به نام خدا

نام دانشجویان:

محمد تقی نسب

9714355031

فاطمه الیاسی

9714355009

نام درس:

آزمون نرم افزار

موضوع تحقیق:

Non-Functional Testing

نام استاد:

خانم آزاده استاد زاده

چکیده:

الزامات غیرکارکردی کیفیت کلی و یا ویژگی های یک سیستم را تعریف کنیم. اگرچه مهم هستند ، اما اغلب به دلایل زیادی از جمله فشار زمان و بودجه مورد غفلت قرار می گیرند. در توسعه نرم افزار چابک ، تمرکز بر پیاده سازی ویژگی و تحویل ارزش به مشتری است و به همین ترتیب ، جنبه های غیرکارکردی یک سیستم نیز باید مورد توجه قرار گیرد. آزمایش الزامات غیرکارکردی به دلیل جنبه های متقابل کارکردی و عدم شفافیت نیازهای آنها توسط بازار در بیشتر بخشهای پروژه ها ، چالش برانگیز است ]1 و 2[ . هدف از این مقاله مروری بر دو تحقیق انجام شده در بررسی تجربی نحوه کار اعضای تیم چابک با آزمایش غیرکارکردی در پروژه های خود ، با هدف شناسایی عوامل اولیه موثر بر آزمایش الزامات غیرکارکردی ، به ویژه عملکرد و امنیت در توسعه چابک است. در تحقیق کاماچو و همکاران با بیست متخصص فناوری اطلاعات در یک شرکت بزرگ چند ملیتی مصاحبه شده است. در نتیجه هفت عامل اصلی موثر بر تست غیرکارکردی و چهار عمل اصلی را که برای غلبه بر چالش های پیش روی اتخاذ می کردند ، شناسایی شد. هدف این تحقیق این بوده است که تحقیقات را در مقیاس وسیع تری تکرار کنند. در همین حال ، این کار کمک های اولیه ای به متحصصان کرده و برای تحقیقات آینده الهامبخش است [2]

مقدمه:

امروزه ، در هر زمینه عملیاتی ، کامپیوتر یک چیز ضروری برای اجرای برنامه های مورد نظر است. توسعه برنامه های نرم افزاری در صنایع کنونی از جمله اتومبیل، مخابرات، خرده فروشی، دولت، بانکداری و غیره نقش مهمی دارد. توسعه نرم افزار شامل برخی مراحل برای پیروی از جمله جمع آوری الزامات، ایجاد برنامه، آزمایش و حفظ برنامه است. هر مرحله ذکر شده با هم پیوند خورده اند، در صورت مشاهده اشکال اساسی می تواند بازسازی زیادی ایجاد کند، بنابراین محصول نرم افزاری باید به درستی آزمایش شود. و با وجود رقابت گسترده بین شرکت های نرم افزاری، زمان تحویل محصول نقش اساسی دارد. همچنین فرایند تست نرم افزار یک فرآیند پرهزینه است و آزمایش کامل عملاً غیرممکن است. بسیاری از سازمانهای نرم افزاری حداکثر 40٪ از منابع خود را برای آزمایش هزینه می کنند.

امروزه در هر زمینه عملیاتی ، کامپیوتر یک چیز ضروری برای اجرای برنامه های مورد نظر است. توسعه برنامه های نرم افزاری در صنایع کنونی از جمله اتومبیل، مخابرات، خرده فروشی، دولت، بانکداری و غیره نقش مهمی دارد. توسعه نرم افزار شامل برخی مراحل برای پیروی از جمله جمع آوری الزامات، ایجاد برنامه، آزمایش و حفظ برنامه است. هر مرحله ذکر شده با هم پیوند خورده اند، در صورت مشاهده اشکال اساسی می تواند بازسازی زیادی ایجاد کند، بنابراین محصول نرم افزاری باید به درستی آزمایش شود. و با وجود رقابت گسترده بین شرکت های نرم افزاری، زمان تحویل محصول نقش اساسی دارد. همچنین فرایند تست نرم افزار یک فرآیند پرهزینه است و آزمایش کامل عملاً غیرممکن است. بسیاری از سازمانهای نرم افزاری حداکثر 40٪ از منابع خود را برای آزمایش هزینه می کنند.

