دوره هفدهم، شماره بهار و تابستان ۱۴۰۳

مجله فناوری اطلاعات در طراحی مهندسی

Information Technology in Engineering Design

https://sanad.iau.ir/journal/ited

چالش‌های مهندسی نیازمندی‌ها در تولید نرم‌افزارهای شهر هوشمند با استفاده از روش‌های چابک: مرور ادبیات سیستماتیک

هوبخت عطاران[(۱)] اسماعیل خیرخواه*[(۲)] حسن شاکری[(۳)]

(۱) گروه مهندسی کامپیوتر، واحد مشهد، دانشگاه آزاد اسلامی، مشهد، ایران

(۲) گروه مهندسی کامپیوتر، واحد مشهد، دانشگاه آزاد اسلامی، مشهد، ایران*

(۳) گروه مهندسی کامپیوتر، واحد مشهد، دانشگاه آزاد اسلامی، مشهد، ایران

(تاریخ دریافت :۱۴۰۲/۱۱/۱۶ تاریخ پذیرش: ۱۴۰۳/۰۴/۰۹)

چکیده

فرآیندهای مهندسی نیازمندی‌های نرم‌افزارهای شهر هوشمند، طراحان نرم‌افزار را قادر می‌سازد تا تغییرات در نیازهای نرم‌افزاری مشتریان را مشخص کنند. با این حال چالش‌هایی در ارتباط با فرآیندهای مهندسی نیازمندی‌های شهر هوشمند وجود دارد که مانع از توسعه سریع و پایدار نرم‌افزارها می‌شوند. تحقیقاتی که به چالش‌ها با راه‌حل‌های موجود رسیدگی می‌کنند؛ جزیره‌ای، متنوع و فراگیر هستند. در این مطالعه ما از یک مرور ادبیات سیستماتیک همراه با طبقه‌بندی موضوعی و تجزیه و تحلیل شکاف‌ها برای بررسی راه‌حل‌های موجود در برابر چالش‌ها استفاده می‌کنیم؛ تا نشان دهیم روش‌های موجود به‌طور کامل کارا نبوده و خلا موجود را شناسایی نموده و پیشنهادات مناسب جهت بازتحقیق‌های آتی برای پر کردن این خلا ارائه گردیده است. گونه‌شناسی طبقه‌بندی چالش‌های پیش روی توسعه نرم‌افزارهای شهر هوشمند و به‌طور خاص در مهندسی نیازمندی‌ها و نحوه برخورد راه‌حل‌ها با چالش‌ها در این مطالعه آمده است. مطالعه ما از سال ۲۰۰۹ تا ۲۰۲۳ را پوشش می‌دهد. Scopus و Web of Science به عنوان بزرگ‌ترین پایگاه داده برای انتشارات دانشگاهی معتبر جست‌وجو شد. با استفاده از معیارهای خروج برای فیلتر کردن مقالات در مجموع ۸۰ مقاله معتبر انتخاب و بررسی شدند. پس از بررسی چالش‌ها و راه‌حل‌ها، کمبودها و نواقص موجود بررسی و چارچوبی پیشنهاد می‌شود که مهندسی نیازمندی‌های شهر هوشمند با یک رویکرد هماهنگ را فراهم می‌سازد. این مطالعه به مرز نظری رویکردهای مهندسی نیازمندی‌های نرم‌افزارهای شهر هوشمتد و استخراج آن‌ها برای عملیاتی شدن، کمک می‌کند.

کلمات کلیدی: مهندسی نیازمندی‌ها، توسعه نرم‌افزار، شهر هوشمند، روش‌شناسی‌های[1] چابک

*عهده‌دار مکاتبات:

اسماعیل خیرخواه

نشانی: گروه مهندسی کامپیوتر، واحد مشهد، دانشگاه آزاد اسلامی، مشهد، ایران

پست الکترونیکی:**e.kheirkhah@gmail.com**

[1] Methodology

۱- مقدمه

مقالات علمی متعددی وجود دارند که رویکردهای توصیه‌ای، در نرم‌افزارهای کاربردی شهر هوشمند را پوشش می‌دهند و همچنین راه‌حل‌های توصیه‌ای در طراحی‌های نرم‌افزاری شهر هوشمند در دنیای واقعی پیاده‌سازی شده‌اند. تعاریف متعدد و زیادی از شهر هوشمند در مقالات متعدد مورد بررسی وجود دارد که رایج‌ترین این تعاریف که در آن تعدادی از ویژگی‌های شهر هوشمند را گردآوری کرده و یک طبقه‌بندی جدید از ابعاد شهر هوشمند، یعنی اقتصاد هوشمند، محیط هوشمند، تحرک هوشمند، حکمرانی هوشمند، زندگی هوشمند و مردم هوشمند را ارائه داده است، در مقاله [1] می‌توان مشاهده کرد. نرم‌افزارهای فعلی موجود در زمینه شهر هوشمند، با توجه به نبود یک تعریف دقیق همراه با جزئیات کامل و همچنین عدم شناخت کافی و تفاوت‌های مشهود و غیرقابل‌انکار در اولویت‌ها و فرهنگ‌ها و نیازهای جوامع مختلف، حالت جزیره‌ای به خود گرفته و از نبود یکپارچگی لازم، رنج می‌برند. در چنین شرایطی، با توجه به این نکته که مهندسی نیازمندی‌های نرم‌افزاری، حلقه مفقوده تمامی روش‌شناسی‌های تولید نرم‌افزار هستند [2]، کاربران به اجبار به سمت روش‌های چابک برای تولید نرم‌افزارهای مورد نیاز رفته‌اند.

روش‌شناسی چابک[1] از زمان معرفی آن در حدود ۲۰ سال پیش، به‌طور گسترده‌ای برای توسعه نرم‌افزار پیاده‌سازی شده است [3]. رویکردهای چابک به سازمان‌ها کمک می‌کنند تا با سرعت [4] و انعطاف‌پذیری برای انطباق سریع و واکنش به موقعیت‌های متغیر [5] عمل کنند. بسیاری از انواع روش‌های چابک وجود دارد به عنوان مثال اسکرام[2] یکی از پرکاربردترین رویکردها در صنعت است [6]، زیرا می‌تواند چرخه‌های توسعه نرم‌افزار کوتاه مدت همراه با نیازمندی‌های استخراج شده، معمولاً ۲ تا ۴ هفته [7] تحت عنوان "اسپرینت[3]" داشته باشد و نیازهای کاربر را تقریبا به موقع شناسایی کند [9]. بر اساس آمارهای موجود، احتمال موفقیت پروژه‌های چابک در مقایسه با پروژه‌های سنتی دو برابر بیشتر است [9]. با این حال گزارش‌های متناقضی وجود دارد که نشان می‌دهد حدود ۵۰ درصد پروژه‌های چابک به دلیل مشکلات بودجه در طول برآورد زمان یا با دامنه کوچک‌تر از آنچه در ابتدا مشخص شده بود به چالش کشیده شده‌اند که اکثرا ناشی از برآورد نادرست یا ناکافی نیازمندی‌ها بوده است [9].

