواند وجود داشته باشد و یکی نیز می تواند بین گزینه های مختلف وجود داشته باشد، جنبه های عملکردی آن ها مربوط به یک تصمیم از این گزینه ها ممکن است به تعویق افتاد. در اینجا یک نیاز فردی، و جنبه های غیر عملکردی مربوط به اسناد و مدارک تجزیه و تحلیل می شود و از تکرار کار طراحی در طول مراحل پروژه بعدی نیز جلوگیری خواهد شد.

• گزینه های طراحی در یک خط محصول گزینه های طراحی پلت فرم را تحت تاثیر قرار می دهد و بر روی برنامه های کاربردی خط تولید نیز تاثیر می گذارد. گزینه طراحی می تواند محدودیت اجرایی را تحت تاثیر را قرار بدهد و یا می تواند به عنوان یک طرح برای طراحی محصول در خدمت قرار گرفته شود. معماری برای خط تولید ممکن است مجموعه ای از گزینه های مختلف برای برخورد با مشترکات و تنوع محصولات در نظر گرفته شود. گرفتن این گزینه ها و منطق برای هر گزینه تیم را قادر می سازد تا گزینه های مختلفی برای تولید محصول در چک لیست داشته باشد. انواع مزایا و محدودیت های باید در زمینه معماری موجود در بخش های مختلف ارزیابی شود. در حالی که انجام تجزیه و تحلیل نوع برای یک نیاز جدید برای خط تولید ما یک نوع انتخاب مناسب و چالش برانگیز است. در سناریوی

توسعه نرم افزار جهانی ذینفعان متعددی برای رسیدن به یک تصمیم نقش دارند. استفاده از معیارهای زیر از فرآیند انتخاب را حمایت می کند:

• ویژگی های کیفیتی از معماری و رنک خود آن ها. ما با استفاده از مجموعه ای از ویژگی های کیفیتی، در یک درخت ابزار خواهیم بود. برای هر ویژگی یا کیفیت ما مجموعه ای از سناریوهای مختلف را در دسترس داریم که زمانی که در مراحل مختلف مورد استفاده قرار می گیرد.

• سهولت ادغام با معماری موجود و اجزای آن. یک معماری که شامل تعداد زیادی از الگوهای طراحی های مختلف است، به راحتی می تواند تناقض تبدیل شدن به یک درک در طول زمان را طراحی کند که به طور یکپارچه ادغام به معماری موجود برای جلوگیری از پیچیدگی اجتناب ناپذیر خواهد شد.

• درصد استفاده مجدد در برابر ساخت هم روی بازار و نگهداری تاثیر می گذارد. استفاده مجدد با تطبیق اجزای موجود ممکن است پیچیدگی طراحی جزء را افزایش دهد اما همان زمان باعث کاهش پیچیدگی معماری کلی با حذف طرح های متعدد مرتبط می شود. از طرف دیگر در نسخه برداری ممکن است قابل قبول باشد اگر پیچیدگی استفاده مجدد بیش از حد بالا نباشد.

• سازگاری فن آوری آینده و استفاده آسان از آنها. در حالی که برخی از فن آوری مزایای روشنی را ارائه می دهند و استفاده از آنها در معماری سودمند است، گنجاندن بی رویه فن آوری های جدید، ممکن است در طول زمان، منجر به یک معماری بیش از حد پیچیده باشد. ما فن آوری هایی را در سطح سازمان پیگیری می کنیم و استفاده خود را با یک نقشه راه تکنولوژی کنترل می کنند.

• تأثیر بر برنامه های کاربردی ساخته شده در بالای خط تولید. سازگاری میان گزینه های طراحی و سادگی و در نتیجه یک عامل تعیین کننده در ساخت یک پلت فرم خط تولید برای نرم افزار و توسعه محصول قابل استفاده است.