در سال 2001 ، توسعه چابک با هدف دستیابی به روشی برای ارائه نرم افزارهای کاربردی و افزودن ارزش تجاری در مراحل اولیه توسعه نرم افزار بوجود آمده است. ارتباطات ، نرم افزارهای کاربردی ، کیفیت و تیم های چند منظوره همانطور که در بیانیه چابک 4 شرح داده شده است، ارزشهایی هستند که در توسعه چابک بسیار مورد توجه قرار می گیرند. تا آن زمان ، آزمایش نرم افزار وابستگی زیادی به الزامات داشت و متخصصان تستها این الزامات را با هدف کلی کیفیت سیستم به موارد آزمایشی 5 تبدیل می کردند. در شرایط چابک، این سناریو دیگر امکان پذیر نیست، فاز مشخصی از استخراج و تجزیه و تحلیل الزامات وجود ندارد که مجموعه ای از الزامات را تولید کند که به عنوان ورودی فرآیند آزمایش استفاده شود. در مورد توسعه آبشاری سنتی 6، پیچیدگی تعریف، توصیف و اولویت بندی الزامات غیرکارکردی مشاهده شد. معمولاً الزامات غیر کارکردی اولویت خود را در برابر الزامات عملکردی از دست می دهند.

در کنار آن، آزمایش الزامات غیرکارکردی جدی گرفته نشده است و به دلیل ویژگی های آن اغلب در رده ریسک پایین قرار می گیرد. آزمایش غیر کارکردی به زمان طولانی اجرا و یک رویکرد باز نیاز دارد. ضرورت یک رویکرد کلی و ضرورت زمان اجرای طولانی نیز می تواند به عنوان نگرانی دیگر ذکر شود، زیرا توسعه چابک تمرکز خود را بر پیاده سازی ویژگی و تحویل سریعتر ارزش به مشتری می گذارد (به طور کلی الزامات کارکردی) ، حتی دشواری بیشتر برای شناسایی جنبه های غیرکارکردی ایجاد میکند. نیازهای تست غیر کارکردی می تواند از منابع مختلف، از تجارت، به عنوان نیاز یک مشتری، قطعه فنی، بخشی از تغییرات معماری یا بخشی از رفتار تولید که در چابک به عنوان نمونه بخشی از ادغام DevOps خواهد بود، استخراج شود.

در تحقیق کاماچو و همکاران تمرکز بر شناسایی عوامل موثر بر فرآیند آزمایش الزامات غیرکارکردی (امنیت و اجرا) در عمل و چگونگی مدیریت این عوامل توسط اعضای تیم در کارهای روزانه شان است و سوالات زیر مطرح شد:

1. عوامل موثر در آزمایش الزامات غیرکارکردی در تیم های چابک چیست؟
2. تیم های چابک چگونه عوامل شناسایی شده را کنترل می کنند؟

برای تست عملکردی کامپایلرهای، آزمایش کنندگان نرم افزار به طور کلی استفاده برای اجرای مجموعه ای از سوئیت تست در نسخه های مختلف نرم افزار بهینه سازی شده و مقایسه نتیجه عملکردی است که می تواند یا عبور (رفتار درست) یا شکست (رفتار نادرست ، تصادفات، یا اشکالات). برای تست غیر کاربردی ، بهبود کد منبع برنامه ها از نظر عملکرد می توانند به چندین متفاوت اشاره کنند خواص غیر کاربردی کد تولید شده مانند اندازه کد، منابع و یا مصرف انرژی، زمان اجرا، در میان دیگران . تست خواص غیر کاربردی است بیشتر به چالش کشیدن به دلیل کامپایلرهای ممکن است تعداد زیادی از ترکیب بهینه سازی بالقوه، آن را سخت و زمان بر برای توسعه دهندگان نرم افزار برای پیدا کردن / ساخت توالی بهینه سازی هایی که کلید خاص کاربر را راضی می کند اهداف و معیارها. همچنین نیاز به درک جامع دارد از معماری سیستم زمینه ای، هدف برنامه، و بهینه سازی های موجود از کامپایلر. در برخی موارد، این بهینه سازی ها ممکن است به طور منفی کاهش یابد کیفیت نرم افزار و خراب کردن عملکرد نرم افزار در طول زمان .