استفاده از روش‌های چابک بدون چالش نیست و چالش‌ها در طول فرآیند توسعه نرم‌افزار متفاوت است. به عنوان مثال مقاله [10] اظهار می‌دارد که در روش‌شناسی چابک، فاز جداگانه آزمایشی مناسبی وجود ندارد و صرف زمان برای آزمایش جامع دشوار است. بنابراین نگه داشتن اسناد تست، داده‌های تست، موارد تست و سناریوها در طول روش چابک،

[1] Agile
[2] Scrum
[3] Sprint

چالشی است که باید روی آن کار کرد. روش چابک یک روش محبوب است که برای مهندسی نیازمندی‌ها[1] استفاده می‌شود. فرآیند حیاتی برای جمع‌آوری نیازمندی‌های توسعه نرم افزار چابک[2] [11,12] به دلیل ذی‌نفعان متعدد و پویایی های متغیر شهر هوشمند، با چالش‌هایی مواجه است. این فرآیند شامل استخراج نیازمندی‌های، تجزیه و تحلیل، مشخص کردن و اعتبار سنجی است که مستلزم تعامل با ذی‌نفعان، اولویت‌بندی نیازمندی‌ها و مدیریت نیازمندی‌های متضاد است [13] و مشخصاتی از جمله نیازمندی‌های عملکردی[3] و غیرعملکردی[4] را تولید می‌کند. از آن‌جایی که نیازهای مشتری به‌طور مداوم در حال تغییر است، تیم‌های چابک به جای تمرکز بر مستندات [10] که می‌تواند طراحی و آزمایش عملکردی و غیرعملکردی را تضعیف کند، بیش‌تر بر ارائه محصول متمرکز هستند. برخی از متداول‌ترین چالش‌های مهندسی نیازمندی چابک، مربوط به نیازمندی‌های غیرعملکردی است [14]. مهندسی نیازمندی‌های ضعیف به عنوان دلیل اصلی شکست حدود ۵۰ درصد پروژه‌ها، شناخته شده است [15].

چالش‌های فوق به رویکرد چابک اختصاص دارند. تحقیقات همچنین چالش‌هایی را نشان می‌دهد که ریشه در محیط‌های سازمانی گسترده دارند. به عنوان مثال، در مقاله [16] اشاره شده که بسیاری از چالش‌ها در کاربرد توسعه چابک در مقیاس بزرگ مربوط به تست نیازمندی‌های عملکردی با نرم‌افزار و به شکل ویژه به هم‌ترازی بین این دو فاز است. تراز به هماهنگی اعمال، مصنوعات[5] و نقش‌ها[6] اشاره دارد. در همین راستا Luong و همکاران [3] نشان دادند که اضطراب، انگیزه اعتماد متقابل و شایستگی ارتباط به‌طور قابل‌توجهی با چالش‌های مربوط به انسان در تیم‌های چابک مرتبط است. راه‌حل‌های این چالش‌ها مستلزم تعامل با ذی‌نفعان، اولویت‌بندی نیازمندی‌ها و مدیریت نیازمندی‌های متضاد، استفاده از مهارت‌های مذاکره، همکاری و ارتباط است [13].

برخی از ادبیات سیستماتیک تا سال ۲۰۲۰ در مورد چالش‌های سیستم‌های چابک علاقه وافری به توصیف چالش‌ها دارند؛ اما عدم تمرکز بر تعیین مشکلات به منظور یافتن راه‌حل به‌طور خاص مشخص نیست که آیا چالش‌های کلیدی ریشه در روش‌شناسی چابک دارند یا در مسائل روش‌شناختی غیرچابک[7] مانند زمینه کسب‌وکار، ماهیت پروژه، تناسب هم‌سویی بین روش چابک انتخابی و زمینه خاص پروژه و همچنین اجرا و مدیریت پروژه‌های چابک. در یک مطالعه

[1] Requirement engineering(RE)
[2] Agile for RE (ARE)
[3] Functional requirements(FR)
[4] Non-functional requirements(NFR)
[5] Artifacts
[6] Roles
[7] Non Agile

اولیه [17] نشان داده شده که چالش‌های چابک در مقایسه با راه‌حل‌ها، نامتناسب و به‌طور قابل‌توجهی ناهماهنگ هستند و این دو سؤال اصلی تحقیق را مطرح می‌کند:

RQ1: طبقه‌بندی چالش‌های پیش روی توسعه نرم‌افزار چابک به‌طور کلی و به‌طور خاص در مهندسی نیازمندی‌ها خصوصا برای پروژه‌های بزرگ مانند شهر هوشمند چیست؟

RQ2: راه‌حل‌ها چگونه چالش‌ها را برطرف می‌کنند؟

پاسخ به سوالات می‌تواند به بحث ادبیات در مورد چالش‌های توسعه نرم‌افزار چابک با بینش برگرفته از آخرین ادبیات ۲۰۲۳-۲۰۲۰ با تمرکز مجدد بر روی یافتن راه‌حل‌های مناسب برای مقابله با چالش‌ها کمک کند.

برای دستیابی به هدفی که در سؤالات تحقیق آمده است، باید چند شرط مرزی را در محدوده این مقاله قرار دهیم. اولا با توجه به ادبیات گسترده و متنوع در مورد توسعه نرم‌افزار چابک ما تحقیق را به مهندسی نیازمندی‌های چابک محدود می‌کنیم. همان‌طور که قبلا ذکر شد، مهندسی نیازمندی‌ها حیاتی‌ترین فرآیند در چرخه توسعه نرم‌افزار است. ثانیا چالش‌های مهندسی نیازمندی‌ها و هم راه‌حل‌های کشف شده از بررسی سیستماتیک خودمان را شامل دوره‌ای از ۲۰۰۹ تا ۲۰۲۳ می‌کنیم.

ساختار بقیه مقاله به شرح زیر است: بخش ۲ روش تحقیق را با توضیح استراتژی تحقیق و فرایندهای استخراج و تجزیه و تحلیل داده‌ها مورد بحث قرار می‌دهد. بخش ۳ نتایج حاصل از تحقیق را ارائه می‌کند. بخش ۴ ضمن بحث‌های تکمیلی، طبقه بندی چارچوب چالش‌ها به همراه راه‌حل‌های کلیدی را ارائه داده و ضمن ارائه پیشنهادهای مناسب جهت بازتحقیق‌های آتی، مطالعه را به پایان می‌رساند.

۲- روش‌شناسی

این مطالعه با رویکردی دو مرحله‌ای به سؤالات تحقیق خواهد پرداخت. ابتدا مرور ادبیات سیستماتیک[1] برای شناسایی و طبقه‌بندی تمام چالش‌ها و دوم راه‌حل‌های مربوط به مهندسی نیازمندی‌های چابک با استفاده از دستورالعمل‌های Webster and Watson [16] انجام شد.