در موارد خاص ممکن است تمام نکات بالا را برآورده سازد اما معرفی برخی از محدودیت ها مورد نیاز است. در این صورت ممکن است ما یک نیاز برای همکاری با ذینفعان کلیدی می باشد. تجزیه و تحلیل نیاز قابل توجه تعادل بین

گزینه های مختلف است، جنبه های عملکردی خود را مربوط به یک نیاز فردی، و جنبه های غیر عملکردی مربوط به معماری کلی در نظر گرفته می شود.

ما یک مورد برای توسعه یک ویژگی جدید با دو نوع طراحی مختلف را در نظر گرفته ایم:

-یکی از ویژگی های جدید که از ابتدا و توسعه می یابد و باعث ایجاد یک مولفه دیگر توسعه یافته نیز می شود.

-ویژگی های جدید توسعه یافته توسط استفاده مجدد و اقتباس از قطعات موجود

توسعه ابتدایی انعطاف پذیری باعث در نظر انطباق ویژگی های مورد نیاز می شود، اما یک مورد بالاتر به بازار و هزینه های نگهداری آینده بالاتری را خواهد داشت. پیچیدگی تطبیق یک جزء موجود کمتر از پیچیدگی مدیریت دو راه حل مستقل بود.

استفاده مجدد از اجزای موجود محدودیت در قابلیت استفاده موجود است اما صرفه جویی قابل توجه و زمان سریع به بازار است. در این مرحله ما به گذشته و در مورد الزامات نگاه کردیم و نقاطی را که به دلیل استفاده برآورده نشود استفاده مجدد شد و تجزیه و تحلیل برای شناسایی جایگزین های ممکن برای رسیدن به اهداف و نیاز جدید در نظر گرفته شده برای تحقق بخشیدن به اهداف موجود در نظر گرفته شده است. این منجر به شناسایی محدودیت ها و ایده های جدید برای دستیابی به اهداف شد. الزامات مورد نیاز در شرایط سهامداران کلیدی اصلاح شد.

C. تصمیمات کلیدی معماری و مبادلات

ویژگی های کیفیت سیستم های نرم افزاری بزرگ تا حد زیادی توسط سیستم معماری تحت تاثیر قرار می گیرد. این ویژگی های کیفیت با اغلب بر روی یکدیگر تاثیر می گذارندو و نمی تواند در انزوا مورد تجزیه و تحلیل قرار بگیرد. به عنوان مثال، اصلاح و انعطاف پذیری بر عملکرد و امنیت تحت تاثیر قرار می گیرد و قابلیت استفاده و عملکرد و پیچیدگی تحت تاثیر قرار خواهد گرفت و همچنین بر روی هزینه تاثیر می گذارد.

اگر ما فقط بر روی یک ویژگی بهینه سازی نماییم، ما ممکن است اهمیت ویژگی های دیگر را نادیده بگیریم. حتی اگر بخواهیم که توجه بیشتری کنیم، اگر ما را برای ویژگی های متعدد تجزیه و تحلیل بیشتری را در نظر بگیریم، ما هیچ راهی برای درک مبادلات ساخته شده در مکان های مختلف برای بهبود یک ویژگی در نظر نگرفته ایم. [12]

معماری با شرایط جدید تکامل می یابد. قبل از گرفتن تصمیم طراحی، ما زیر جنبه های کلیدی را نیز در نظر میگیریم:

• درک مشترک از اهداف کسب و کار، نیاز، و تصمیم گیری های طراحی در میان ذینفعان کلیدی توسعه پیدا می کند. اگر منطق پشت گزینه های طراحی و تصمیم گیری برای ذینفعان کلیدی وجود داشته باشد و توضیح داده شود، آنها ممکن است تمایل بیشتری به سازش در عملکرد و تنظیم نیازها داشته باشند.

• محدودیتها، تصمیمات و معاوضه کلیدی معماری موجود. تصمیمات گذشته برای قضاوت در مورد اینکه آیا یک تصمیم برای طراحی جدید سازگار است یا خیر، یا منجر به تناقض و فرسایش طراحی می شود نیز مهم است.