تاریخچه:

الزامات غیر کارکردی، همچنین به عنوان الزامات فنی ، کیفیت خدمات ، ویژگی های کیفیت متمرکز بر جنبه هایی که به طور معمول چندین مورد از الزامات کارکردی را شامل می شوند شناخته می شود. اگرچه برای موفقیت در پروژه مهم و حیاطی تلقی می شوند، این معمول است که الزامات غیرکارکردی در مقایسه با الزامات کارکردی مورد توجه قرار نگیرند ]1 و 2[ . با این حال ، بسیاری از مشتریان ممکن است در بخش تجاری برنامه تمرکز کنند و لزوماً جنبه مهم بسیاری از الزامات غیرکارکردی را درک نمی کنند[2].

کریسپین و گرگوری (2009( استدلال می کنند که شرکای تجاری ممکن است تصور کنند که توجه تیم توسعه به الزامات غیرکارکردی مانند اجرا، امنیت محصول نهایی را به خطر

می اندازد. با اینکه همه پروژه ها نگران تست غیرکارکردی نیستند، الزامات غیرکارکردی باید برای اطمینان حداقل یک بار چک شوند که توسط تیم مورد توجه قرار میگیرند و از مشتری اهمیت آن برای بیزینسشان پرسیده شده باشد. این مطالعات نشان می دهد که زمان تفکر در مورد تستهای غیرکارکردی هنگام نشر یا برنامه ریزی سرعت است. با این حال ، شروع آزمایش بدون الزامات دشوار است. مسئله دیگر این است که تیمهای توسعه چابک معمولا از تعداد کمی از توسعه دهنده ها تشکیل می شوند که اغلب به عنوان آزمایش کننده نیز عمل میکنند. با این حال آزمایشات غیرکارکردی مانند عملکرد به ابزارهای ویژه ای مانند پروفایلر و ممکن است به دانش تخصصی نیازمند باشند. نیاز به دانش تخصصی منجر به نیاز به اعضای تیم با مهارتهای خاص برای موفقیت سیستم یا جلوگیری از مشکلات تولید می شود. ممکن است برنامه نویسان به این نکته آگاه نباشند که تست غیرکارکردی مانند عملکرد و امنیت ممکن است اولویت بالایی داشته باشد و کلید کیفیت باشد. علاوه بر این، به دلیل ماهیت درگیر کردن ویژگی های بسیار زیاد سیستم مورد نظر، تستهای غیر کارکردی نمی توانند به عنوان بخشی از آزمایش واحد اجرا شوند و معمولا زمانبر بوده و نمیتوانند در چرخه یک پارچگی پایدار نرمال اجرا شوند ]1 و 2[.

آزمایش به دلایل زیادی اتفاق می افتد. بسته به ویژگی و روند نرم افزاری که دنبال می شود می توان انواع مختلف آزمایش را برای دستیابی به اهداف مختلف انجام داد. این مسئله در توسعه نرم افزار چابک، جایی که توسعه حتی می تواند از دیدگاه آزمون شروع شود ، تفاوتی ندارد. هدف آزمایش امنیت، اعتبارسنجی سیستم از نظر آسیب پذیری های امنیتی مانند محافظت از داده ها ، محرمانه بودن ، یکپارچگی ، احراز هویت ، در دسترس بودن و موارد مجوز است. آزمایش امنیت وابستگی زیادی به الزامات امنیتی اعمال شده توسط سیستم تجاری به عنوان مثال یک برنامه مالی دارد. بسته به سیستم می توان چندین نوع آزمون امنیتی را اعمال کرد ، مانند: اسکن آسیب پذیری ، ارزیابی آسیب پذیری ، ارزیابی امنیتی و تست نفوذ [1].

روش تحقیق:

در مطالعه ارائه شده در تحقیق کاماچو و همکاران و تحقیق دیو 3 و همکاران، هدف بررسی عوامل موثر بر آزمایش الزامات غیرکارکردی در تیم های چابک و نحوه به کارگیری آنها توسط اعضای تیم بود. در تحقیق کاماچو و همکاران برای دستیابی به سوالات مطرح شده در این تحقیق با20 شرکت کننده از یک شرکت IT چند ملیتی محاسبه انجام شد و سپس داده ها با استفاده از تحلیل موضوعی مورد تجزیه و تحلیل قرار گرفتند .