۲-۱- روش استخراج مرور ادبیات

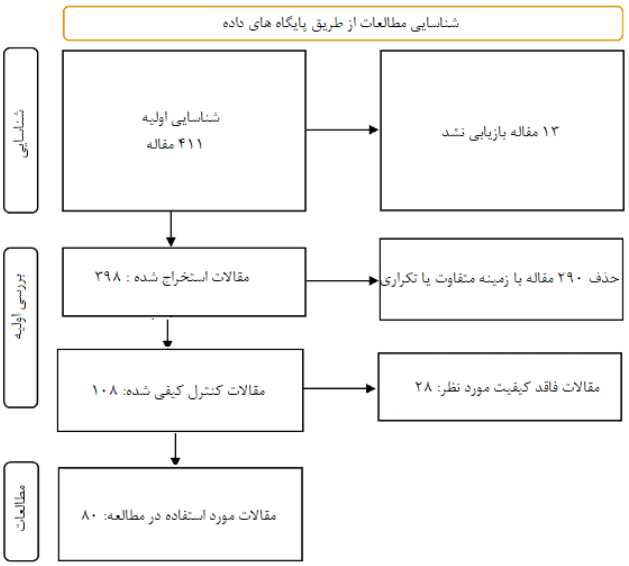
[1] Systematic literature review(SLR)

به دنبال رویکرد مرور ادبیات سیستماتیک پیشنهاد شده توسط Webster and Watson [16] جست‌وجوی گسترده‌تر برای مطالعات منتشر شده در دو دهه گذشته برای به‌دست آوردن دیدگاهی جامع از روش چابک، چالش‌ها و راه‌حل‌های مهندسی نیازمندی‌ها، مطالعات با معیارهای زیر انتخاب شدند:

- مقالات مجلات و کنفرانس‌ها بررسی شده بین ۲۰۰۹ و ۲۰۲۳.
- مطالعاتی که چالش‌های مهندسی نیازمندی‌های چابک را مورد بحث قرار نداده‌اند و مطالعاتی که تنها روش‌شناسی چابک را بدون بحث بیشتر به عنوان مثال ذکر کردند، در مطالعه اصلی حذف شدند.
- Scopus و Web of Science به عنوان بزرگ‌ترین پایگاه داده‌های معتبر برای دانشگاهیان، عمدتا برای این تحقیق استفاده شد. منابع متعدد دیگر نیز مورد بازبینی قرار گرفت و مقالات و موضوعات تکراری از این منابع متعدد حذف شدند.
- رشته‌های جست‌وجوی مورد استفاده برای استخراج مقالات عبارت بودند از:

KEY(“agile methodology” or “agile method” or “agile approach” or “agile development” or “agile practices” or “agile requirements engineering practices” or “agile way” or “agile software development”)) AND (TITLE-ABS-KEY(“requirements engineering” or “feature” or “task” or “requirement”)) AND (TITLE-ABS-KEY(“challenge” or “problem” or “issue” or “obstacle” or “limitation”)

شکل ۱ فرآیند انتخاب دنبال شده را نشان می‌دهد.



شکل ۱: فرآیند انتخاب مقالات

۲-۲- تجزیه و تحلیل داده‌ها

رویکرد مفهوم محور Webster and Watson's [18] برای جدول‌بندی چالش‌ها و راه‌حل‌ها برای تجزیه و تحلیل داده‌ها استفاده شد. راه‌حل‌های شناسایی‌شده برای چالش‌های مهندسی نیازمندی‌های چابک در مطالعات گنجانده شده است. به گفته Webster and Watson's [18]، یک رویکرد منطقی برای تجزیه و تحلیل سیستماتیک ادبیات، گروه بندی و ارائه آن‌هاست. چالش‌های گزارش‌شده به منظور شناسایی رایج‌ترین اصطلاحات استفاده شده است. چالش‌های مرتبط با رایج‌ترین اصطلاحات در ۱۱ زیرگروه سازماندهی شده و به همین ترتیب، راه‌حل‌های مرتبط ارائه شدند.

در مرحله بعد ۱۱ چالش و ۹ راه‌حل را با توجه به ارتباط موضوعی آن‌ها با اولا چالش‌های سازمانی کسب و کار، ثانیا چالش‌های مدیریت پروژه و ثالثا چالش‌های روش‌شناسی چابک به سه بعد طبقه‌بندی کرده که این طبقه‌بندی بر اساس مضامینی است که با ارجاع به تحقیقات قبلی که در آن یک یا چند مورد از این ابعاد عمدتاً ذکر شده، پدیدار گشته است [22–26] .

به عنوان یک مرور سیستماتیک، قادر به محاسبه فراوانی موضوعات پرداخته شده در هر مطالعه هستیم. اگرچه فراوانی لزوماً به معنای اهمیت نیست اما نشان‌دهنده اشتراک یا رواج موضوع در توسعه و کاربرد چابک و همچنین تمرکز توجه پژوهشی با وجود تنوع در پروژه‌های چابک با زمینه‌های سازمانی متفاوت است.

۳- نتایج حاصل از مرور ادبیات سیستماتیک

پس از بررسی‌های کامل مقالات و اولویت‌بندی آن‌ها و حذف موارد تکراری و یا اهمیت کمتر و همچنین مواردی که در این مطالعه مد نظر نبوده است، نتایج جالبی به‌دست آمد. ابتدا مهم‌ترین چالش‌های موجود به‌دست آمدند و سپس راه حل‌های این چالش‌ها نمایان گردیدند که در ادامه بیان خواهند شد. در ادامه یک طبقه‌بندی سه بعدی از چالش‌ها و راه‌حل‌ها به‌دست آمد که تا حدود زیادی می‌تواند پاسخگو باشد و در نهایت خلا اصلی مهندسی نیازمندی‌ها، نمایان گردید.

۳-۱- چالش‌های مهندسی نیازمندی‌های چابک

از ۸۰ مقاله بررسی شده ۱۱ چالش را شناسایی کردیم که در زیر به اختصار مورد بحث قرار می‌گیرند.

چالش۱(C1): نیازمندی‌های کیفیت

این نیازمندی‌ها معمولا نادیده گرفته شده است. تمرکز عمدتا بر روی نیازمندی‌های عملکردی [27] باقی مانده و نیازمندی‌های کیفی[1] در طول مراحل اولیه نادیده گرفته می‌شوند [28]. آزمایش یکپارچه‌سازی دیرهنگام برای تأیید نیازمندی‌ها می‌تواند نقص نیازمندی‌های کیفی را آشکار کند[29]. آزمایش دیرهنگام و نیازمندی‌های کیفی دیرهنگام می‌توانند موجب دوباره کاری شوند [29,30]. تغییرات در نیازمندی‌های عملکردی بر نیازمندی‌های کیفی تأثیر گذاشته [31] و نیازمندی‌های کیفی به‌ندرت در یک قطعه کد پیاده‌سازی می‌شوند [29]. نیازمندی‌های ذی‌نفعان مختلف ممکن است در تضاد [32] باشد که منجر به نیازمندی‌های کیفی ضعیف شود.

چالش۲ (C۲): حداقل مستندات

روش‌های چابک کمتر مستندگرا بوده و مستندسازی در این روش‌ها متمایل به صورت‌های موقت نادرست، ناقص یا عدم وجود هستند [32]. روش چابک بر اشتراک‌گذاری دانش ضمنی [8] متکی است که می‌تواند در طول زمان معنی خود را از دست داده [34,35] و از طریق جابه‌جایی پرسنل [23,36] مشکلات زمانی به وجود می‌آورند و ممکن است یک ارتباط قطع شود [25]. به عنوان مثال اعتبارسنجی داستان‌های کاربران دشوار بوده و نگهداری اسناد می‌تواند یک کار دلهره‌آور باشد [23].