• مقایسه معماری موجود در برابر صفات کلیدی یک نیاز جدید را در نظر می گیرد. در زمان تجزیه و تحلیل، جنبه های عملکردی اغلب مقدم بر جنبه غیر کاربردی است. این به طور معمول به راه حل غیر رضایتبخش با یک ضربه ی احتمالی از هزینه بالا برای تغییر در مراحل بعد منجر می شود.

• لیست نقاط حساسیت معماری. دانش در مورد نقاط حساس از معماری مورد نیاز است و برای قضاوت در مورد خطرات ناشی از یک گزینه طراحی و یک عامل تعیین کننده نیز مهم می باشد. اینکه آیا نمونه سازی اولیه برای اعتبارسنجی یک تصمیم حیاتی است یا خیر.

• طوفان فکری موقتی در روش های مختلف، منطق و تاثیر آن ها. ذینفعان از جمله مدیران محصول، مهندسان و یا معماران اغلب بر این باورند که تنها "یک گزینه طراحی صحیح" وجود دارد. طوفان مغزی کمک می کند تا فضای راه حل ها باز شود.

ما مراحل ذکر شده در بالا را در یکی از موارد زیر برای نیاز های جدید که شرایط سخت تری از نظر زمان برای استفاده مجدد دارد را در نظر می گیریم و استفاده مجدد نمی تواند منجر به راضی کردن شرایط ویژه بدون تغییر را داشته باشد. ما اولویت بندی مختلف را با محدودیت های زمان شناسایی کردیم و همراه با تجزیه و تحلیل مبادلات از ویژگی

های با کیفیت برای هر گردش کار را شناسایی نمودیم و راه حل های مختلف را برای کمک به این موضوع در نظر گرفتیم. در حالی که هرگز شکی قابلیت های جدید برای ارائه و انعطاف پذیری بیشتر در جزء برای نگهداری وجود ندارد، نمونه سازی لازم است تا اینکه آیا شرایط تاخیر دقیق می تواند از طریق طراحی مجدد یک جزء موجود برآورده شود یا خیر.

D. طرح استفاده مجدد از اجزاء نرم افزار

این مهم است که از یک رویکرد ساختار یافته در هنگام گرفتن تصمیمات استفاده مجدد استفاده شود. ما با چالش های اضافی ناشی از این واقعیت مواجه می شویم که تیم های طراحی در سایت های مختلف قرار دارند، در تصمیم گیری با طراحی تاثیر جهانی نیاز به هماهنگ شدن دارد. نکات برجسته زیر رویکرد ما در ارزیابی نامزدها برای استفاده مجدد هستند:

• جزئ مخزن - ما یک مخزن جهانی از اجزای نرم افزار را نگهداری می کنیم. این مخزن به طور مداوم در روز آپدیت می شود و جنبه های کلیدی مانند مسئولیت، ویژگی های غیر کاربردی، انتساب به لایه ها و رابط ها به روز می شوند.

• مبادلات معماری و طراحی - تصمیمات گرفته شده به عنوان بخشی از تجزیه و تحلیل نوع جمله معاوضه می شوند و سپس در مخزن سند می شوند.

• بدهی فنی و محدودیت ها - محدودیت طراحی و بدهی فنی مستند شده است. از یک طرف این خدمت ورودی به عنوان کلید برای یک تیم به حساب می آید که آیا یک مولفه را می توان مورد استفاده مجدد قرار داد یا خیر، از سوی دیگر سطح استفاده مجدد می تواند باعث ایجاد اولویت سازی برای پرداختن به محدودیت ها و بدهی های فنی می شود.

• طراحی و هدف کیفیت محصولات - اهداف کیفیتی را برای یک جزء بر اساس اهمیت آن در نظر می گیرد، به عنوان مثال سطح مورد انتظار در استفاده مجدد و یا تاثیر شکست ها.