نتایج:

در طول تجزیه و تحلیل داده ها در تحقیق کاماچو و همکاران، هفت عامل اصلی از مصاحبه با آزمایش کنندگان، توسعه دهندگان، معماران نرم افزار و صاحبان محصولات شناسایی شد. کدها از منابع اطلاعات در هفت موضوع اصلی سازماندهی شده اند: اولویت، فشار زمان، هزینه، مسائل فنی، آگاهی، فرهنگ و تجربه. در تحقیق دیو و همکاران نیز عوامل تجربه ، فرهنگ و آگاهی به عنوان مهمترین عوامل تأثیرگذار بر شناسایی نیازهای آزمایش چابک ، اجرا و حل مسائل غیرکارکردی مورد تأکید قرار گرفتند.

بهینه سازی کامپایلر:

در گذشته محققان نشان داده اند که انتخاب بهینه سازی توالی ها ممکن است عملکرد نرم افزار را تحت تاثیر قرار . به عنوان یک نتیجه ، نرم افزار - بهینه سازی عملکرد می شود یک هدف کلیدی برای هر دو ، صنایع نرم افزار و توسعه دهندگان، که اغلب مایل به پرداخت هزینه های اضافی برای دیدار با اهداف عملکرد خاص، به خصوص برای منابع محدود سیستم. توالی های جهانی و از پیش تعریف شده، به عنوان مانند، O1 تا O3 در GCC، ممکن است همیشه نتایج عملکرد خوبی تولید نمی کند و ممکن است بسیار وابسته به معیار و کد منبع آنها آزمایش شده است. واقع هر یک از این بهینه سازی ها با کد تعامل و به نوبه خود، با تمام بهینه سازی های دیگر به روش های پیچیده.

راه اندازی تجربی:

1- برنامه های مورد استفاده در مطالعه تجربی: برای کشف تاثیر بهینه سازی کامپایلر مجموعه ای از برنامه های ورودی مورد نیاز هستند. برای این کار از یک ژنراتور برنامه C تصادفی استفاده می کنیم به نام Csmith . Csmith ابزاری است که می تواند تصادفی تولید کند برنامه های C که به صورت ایستا و پویا با استاندارد C99 . به طور گسترده ای برای انجام عملکردی مورد استفاده قرار گرفته است تست کامپایلرهای [1], [3], [22] اما نه مورد برای چک کردن الزامات غیر کاربردی. Csmith می تواند برنامه های C تولید که استفاده از طیف بسیار گسترده تری از ویژگی های C از جمله پیچیده جریان کنترل و ساختارهای داده مانند اشاره گرها، آرایه ها،و structs. برنامه های Csmith با سوئیت تست خود را می آیند که ساختار برنامه های تولید شده را کاوش می کنند. نویسنده استدلال می کنند که Csmith یک ابزار موثر پیدا کردن اشکال است چرا که آن را تولید تست است که کشف ترکیب غیر قابل استفاده از C ویژگی های زبان. آنها همچنین استدلال می کنند که برنامه های بزرگتر موثرتر هستند برای تست عملکردی.

2- تنظیم پارامترها: یک جنبه مهم برای متاهوریستی الگوریتم های جستجو نهفته در تنظیم پارامترها و انتخاب، که لازم است برای اطمینان از نه تنها مقایسه عادلانه، بلکه برای تکرار بالقوه. اطلاع رسانی پیاده سازی سه الگوریتم جستجوی تک هدفه (جستجوی تصادفی ). هر جمعیت اولیه / راه حل های مختلف الگوریتم ها کاملاً تصادفی هستند. معیار توقف این است هنگامی که حداکثر تعداد ارزیابی تناسب اندام رسیده است. تنظیمات پارامتر مشابه به همه الگوریتم های تحت مقایسه اعمال می شود. که نگرانی اصلی ما در این کار است. در طول فرایند تکاملی، هر راه حل با استفاده از متریک تازگی ارزیابی می شود. متریک تازگی محاسبه شده برای هر راه حل با در نظر گرفتن میانگین آن 15 نزدیکترین توالی های بهینه سازی از نظر شباهت (با توجه به همه دنباله ها در جمعیت فعلی و در آرشیو). در ابتدا آرشیو خالی است. فاصله تازگی عادی می شود در محدوده [0-100]. سپس برای ایجاد جمعیت های بعدی، یک نخبه از 10 موجودات رمان بدون تغییر کپی می شود، پس از آن که بقیه جمعیت جدید توسط تورنمنت ایجاد می شوند (انتخاب با توجه به تازگی). در واقع این آستانه پویا است. هر ارزیابی، بررسی می کنیم که چند نفر بوده اند که در آرشیو کپی شداند. اگر این عدد زیر ۳ باشد آستانه با ضرب آن در ۰٫۹۵ افزایش می یابد، در حالی که اگر راه حل ها اضافه شده به بایگانی بالاتر از 3، آستانه کاهش می یابد توسط ضرب آن در 1.05. علاوه بر این ، به عنوان اندازه آرشیو رشد می کند، نزدیک ترین محاسبه همسایه است که تعیین می کند نمرات تازگی برای افراد محاسباتی تر می شود.