چالش ۳ (C3): روش اولویت‌بندی نامناسب

مقیاس‌های ترتیبی و رتبه‌بندی برای اندازه‌گیری اولویت دشوار است. مشکلات مقیاس‌پذیری با تعداد زیادی از نیازمندی‌ها [37] وجود دارد که انتخاب نیازمندی‌های بهینه [32] و حفظ فهرست اولویت‌های نیازمندی را دشوار می‌سازد [38]. نیازمندی‌های عملکردی اغلب بر نیازمندی‌های غیرعملکردی اولویت دارند [39]. ارزش تجاری بر جنبه‌های دیگری که منجر به نادیده گرفتن نیازمندی‌های غیرعملکردی می‌شود، غلبه می‌کند. برای مثال در قابلیت نگهداری یا امنیت موارد گزارش‌شده وجود دارند که در آن مشتریان تمایلی به اولویت‌بندی یا ارائه دقیق نیازهای خود نداشتند [36].

چالش۴ (C4): مدیریت تغییر

[1] Quality requirements (QR)

نیازهای کسب‌وکار [22] که خود از نیازهایی با تغییرات سریع و اولویت‌بندی مجدد استمرار می‌یابد [40] تکامل یافته که باعث یک مسئله بی‌ثباتی می‌شود [41]. تأثیر و خطر تغییرات نیاز باید در نظر گرفته شوند [42]. تغییرات نیازمندی‌ها باعث می‌شود اسناد موجود به‌روز شوند [43] و می‌توانند بر هزینه و زمان‌بندی پروژه تأثیر بگذارند [44]. زمانی که تغییرات نیازمندی محدود به شروع یک اسپرینت نباشد؛ اختلال یا حتی لغو یک اسپرینت ممکن است اتفاق بیفتد [45].

چالش۵ (C۵): نیازمندی‌های نوشته شده ضعیف

نیازمندی‌های نوشته شده می‌توانند ناکافی [46]، بد نوشته شده [30]، با نقص کیفیت [23]، مبهم [47,48]، نامشخص، ناقص، ناسازگار و غیرقابل‌اجرا [40] و بیش از حد دانه‌ای یا بسیار سطح بالا باشند که باعث تفاوت می‌شوند. تفاسیر داستان‌های کاربر به جای نیازمندی‌های کیفیت، بر توصیف عملکرد تمرکز دارند [49].

چالش۶ (C۶): برآورد منابع نادرست

برآوردها را می‌توان به‌طور اساسی با تغییر نیاز در تکرارهای بعدی تغییر داد [50,22]. تخمین غیرقابل‌اعتماد به دلیل عدم وجود داده‌های تاریخی و کارشناسان [52,51,23] به ویژه برای داستان‌های کاربران سیستم‌های بزرگ [53] و برای تیم‌های چابک در مکان‌های جغرافیایی توزیع‌شده [54] اتفاق می‌افتد.

چالش۷ (C7): مشتری در دسترس نیست.

مشارکت و تعامل مشتری با تیم توسعهIT چالش برانگیز است [55,29,26]. این چالش همچنین به مشتریان فاقد اختیار برای تصمیم‌گیری و عدم انتقال نیازهای واقعی از مشتریان جایگزین [56] یک رابطه ضعیف [40] و در دسترس بودن کم مشتریان برای مذاکره شفاف‌سازی و بازخورد اشاره دارد [42,29]. زمانی که فاصله زیادی بین مشتری واقعی و تیم توسعه وجود داشته باشد، مالکیت نیازمندی‌ها مشکل است [57].

چالش ۸ (C۸): دانش مشتری

هنگامی که مشتریان در مورد روش چابک دانش کافی نداشته یا به رویکرد چابک اعتقاد ندارند، دریافت بازخورد با کیفیت از آن‌ها دشوار است [56]. این چالش همچنین شامل ناتوانی مشتریان [48,26]، فقدان صلاحیت فنی [56]، دانش ناقص حوزه [58,40]، دانش ناکافی از نیازمندی‌ها [23] و روش‌های توسعه نرم افزار چابک [49] می‌شود.

چالش۹ (C۹): معماری نامناسب

تصمیمات معماری اتخاذ شده در چرخه‌های اولیه می‌توانند زمانی که نیازمندی‌های جدید ایجاد می‌شوند، اضافی شوند [59,48,29]. این تصمیمات منجر به هزینه‌های اضافی برای بازسازی نرم‌افزار برای رفع نیازهای مشتری خواهد شد [48] .

چالش۱۰ (C۱۰): روش‌های ارتباطی

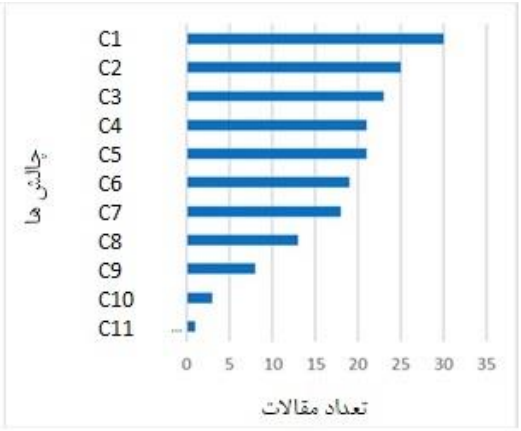
ممکن است برقراری ارتباط موثر با ذی‌نفعان دشوار باشد [42]. موردی گزارش شده است که در آن روش‌های ارتباطی بسیار پرهزینه بوده و به پیش‌زمینه فنی برای استفاده نیاز دارند و برای چابک طراحی نشده‌اند. ملاقات‌های حضوری بهترین روش ارتباطی و پیام‌رسانی فوری بدترین روش ارتباطی است [60].

چالش۱۱ (C۱۱): حفظ مشخصات مورد نیاز نرم‌افزار

برای یک تیم پرکار که روی چندین پروژه به‌طور همزمان کار می‌کنند، حفظ مشخصات نرم‌افزار دشوار است [61].

چالش‌های طبقه‌بندی نشده:

برخی از چالش‌ها مستقیما با مهندسی نیازمندی‌ها مرتبط نیستند. اما می‌توانند برای پروژه‌های چابک کلی باشند. از این رو در این دسته طبقه‌بندی‌نشده قرار می‌گیرند. به عنوان مثال محدودیت‌های قراردادی [57,26,25] به عنوان یک چالش ذکر شده، که قراردادهای قیمت ثابت می‌توانند از تغییرات جلوگیری کنند. همچنین پشتیبانی ناکافی توسط سرپرست پروژه، اجرای ویژگی‌ها بدون نیاز و به‌طور کلی زمان کافی نداشتن چالش‌های دیگری از این قبیل هستند [55]. شکل ۲ فراوانی چالش‌ها در مطالعات انجام شده را نمایش می‌دهد. جدول ۱ نشان دهنده منابع بررسی شده برای هر چالش است.



شکل۲: فراوانی چالش‌ها در مقالات

جدول۱: منابع بررسی شده برای هر چالش

چالش	منابع	تعداد
C1	8,12,14,19,22,23,25-31,39,40,42,48,55,56,65,66,78-84	30
C2	8,12,19,23,25,26,29,30,33-36,39,43,48,49,53,55-58,66,67,79,84	25
C3	12,22,23,25,26,29,32,36-39,41,42,48,49,56,61,62,64,76,79,85,86	23
C4	22,23,25,26,30,36,37,40-45,49,55,67,84-88	21
C5	22,23,29,30,36,39,40,42,46-49,56,57,61,73,76,80,89,90	21
C6	12,22,23,25,26,29,33,42,45,48,50-56,64,91	19
C7	12,22,23,25,26,29,30,36,40,42,49,55-57,61,67,76,79	18
C8	23,25,26,33,40,48,49,56-58,78,79,92	13
C9	23,25,29,36,48,56,59,78	8
C10	42,48,60	3
C11	61	1

۳-۲- راه‌حل‌های مهندسی نیازمندی‌های چابک

راه‌حل‌های گزارش شده مربوط به مهندسی نیازمندی‌ها در زیر خلاصه شده است.