4. چالش ها

ما چهار رکن را هنگامی اضافه کردیم که ما به مجبور به اضافه کردن قابلیت های قابل توجه به پلت فرم خط تولید و چالش های مواجه شده زیر بودیم:

خصوصیات مورد نیاز ناقص بود. مشخصات به طور عمده توسط راه حل های در محصولات موجود به وجود آمده بودند. جنبه های با کیفیت ویژه مانند عملکرد و قابلیت استفاده که در آن تحت پوشش نباشد. وجوه مشترک و تنوع بین دامنه های مختلف توسط پلت فرم خط تولید ما که به اندازه کافی شناخته شده نیستند، پشتیبانی شده است.

مدیریت محصولات مسئول خصوصیات مورد نیاز قرار گرفته بودند و پرسنل اخیر را تغییر داد و در یک مکان مختلف جغرافیایی نیز واقع شده است.

ما این کمبودها را توسط بازرسی محصولات از تمام حوزه های مربوطه آدرس دهی کردیم. ما به طور خاص برای بازخورد مشتری راه حل های مختلفی را در نظر گرفتیم و در نتیجه نیازهای جدیدی را نیز در نظر گرفتیم. ما در جلسات منظم با مدیران محصول سازمان به بحث در مورد دانش اضافی که ما در طول تجزیه و تحلیل به دست آورده بودیم، پرداختیم. در طول این جلسات به طور منظم ما اغلب تا به حال پرسش هایی را مطرح نموده ایم که چرا در مقابل آنچه به عنوان نسخه های اولیه به وجود آمده است، در آن بر روی یک راه حل واحد متمرکز نشده ایم و تنها بر روی یک مورد تمرکز شده است.

در طول تجزیه و تحلیل نوع ما یک جزء به عنوان نامزد استفاده مجدد را مورد بررسی قرار داده ایم و دلیل آن است که اکثر قابلیت های مورد نیاز ارائه شده شده اند و صرفه جویی قابل توجهی در توسعه شده است و در نتیجه زمان ارائه به بازار سریعتر از زمان وعده داده شده است. چالش برای ارزیابی اینکه آیا جزء موجود می تواند متناسب باشد یا باعث برآورده کردن نیازهای عملکرد بالاتر بدون ارائه پیچیدگی غیر قابل مدیریت باشد یا خیر.

ما با توسعه نمونه های اولیه آغاز کردیم، برای ارزیابی اینکه آیا الزامات می تواند تاثیری بر روی تغییرات بر اساس کدهای موجود داشته باشد یا خیر. در طول این فعالیت ما متوجه شدیم ما می توانیم تمام نیازهایی که مورد استفاده مجدد از یک جزء موجود قرار می گیرد را برآورده نماییم. ما تا به حال به معرفی محدودیت های اضافی برای نیاز کرده

ایم. به منظور متقاعد کردن مدیریت محصول به قبول یک راه حل که برای برطرف کردن نیاز موجود لازم می باشد، انحراف از قابلیت های ارائه شده در محصولات ما مجبور شدیم تا به وضوح اقرار کنیم به مبادلاتی که در این مورد انجام دادیم. قابلیت های کامل در مقابل یک راه حل ساده تر به طور قابل توجهی باعث کاهش هزینه های تعمیر و نگهداری نرم افزار و عملکرد بهتر شده است. علاوه بر این ما می توانیم نشان دهیم که استفاده مجدد نیز به یک محصول با سازگاری بیشتر خواهد شد و به طور کلی با قابلیت استفاده بهتر در هم آمیخته است.

5. درس های یاد گرفته شده

• مهندسی مورد نیاز یک فرد محور فعالیت با تکرار بالا است. که با دانش بیشتر تکامل می یابد.

• نیازها تمام ابعاد را ارزیابی و فیلتر کردند و در توافق با ذینفعان کلیدی باعث افزودن ارزش به خط تولید شدند.

• معماری که باز به تغییر و تکامل برای هر محصول دارد برای همه مفید است. معماری ساده، قابل فهم و سازگار ذاتا برای تغییر باز است. انعطاف پذیری و پیچیدگی باید محدود به مواردی باشد که در آن یک نیاز های ثابت برای تنوع وجود دارد.