از این رو، از یک ساختار داده اول در اول پیروی می کند که به معنی که وقتی یک راه حل جدید اضافه می شود، قدیمی ترین راه حل در آرشیو تازگی دور انداخته خواهد شد. بنابراین، ما فرد را تضمین می کنیم تنوع با از بین بردن توالی های قدیمی که ممکن است دیگرقابل دسترس از جمعیت فعلی نباشد. پارامترهای الگوریتم به صورت انفرادی در مقدماتی آزمایش کوک شدند. هر پارامتر، مجموعه ای از مقادیر آزمایش بود. مقادیر پارامتر انتخاب شده هستند که بیشتر در استفاده literature می شود. ارزشی که بیشترین عملکرد را به دست آورد امتیاز انتخاب خواهد شد.

بحث:

الزامات غیرکارکردی مفید مشخصات عمومی یا ویژگی های یک ابزار را به طور خلاصه بیان می کند. اگرچه مهم هستند ، اما ممکن است به طور منظم به دلیل انگیزه های زیادی کنار گذاشته شوند، که شامل زمان و هزینه های مالی می شود. در بهبود برنامه نرم افزار چابک ، یک نقطه عطف در مورد عملکرد و انتقال ارزش به خریدار وجود دارد و به همین ترتیب ، عناصر غیرکارکردی مفید یک سیستم نیز باید مورد توجه باشند.

امتحان کردن الزامات غیرکارکردی به دلیل داشتن عناصر مفید و عدم شفافیت خواسته های آنها با کمک بنگاه های اقتصادی در حداکثر بخشی از کارها ، چالش برانگیز هستند. شرکت کنندگان در تحقیق دیو و همکاران همچنین اظهار داشتند که ارزیابی کد به عنوان ابزاری است که امکان شناسایی و اجرای بررسی غیرکارکردی را فراهم می کند [1].

در تحقیق کاماچو و همکاران به صورت زیر به پاسخ سوالات دست پیدا کردند.

1. عوامل موثر در آزمایش الزامات غیرکارکردی در تیم های چابک چیست؟

هنگام تجزیه و تحلیل نتایج به دست آمده، مشهود بود که فاکتورهایی که در این تحقیق مورد بحث قرار گرفتند مانند زمان و هزینه موثر هستند اما جنبه هایی مانند تجربه نیز میتوانند بیشتر مورد رسیدگی قرار گیرند. تجربه به میزان زیادی توسط پاسخ دهندگان به دلیل ظرفیتشان در شناسایی بهتر نیازهای آزمایش غیرکارکردی براساس تجربیات گذشته شان به عنوان فاکتور مهم موثر بر آزمایش غیر کارکردی یاد می شد. به علاوه مشهود بود که اعضای ارشد بر روی تیم از لحاظ توجه به آزمایش غیرکارکردی تاثیر می گذارند. آگاهی و فرهنگ اهمیت آزمایش غیر کارکردی توسط اعضای باتجربه بسیار ذکر شده است و همچنین می تواند تاثیر دیگر فاکتورهای ذکر شده در نتایج مانند فشار زمان ، به عنوان مثال ، به دلیل توانایی مذاکره آنها به حداقل برسد. هنگامی که فرهنگ ایجاد شود و اعضای تیم از اهمیت آزمایش غیرکارکردی آگاه شوند و کیفیت بالایی را در ذهن داشته باشند، می توان عواملی را که به عنوان نقطه ضعف شناخته می شوند ، بهبود بخشید. اولویت بندی نیز به طور عمده توسط هر پاسخ دهنده ذکر شد. در چابک، اولویت براساس ارزش افزوده مشتری تعیین