راه‌حل ۱ (S۱): اطلاعات مورد نیاز را ارائه دهید.

این راه‌حل شـامل ارائه نیازمندی‌ها در قالب موارد آزمایشـی [15]، چارچوب ترکیب مجدد[1] برای مشـخص کردن نیازمندی‌ها به روشی منعطف برای پشتیبانی از تغییر نیازمندی‌ها [25]، داستان‌های هدف‌مدار کاربر و داستان‌های تحویل [49,26]، کارت‌های داستانی [14]، ایجاد نمودارهای مورد کاربرد [62,47] و معیارهای پذیرش مختصر در داستان‌های کاربر [29]. کیفیت داستان‌های کاربر را می‌توان از طریق یک ابزار یا با بررسی مجموعه‌ای از معیارها ارزیابی کرد [63]. در این ارزیابی یک کارشـناس حوزه با نقش تحلیلگر تجاری می‌تواند به بهبود کیفیت و صـحت نیازمندی‌ها کمک کند [64]. اسپرینت‌های کوتاه مدت با نیازهای کمتر، مشخصات نرم‌افزاری واضح‌تری دارند [61].

راه‌حل۲ (S۲): نیازمندی‌های کیفیت را مدیریت کنید.

آزمایش یکپارچه‌سازی به موقع کل سیستم، می‌تواند به شناسایی زودهنگام هرگونه نقص نیازمندی‌های کیفیت کمک کند. یک تیم پروژه می‌تواند یک بک لاگ نیازمندی‌های کیفیت جداگانه داشته باشد و یک نیازمندی کیفیت را در اولویت قرار دهد یا یک اسـپرینت کامل را می‌توان برای کار بر روی نیازمندی‌های کیفیت اختصـاص داد. برخی از پروژه‌های بزرگ و توزیع‌شده مالکیت نیازمندی‌های کیفیت را به تیم‌های متخصص می‌دهند که از اجرای آن‌ها اطمینان حاصل شود [29]. نیازمندی‌های کیفیت را با خطرات مرتبط خود اسـتخراج و مدیریت کنید [65]. از دسـتورالعمل‌های توسـعه‌یافته برای حمایت از نیازمندی‌های کیفیت پیروی کنید [66,39].

راه‌حل۳ (S۳): دانش را در مورد نیازمندی‌ها به اشتراک بگذارید.

در میان تیم‌های توسعه نرم‌افزار چابک، Borrego و همکاران [34] دریافتند که پیام‌های فوری و ایمیل‌ها روش‌های بهتری برای برقراری ارتباط هستند و روشی را برای استخراج دانش معماری از این پیام‌ها توسعه دادند. روش‌های دیگر عبارتند از: یک مرکز ارتباطی بین توسـعه‌دهندگان و مشتریان [60]، چون نمایندگان مشتری در محل می‌توانند تعاملات کاربر و توسـعه‌دهنده را بهبود بخشـند [67]. ارتباط مکرر از طریق تعاملات چهره به چهره جلسـات هفتگی و جلسات مصـاحبه [68,34,25] توصیه می‌شود. در صورت در دسترس نبودن مشتری از یک مشتری جانشین استفاده کنید [26]. برای به اشـتراک گذاشتن دانش و همگام‌سازی تلاش‌های توسعه، گروه‌های کاری مختلفی می‌توانند حول منافع مشترک ایجاد شوند [62].

راه‌حل۴ (S۴): مدیریت یک محصول عقب‌افتاده

[1] RE-COMBINE

هدف این راه حل پوشش دادن دیدگاه‌های مختلف مشتری، معماری نیازمندی‌های عملکردی و کیفی، با استفاده از پس‌افت‌های[1] متعدد است [62,29]. Gaikwad and Joeg [49] استفاده از مدل تحویل افزایشی چابک را برای مذاکره مجدد و تعیین درباره اولویت‌های نیازمندی، پیشنهاد می‌کنند. تمام تغییرات شناسایی‌شده در طول یک اسپرینت در حال اجرا در پس‌افت محصول منعکس‌شده و برای اسپرینت‌های آینده در نظر گرفته می‌شود [45].

راه‌حل۵ (S۵): فرآیند تخمین را بهبود بخشید.

نیازمندی‌ها در ابتدا ناپایدار هستند [26]. راه‌حل‌ها شامل استخراج منطق از هر برآورد برای کمک به تیم برای دستیابی به اجماع برای برآورد اختصاص زمان [45] و استفاده از تکنیک‌های داده‌محور برای برآورد تلاش [52] است.

راه‌حل۶ (S۶): قابلیت ردیابی نیازمندی‌ها را حفظ کنید.

Firdaus و همکاران [31] مدلی را برای ردیابی نیازهای کیفی، امنیتی و عملکردی پیشنهاد می‌کند. Urbieta و همکاران [69] رویکردی را با استفاده از واژگان توسعه‌یافته زبان برای سازماندهی داستان‌های کاربر و کمک به قابلیت ردیابی پیشنهاد می‌کنند. ابزارهای خوب شناخته‌شده‌ای مانند جیرا[2] می‌توانند مدیریت نیازمندی‌ها را تسهیل کنند [25].

راه‌حل۷ (S۷): حداقل اسناد مورد نیاز را شناسایی کنید.

یک نرم‌افزار تلفیقی مشخصات نیازمندی‌ها به جای مصنوعات مختلف توصیه می‌شود [61]. معماری اطلاعات پشتیبانی برای روش چابک مورد نیاز است. Gahyyur و همکاران [70] یک مدل زیرسیستم، فهرست اجزای قابل استفاده مجدد، ویژگی‌های مهم معماری و یک منطق مستندشده را توصیه می‌کنند.

راه‌حل۸ (S۸): روش اولویت‌بندی نیازمندی‌ها را بهبود بخشید.

سه مورد را به عنوان معیارهای اولویت‌بندی در نظر بگیرید: «رضایت ذی‌نفعان»، «ریسک اجرا» و «در دسترس بودن منابع» [32].

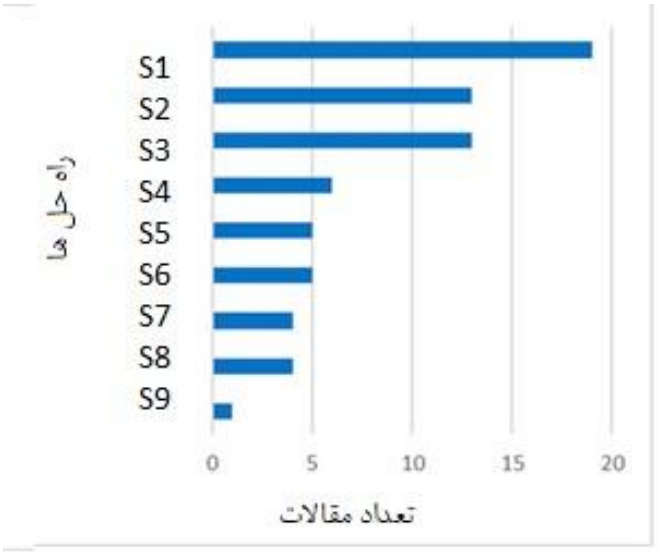
راه‌حل۹ (S۹): چشم‌انداز محصول را به اشتراک بگذارید.