• در هر مرحله در روند طراحی معماری و پیاده سازی باید اجازه داده شود تا در صورت نیاز به مراحل قبلی برای رفتن به الزامات مورد نیاز برای تجدید نظر راهی باز شود.

• مستندات گذشته و تصمیم گیری های طراحی فعلی نیز می تواند مفید باشد.

• بازخورد استفاده زودرس ثابت برای کشف مسائل با داشتن یک راه حل تعریف شده با سیستم قدیمی بیش از حد نیز می تواند در ذهن مفید باشد.

• تجزیه و تحلیل متغیر یک گام بسیار مهم است و ورودی برای تجزیه و تحلیل نوع آینده و جزء استفاده مجدد نیز باید مهم در نظر گرفته شود. نتایج برای درک و ارزیابی تصمیمات تاریخی هنگام ارزیابی شرایط جدید مورد نیاز است.

• تجزیه و تحلیل معماری نه تنها در حفظ تعادل صفاتی کیفیت در زمان معماری ایجاد شده مهم است، بلکه باعث افزایش کمک به ارزش برای شرایط جدید و ارزیابی احتمالات برای جزء استفاده مجدد خواهد شد.

• ارزیابی محتاطانه استفاده مجدد در برابر کیفیت مورد نیاز کسب و کار جدید باعث کمک برای ایجاد قابلیت استفاده مجدد نمونه و همچنین صرفه جویی در وقت و هزینه خواهد شد.

• کامپوننت استفاده مجدد دارای مزایای متعدد مانند زمان ارائه به بازار، ثبات، learnability (کاهش تلاش های آموزشی) و به خصوص نگهداری خواهند شد، اما باید مراقبت شود تا از جزء استفاده مجدد در موارد به طور گسترده ای و موارد که غیر کاربردی و مهم نیستند نیز جلوگیری شود.

REFERENCES

[1] L. Bass, P. Clements, and R. Kazman. Software Architecture in Practice, 3rd Edition. Addison- Wesley, 2012.

[2] Mourad Oussalah , Software architecture: Principles, techniques and tools.By Copyright c 2013 John Wiley & Sons, Inc, Chapter 2 SOFTWARE ARCHITECTURE:ARCHITECTURE CONSTRAINTS, p1

[3] R. Kazman. Software Architecture. In Handbook of Software Engineering and Knowledge Engineering, S-K Chang (ed.). World Scientific Publishing, 2001.

[4] P. C. Clements and L. M. Northrop. Software Architecture: An Executive Overview. CMU/SEI-96-TR-003, 1996

[5] Philippe Kruchten, Rafael Capilla, and Juan Carlos Duenas. The decision view's role in soft- ware architecture practice. IEEE Software, 26(2) :36–42, 2009.

[6] P. Clements, F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, and J. Stafford. Documenting Software Architectures, Views and Beyond, Second Edition. Addison-Wesley, 2010.

[7] Chouki Tibermacine, Christophe Dony, Salah Sadou, and Luc Fabresse , Architecture Constraints as Customizable, Reusable and Composable Entities

[8] Rick Kazman, Mark Klein, Mario Barbacci, Tom Longstaff, Howard Lipson, Jeromy Carriere , The Architecture Tradeoff Analysis Method

[9] Jaya Vijayan and G.Raju , A New approach to Requirements Elicitation Using Paper Prototype by

[10] Lin Liao ,From Requirements to Architecture: The State of the Art in Software Architecture Design

[11] Felix Bachmann and Len Bass ,Managing Variability in Software Architectures

[12] Rick Kazman, Mario Barbacci, Mark Klein, S. Jeromy Carrière , Experience with Performing Architecture Tradeoff Analysis

[13] http://www.referenceforbusiness.com/management/Or-Pr/Product-Life-Cycle-and-Industry-Life-Cycle.html