می شود. هنگام تجزیه و تحلیل داستان کاربر، تیم ها بررسی می کنند که با توجه به پروژه یا ویژگی در حال اجرا ، چه چیزی به کسب و کار ارزش افزوده می دهد ، در صورتی که آزمایش غیرکارکردی در آن لحظه ارزش افزوده داشته باشد ، در طول توسعه اولویت خواهد یافت. جنبه های فنی نیز به عنوان عوامل موثر بر آزمایش غیرکارکردی ذکر شد. موارد فنی مانند استفاده زیاد از منابع می توانند اولویت موارد را تحت تأثیر قرار دهد. مشاهده اصلی این بود که عوامل با هم مرتبط هستند و بر یکدیگر تأثیر می گذارند. اولویت می تواند تحت تأثیر تجربه تیمی و فرهنگ آزمایش غیرکارکردی و آگاهی از اهمیت آن باشد. فشار زمان و هزینه را می توان براساس اعضای تیم باتجربه بهتر تحلیل کرد. با توجه به نتایج ، وقتی اهمیت آزمایش غیر کارکردی برای همه روشن نیست، به دلیل عدم اولویت بندی، زمان و بودجه اضافی اجرای آزمایش غیرکارکردی به مشکل می خورد، و همچنین به دلیل عدم شفافیت اهمیت آن از نظر کسب و کار و توسعه دهندگان، مسائل به وجود آمده که به برند مشتری یا بهره وری کاربر آسیب می رساند به اوج خود می رسد [2].

1. تیم های چابک چگونه عوامل شناسایی شده را کنترل می کنند؟

چهار عمل اصلی توسط شرکت کنندگان برای کنترل بهتر به همراه فاکتورهای موثر به دست آمده ذکر شد که در جدول 1 نشان داده شده است [2].

جدول 1 - عملکرد ها

|  |  |
| --- | --- |
| فاکتور | عمل |
| اولویت بندی | بحث کردن بر روی جنبه های غیر کارکردی در هنگام شروع پروژه ، برنامه ریزی سرعت و توسعه داستان کاربر |
| فرهنگ  اگاهی | برای داشتن اعضای تیم با نقشهای مختلف (توسعه دهندگان ، آزمایش کنندگان ، معماران نرم افزار و دارندگان محصولات) که الزامات غیر کارکردی و نیازهای آزمایش را بررسی می کنند. |
| اولویت  فرهنگ  آگاهی  فشار زمانی  هزینه  مسائل فنی | شفاف کردن ارتباط بین اعضای تیم و مشتریان از جمله تیمهای پشتیبانی تولیدDevOps) ) |
| اولویت  فرهنگ  آگاهی | کار با یک ذهینیت با کیفیت . آگاه کردن تیم از اهمیت الزامات غیرکارکردی و آزمایش |

در طول مصاحبه ها ، شرکت کنندگان لحظات اساسی را ذکر کردند که در آن باید جنبه های غیرکارکردی مورد بحث و تفصیل قرار گیرند.