[1] Backlog
[2] JIRA

بینش محصول مشترک بین مشتری و تیم توسعه حفظ کنید تا درخواست‌های تغییر را برای یک سیستم در حال اجرا کاهش دهید [45].

راه‌حل‌های طبقه‌بندی نشده:

Okesola و همکاران [26] اتخاذ تدابیر قانونی در عقد قراردادها، برای حمایت از ماهیت انعطاف‌پذیر مهندسی نیازمندی‌های چابک را توصیه می‌کنند. Inayat و همکاران [25] استفاده از پرداخت‌های ثابت در هر نسخه را پیشنهاد می‌کنند. این موارد در طبقه‌بندی گنجانده نشده‌اند زیرا مستقیما با مهندسی نیازمندی‌ها مرتبط نیستند. با توجه به فراوانی این راه‌حل‌ها در مقالات، رایج‌ترین (نه لزوما مهم‌ترین، بلکه رایج‌ترین) راه‌حل‌ها در شکل۳ نشان داده شده است. جدول ۲ نشانگر منابع بررسی شده برای هر راه‌حل است.



شکل ۳: راه‌حل‌های رایج

جدول۲: منابع بررسی شده برای هر راه حل

راه حل	منابع	تعداد
S1	15,19,25,26,29,30,44,47-49,55,61-63,73,76,85,93,94	19
S2	14,24-26,29,39,65,66,76,77,83,95,96	13
S3	24-26,30,34,41,49,60,66-68,97	13
S4	25,29,45,49,62,67	6
S5	12,26,45,52,91	5
S6	25,31,67,69,94	5
S7	30,48,61,70	4
S8	27,32,86,97	4
S9	45	1

۴- بحث و نتیجه‌گیری

چالش‌ها و راه‌حل‌های گزارش‌شده همان‌طور که در بخش‌های بالا خلاصه شده‌اند، معمولاً به صورت موضوعی هستند زیرا هر یک از زمینه خاصی که مطالعه در آن انجام شده است، استخراج شده و همه آن‌ها ریشه در کمبودهای روش‌شناسی چابک ندارند. برای پاسخ به سؤالات تحقیق ما بررسی بیشتر با رد شدن از زمینه خاص و تمرکز بر معانی موضوعی این چالش‌ها و راه‌حل‌ها انجام شد. آشکار می‌شود که بسیاری از چالش‌ها و راه‌حل‌ها علاوه بر روش‌شناسی چابک ریشه در محیط‌ها و فرهنگ‌های سازمانی، مشکلات تجاری و مسائل مدیریت پروژه و ابعاد پروژه دارند. از این رو ما از این سه بعد برای ترسیم تمام چالش‌ها و راه‌حل‌ها استفاده می‌کنیم. خاطرنشان می‌شود که برخی از چالش‌ها و راه‌حل‌ها در ابعاد چندگانه قرار می‌گیرند.

۴-۱- بعد محیط و فرهنگ سازمان

موفقیت رویکرد چابک به میزان تناسب آن با ساختار منابع فرآیندها و فرهنگ سازمان مربوط می‌شود. برخی از مسائل در این بعد بر تمرین چابک تأثیر منفی می‌گذارد از جمله تنظیمات سازمانی بزرگ، جایی که سطوح زیادی بین مشتری واقعی و تیم توسعه وجود دارد، نیاز به مالکیت و از دست دادن اطلاعات در هر سطح یک مشکل است. مشتریان

واقعی ممکن است به‌طور موثر با تیم چابک درگیر نباشند. نیازمندی‌های اشتراک دانش و بازخورد [57] در این مرحله به شدت نمایان است.

۴-۲- بعد مدیریت پروژه

بیشتر چالش‌های چابک شناسایی‌شده از این مطالعه به چالش‌های مدیریت پروژه (PM) مربوط می‌شود. همان‌طور که در قسمت‌های قبل مشاهده می‌شود، شش چالش در این بعد طبقه‌بندی می‌شوند که عبارتند از: C1 نیازمندی‌های کیفیت نادیده گرفته شده است، C3 روش اولویت بندی نامناسب، C4 مدیریت تغییر، C6 برآورد تلاش نادرست، C9 معماری نامناسب و C11 حفظ مشخصات موردنیاز نرم‌افزار. پروژه‌های چابک در مقیاس بزرگ مستلزم یک مدیر پروژه باتجربه است که مسئولیت کل تیم پروژه را بر عهده بگیرد و پاسخگوی موفقیت یا شکست پروژه باشد [71]. شش چالش کشف‌شده در مهندسی نیازمندی‌های چابک در واقع ریشه در پنج فرآیند مدیریت پروژه دارند که عبارتند از: مدیریت اجرای پروژه، توسعه طرح ساختار پروژه، توسعه برنامه پروژه، برآورد و تعریف هزینه‌ها بر اساس نیازها و توسعه و مدیریت تیم. یک تیم چابک تحت ساختار ماتریس مدیریت پروژه، معمولاً اعضای خود را از حوزه‌های عملکردی مختلف و در سطوح مختلف با ذی‌نفعان متعدد، متخصصان چابک خارجی، تحلیلگران و توسعه دهندگان دارد. همان‌طور که در این مطالعه مرور شد سوءتفاهم و تضاد در نیازمندی‌های هزینه‌ها، منافع و غیره می‌تواند علت اصلی شکست یک پروژه چابک باشد.

۴-۳- بعد روش‌شناسی چابک در پروژه‌های بزرگ

سه چالش به دلیل ارتباط مستقیم با روش چابک تحت این بعد گروه‌بندی می‌شوند. C۱ نیازمندی‌های کیفیت نادیده گرفته شده‌اند، C۲ اسناد حداقل و C۵ نیازمندی‌های نوشته‌شده ضعیف. روش چابک کمتر مستندگرا است و می‌تواند برای آزمایش مشکل ایجاد کند. نیازمندی‌های عملکردی به شدت به کاربر/مشتری وابسته است، در حالی که نیازمندی‌های غیرعملکردی را نمی‌توان به‌طور خاص در شروع پروژه توسط مشتریان مشخص کرد و از این رو در بسیاری از پروژه‌های چابک نادیده گرفته می‌شوند. واضح است که جا برای بهبود روش‌های چابک وجود دارد و این تنها راه حل برای مقابله با تمام چالش‌ها نیست.

۴-۴- ابعاد راه‌حل‌های روش‌شناسی چابک

با پیروی از همین رویکرد، تمامی راه حل‌ها، بررسی شده و در سه بعد طبقه‌بندی می‌شوند. جالب است بدانید که هفت راه حل شناسایی‌شده به احتمال زیاد در دسته بهبود روش‌شناسی چابک قرار می‌گیرند. پنج مورد از آن‌ها به مدیریت پروژه مرتبط هستند و دو مورد جهت به بهبود تنظیمات سازمانی به‌کار می‌روند. به نظر می‌رسد چند راه‌حل از مرزهای طبقه‌بندی عبور می‌کنند. به عنوان مثال، S3 در روش‌های اولویت‌بندی، S8 اشتراک دانش و S9 حفظ چشم‌انداز محصول مشترک، از این دسته هستند.

بعد سازمان/کسب‌وکار دو راه حل دارد: اشتراک دانش (S3) و حفظ چشم‌انداز محصول مشترک (S9). این‌ها برای موفقیت پروژه‌های چابک ضروری هستند. به اشتراک‌گذاری دانش مشتری در میان تیم پروژه چابک نشان دهنده هنجار و فرهنگ یک سازمان است که در رتبه سوم رایج‌ترین راه‌حل‌ها قرار گرفته است. همچنین مستلزم هماهنگی و ارتباط موثر بین تیم‌ها و بخش‌های توزیع‌شده است.

حفظ یک چشم‌انداز پروژه مشترک نیاز به یک رهبری رویایی دارد؛ تا اهداف را در سطوح و تیم ها به وضوح بیان کند که به مشخصات نیازمندی‌های صریح تبدیل می‌شود. بعد مدیریت پروژه شامل پنج راه‌حل است: مدیریت نیازمندی‌های کیفیت (S2)، بهبود فرایند تخمین (S5)، بهبود روش اولویت‌بندی نیازمندی‌ها (S8) و راه‌حل‌های فرامرزی S9 و S3. این راه‌حل‌ها به سمت مدیریت بهتر چابک پروژه هدایت می‌شوند. همان‌طور که قبلا ذکر شد اشتراک دانش و چشم‌انداز مشتری را می‌توان یک ویژگی مدیریت پروژه موثر در زمینه توسعه چابک در نظر گرفت.

نگاشت مشترک بین چالش‌ها و راه‌حل‌ها، نشان‌دهنده عدم تطابق ظاهری بین ابعاد چالش‌ها در مقابل راه‌حل‌ها است. با وجود این واقعیت که بسیاری از چالش‌ها در واقع ریشه در ابعاد سازمانی کسب‌وکار و مدیریت پروژه دارند، به عنوان مثال به نظر می‌رسد توجه بیشتری به راه‌حل‌های روش چابک داشته باشد تا ملاحظات تلاش‌های سازمانی. یافته‌های حاصل از این مطالعه به ما کمک کرد تا بینش‌های زیر را به‌دست آوریم:

- در ادبیات توسعه نرم‌افزار چابک، علاقه‌ی پژوهشی زیادی به چالش‌ها وجود دارد، اما به‌طور نامتناسبی راه‌حل‌ها در کاوش و بهره‌برداری کم و بعضا غیرقابل‌استفاده است.
- از راه‌حل‌های خلاصه‌شده توسط این مطالعه، بیش‌ترین بخش با هدف یافتن راه‌هایی است برای بهبود رویکردهای چابک و تا حدودی بهبود مدیریت پروژه. چالش‌هایی که ریشه در محیط و فرهنگ سازمانی و زمینه کسب‌وکار و ابعاد پروژه وجود دارند، نادیده گرفته شده‌اند.

- چالش‌های پیش روی پروژه‌های چابک(تیم‌ها) ریشه در سه بعد فوق‌الذکر دارد، ازاین‌رو، بهبود روش‌های چابک همه مشکلات را حل نمی‌کند و هر تلاشی برای بهبود عملکرد چابک باید سازمان/کسب‌وکار را در نظر بگیرد.
- محیط سازمانی و کسب‌وکار پیچیده و پویا هستند که چالش‌های زیادی ایجاد کرده و راه‌حل‌های متفاوتی را می‌طلبند. تناسب (یا تراز) بین این پیچیدگی‌ها و چابک باید قبل از شروع پروژه ارزیابی شود. مجدداً تاکید می‌شود که مهندسی نیازمندی‌ها کلیدی است؛ زیرا روش‌های چابک بر واکنش‌پذیری و ارتباطات غیررسمی تأکید دارند تضمین آن‌ها در میان تیم‌های چند رشته‌ای در یک سازمان بزرگ دشوار است.
- عملیات یک پروژه چابک تحت تأثیر چهار نیروی محرکه قرار می‌گیرد: (الف) تنظیمات سازمانی/تجاری حاکمیت، ترکیب اعضا، نقش‌ها و مسئولیت‌ها، منابع و ذی‌نفعان؛ (ب) صلاحیت مدیریت پروژه تجربه تعیینکننده اثربخشی شروع برنامه‌ریزی، کنترل، هماهنگی و ارتباطات است؛ (ج) نیازمندی‌های تجاری و عملکردهای موردانتظار نرم‌افزار (نیازمندی‌های عملکردی و غیرعملکردی)، (د) توانایی روش‌های چابک. از این رو یک رویکرد هماهنگ شده موردنیاز است.
- مهندسی نیازمندی‌ها کلید فرآیند چابک است. اکثر نیازمندی‌های عملکردی را می‌توان با مشتریان مناسب با دانش و در دسترس بودن شناسایی و به صراحت مشخص کرد. با این‌حال ما شک داریم که نیازمندی‌های غیرعملکردی را بتوان به‌طور دقیق در اسپرینت‌های اولیه مشخص کرد؛ زیرا این‌ها ویژگی‌های عملکرد سیستمی هستند که نمی‌توانند توسط مشتریان از قبل مشخص شوند تا زمانی که نحوه عملکرد سیستم‌ها را تجربه کنند.

۵- نتیجه‌گیری و پیشنهادها جهت بازتحقیق‌های آتی

در این مقاله با مطالعه مقالات متعددی که در زمینه مهندسی نیازمندی‌ها، با استفاده از روش‌های چابک در ظرف مدت ۱۴ سال ارائه شده بودند و توجه به این نکته که هنوز مشکلات بسیاری در زمینه مهندسی نیازمندی‌ها وجود دارد، چالش‌های موجود (۱۱ چالش) و راه‌حل‌های ارائه‌شده (۹ راه‌حل) در این مدت مورد بررسی قرار گرفته شد. مهم‌ترین پیامد این مطالعه این است که روش‌شناسی چابک همه چالش‌ها را ایجاد نمی‌کند و بنابراین نباید انتظار داشت که همه مشکلات را حل کند[98]. چالش‌هایی که ریشه در ابعاد دیگر دارند باید برای راه‌حل‌های جایگزین در