شرکت کنندگان آغاز پروژه را به عنوان زمانی که ویژگی تجاری باید بررسی شود ، یاد کردند. به گفته آنها ، این تجزیه و تحلیل به شناسایی نیاز آزمایش غیرکارکردی برای تعریف اولویت کمک می کند. همین عمل در طول برنامه ریزی سرعت انجام می شود. اعضای تیم ارشد پس از مرور داستان کاربر برای توسعه، نیاز به بررسی عمیق تر در جنبه های غیرکارکردی مربوط به داستان کاربر را تقویت می کنند. شرکت کنندگان گزارش دادند که یک روش کار خوب با یک ذهنیت "نگران" است. هنگام داشتن ذهنیت ریسکی و توضیح در مورد چگونگی تأثیرگذاری بر تجارت ، سایر عوامل می توانند هزینه و زمان را به حداقل برسانند، زیرا نیازهای غیرکارکردی را بهتر می توان درک کرد. شرکت کنندگان اهمیت حضور داشتن نمایندگان همه نقش ها در طول مراسم چابک ، از جمله توسعه دهندگان ، معماران ، صاحبان محصولات ، مشاغل و در صورت امکان ، متخصصان غیرکارکردی به ویژه در برنامه ریزی آغازین و برنامه ریزی سرعت، گزارش کردند. آنها همچنین از اهمیت حضور اعضای ارشد در این جلسات خبر دادند. به گفته شرکت کنندگان ، مکالمه در مورد الزامات غیرکارکردی و آزمایش باید در طول توسعه داستان کاربر و از طریق بحث در مراسم تیم چابک مانند جلسات standup ادامه یابد. ارتباطات به طور گسترده ای به عنوان یک کانال برای کنترل با عوامل موثر بر آزمایش غیرکارکردی ذکر شده است. شرکت کنندگان بر این باورند که برقراری ارتباط خوب پیرامون مسائل غیر کارکردی با داشتن ویژگی های تیم های همسو با ارتباطات DevOps یا آگاهی داشتن توسعه دهندگان از رفتار برنامه در تولید ، می تواند در شناسایی نیازهای تست غیرکارکردی کمک کند. آنها معتقدند که گاهی اوقات، توسعه دهندگان و آزمایش کنندگان اطلاعات کافی برای تهیه یک استراتژی آزمایش خوب ندارند. از نظر آنها ، ارتباط با سایر تیم ها و کاربران واقعی می تواند منجر به شناسایی زمان تمرکز بر روی اولویت بندی، شود. همچنین ، کار بر روی درک کیفیت از نظر شرکت کنندگان بسیار مهم است. شرکت کنندگان همچنین از روش مرور کد به عنوان ابزاری که در شناسایی و اجرای آزمایش غیرکارکردی کمک می کند ، نام بردند [2].

نتیجه گیری:

تجربه ، فرهنگ و آگاهی به عنوان مهمترین عوامل تأثیرگذار بر شناسایی نیازهای آزمایش چابک ، اجرا و حل مسائل غیرکارکردی مورد تأکید قرار گرفتند. موارد فنی همچنین به عنوان عواملی که بسیار هم راستا با هزینه و آگاهی در مورد جنبه های غیرکارکردی و رفتار سیستم در تولید هستند ، به عنوان مثال زمان پاسخگویی بالا ، گزارش شدند. با توجه به اولویت ، فرهنگ و آگاهی ، راه حل های ممکن تجزیه و تحلیل جنبه های غیر کارکردی در جلسات پروژه ، مانند شروع پروژه و برنامه ریزی سرعت و انعکاس غیرکارکردی در طول توسعه داستان کاربر است. مشارکت نقشهای مختلف در جلسات و همچنین تجربه اعضای تیم می تواند در به حداقل رساندن مسائل کمک کند. هزینه و فشار زمان می توانند با فرهنگ اولویت و آگاهی مرتبط باشند ، زیرا اعضای تیم دیدگاه روشنی درباره اهمیت جنبه های غیرکارکردی دارند ، بنابراین با مکالمه خوب با مشتری و اولویت بندی کارها ، عوامل باید به حداقل برسند. مسائل فنی میتواند با مشارکت تیم با توجه به تجزیه و تحلیل بهتر توسط تجربه اعضای تیم در راستای ارتباط با تیم های دیگر ، مانند یکپارچه سازی بهتر DevOps ، حل شود. البته در تحقیق ماچو و همکاران تعداد افراد کم و محدود بود. اکثر پاسخ دهندگان مهندس اجرایی بودند و از هیچ آزمایشگر امنیتی مصاحبه انجام نشد. بنابراین ، نتایج گزارش شده در این تحقیق می تواند با آنچه تیم های دیگر شرکت ممکن است بیان کنند به دلیل تفاوت در زمینه و پس زمینه فرهنگی، متفاوت باشد.

منابع:

[1] Dave, Ms Shivani, Mr Hitesh Patel, and Mr Ashvin Prajapati. "Non-functional testing in agile development." (2018).

[2] Camacho, Cristina Rosa, Sabrina Marczak, and Daniela S. Cruzes. "Agile team members perceptions on non-functional testing: influencing factors from an empirical study." 2016 11th international conference on availability, reliability and security (ARES). IEEE, 2016.

[3] A Framework for Non-functional Testing of Compilers