نظر گرفته شـوند[99]. مشـکلات کسـب‌وکار و ویژگی‌های تنظیمات سـازمانی، اهداف چندگانه، تیم‌های پراکنده، مالکیت و تصـمیم‌گیری مشکل توزیع‌شده، منابع مشتری و در دسترس بودن دانش و غیره به عنوان پیش‌شرط در نظر گرفته می‌شـوند. این بدان معناست که اگرچه روش چابک مزایای زیادی را ارائه می‌دهد، ممکن است با موقعیت‌های سازمانی پیچیده که نیازمندی‌هایش به‌طور قابل‌توجهی متفاوت است، مناسب نباشد. در سطوح مختلف و موقعیت‌های جغرافیایی مختلف ما از این به عنوان ارزیابی واجد شـرایط بودن روش چابک یاد می‌کنیم. با توجه به مشـخصـات نیازمندی‌های غیرعملکردی، بعید اسـت که قبل از اینکه کاربران و صـاحبان مشکل نتایج اولیه اسپرینت را در مراحل مختلف ببینند، به‌طور صـریح شـناخته شـوند. از این رو توسعه‌دهندگان چابک باید نیازمندی‌های غیرعملکردی را به روشـی قابل اثبات آغاز کنند[100] تا کاربران و صـاحبان مشـکل را برای تأیید و بهبود درگیر کنند. دلیل این امر این اسـت که توسـعه سریع فناوری‌های دیجیتال در امنیت مقیاس‌پذیری و معماری به این معنی است که انتظار نمی‌رود، کاربران تجاری پتانسـیل فناوری‌های پیشـرفته را در طول توسـعه چابک پیش‌بینی کنند [101]. چالش‌ها و مشکلات مهندسـی نیازمندی‌های چابک همه ریشـه در کاستی‌های روش‌شناسی چابک ندارند [102]. اگرچه جایی برای بهبود وجود دارد، روش‌های چابک تنها زمانی می‌توانند موثر باشـند که یک محیط سازمانی پشتیبان همراه با مدیریت پروژه مناسب وجود داشته باشد. محیط سازمانی زمینه کسب‌وکار نحوه شروع و اجرای پروژه‌های چابک را شکل می‌دهد و از این رو یک رویکرد هماهنگ برای رسـیدگی به چالش‌های سازمانی کسب‌وکار و پروژه مورد نیاز است. چارچوبی مبتنی بر آنتولوژی یا پردازش زبان طبیعی برای ایجاد ابعاد کلیدی و روابط متقابل پیشنهاد می‌شود [103] که توجه را فراتر از تمرکز فعلی بر روش‌شـناسی چابک سوق دهد. این چارچوب راه جدیدی برای تلاش‌های تحقیقاتی آینده در جست‌وجوی راه‌حل‌های مهندسی نیازمندی‌ها در پروژه‌های بزرگ باز می‌کند.

مراجع

[1] Houbakht Attaran, , Nahid Kheibari, and Davoud Bahrepour. "Toward integrated smart city: a new model for implementation and design challenges." *GeoJournal* (2022): 1-16.

[2] هوبخت عطاران، اسماعیل خیرخواه. "بررسی روش شناسی‌های تولید نرم‌افزار برای مهندسی نیازمندی‌ها در پروژه‌های بزرگ"، پانزدهمین کنفرانس بین‌المللی ساستک، موسسه آموزش عالی خاوران، ۱۴۰۱

[3] Luong, T.T.; Sivarajah, U.; Weerakkody, V. Do agile managed information systems projects fail due to a lack of emotional intelligence? Inf. Syst. Front. **2021**, 23, 415–433.

[4] Xu, Y.; Koivumäki, T. Digital business model effectuation: An agile approach. Comput. Hum. Behav. **2019**, 95, 307–314.

[5] Janssen, M.; van der Voort, H. Agile and adaptive governance in crisis response: Lessons from the COVID-19 pandemic. Int. J. Inf. Manag. **2020**, 55, 102180.

[6] Torrecilla-Salinas, C.J.; Sedeño, J.; Escalona, M.J.; Mejías, M. Estimating, planning and managing agile web development projects under a value-based perspective. Inf. Softw. Technol. **2015**, 61, 124–144.

[7] Manifesto for Agile Software Development. Available online: http://agilemanifesto.org/

[8] Alsaqaf, W.; Daneva, M.; Wieringa, R. Agile quality requirements engineering challenges: First results from a case study. In Proceedings of the IEEE International Symposium on Empirical Software Engineering and Measurement, Toronto, ON, Canada, 9–10 November 2017; pp. 454–459.

[9] Mersino, A. Agile Project Success Rates Are 2X Higher than Traditional Projects. Available online: https://vitalitychicago.com/blog/agile-projects-are-more-successful-traditional-projects/ (accessed on 14 April 2023).

[10] Rehman, A.U.; Nawaz, A.; Ali, M.T.; Abbas, M. A Comparative Study of Agile Methods, Testing Challenges, Solutions & Tool Support. In Proceedings of the 14th International Conference on Open Source Systems and Technologies (ICOSST), Lahore, Pakistan, 16–17 December 2020; pp. 1–5.

[11] Govil, N.; Sharma, A. Information extraction on requirements prioritization approaches in agile software development processes. In Proceedings of the 5th International Conference on Computing Methodologies and Communication, Erode, India,8–10 April 2021; pp. 1097–1100.

[12] Ramesh, B.; Cao, L.; Baskerville, R. Agile requirements engineering practices and challenges: An empirical study. Inf. Syst. **2010**,20, 449–480.

[13] Razali, R.; Anwar, F.; Rahman, M.A.; Ismail, F.F. Mixed Methods Research: Insights from Requirements Engineering. J. Bus. Res. **2016**, 14, 125–134. Available online: https://academicpublishing.org/index.php/ejbrm/article/view/1347

[14] Jarz¸ebowicz, A.;Weichbroth, P. A qualitative study on non-functional requirements in agile software development. IEEE Access **2021**, 9, 40458–40475.

[15] Bjarnason, E.; Unterkalmsteiner, M.; Borg, M.; Engström, E. A multi-case study of agile requirements engineering and the use of test cases as requirements. Inf. Softw. Technol. **2016**, 77, 61–79.

[16] Neto, F.G.D.O.; Horkoff, J.; Knauss, E.; Kasauli, R.; Liebel, G. Challenges of Aligning Requirements Engineering and System Testing in Large-Scale Agile: A Multiple Case Study. In Proceedings of the IEEE 25th International Requirements Engineering Conference Workshops, Lisbon, Portugal, 4–8 September 2017; pp. 315–322.

[17] Hoy, Z. Requirements engineering for agile teams and startups: A challenge-solution gap analysis from a systematic literature review. In Software Engineering Practices for Start-Ups; CRC Press (Taylor & Francis): Boca Raton, FL, USA, 2023.

[18] Webster, J.; Watson, R.T. Analyzing the Past to Prepare for the Future: Writing a Literature Review. MIS Q. **2002**, 26, xiii–xxiii. Available online: https://www.jstor.org/stable/4132319

[19] Karhapää, P.; Behutiye, W.; Rodríguez, P.; Oivo, M.; Costal, D.; Franch, X.; Aaramaa, S.; Chora´s, M.; Partanen, J.; Adherve, A. Strategies to manage quality requirements in agile software development: A multiple case study. Empir. Softw. Eng. **2021**.

[20] Dybå, T.; Dingsøyr, T. Empirical studies of agile software development: A systematic review. Inf. Softw. Technol. **2008**, 50, 833–859. Information **2023**.

[21] Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ **2021**.

[22] Alam, S.; Shah, S.A.A.; Bhatti, S.N.; Jadi, A.M. Impact and Challenges of Requirements Engineering in Agile Methodologies: A Systematic Review. Int. J. Adv. Comput. Sci. Appl. **2017**.

[23] Curcio, K.; Navarro, T.; Malucelli, A.; Reinehr, S. Requirements engineering: A systematic mapping study in agile software development. J. Syst. Softw. **2018**.

[24] Elghariani, K.; Kama, N. Review on Agile requirements engineering challenges. In Proceedings of the 3rd International Conference on Computer and Information Sciences, Kuala Lumpur, Malaysia, 15–17 August 2016.

[25] Inayat, I.; Salim, S.S.; Marczak, S.; Daneva, M.; Shamshirband, S. A systematic literature review on agile requirements engineering practices and challenges. Comput. Hum. Behav. **2015.**

[26] Okesola, J.; Adebiyi, M.; Okokpujie, K.; Odepitan, D.; Goddy-Worlu, R.; Iheanetu, O.; Omogbadegun, Z.; Adebiyi, A. A Systematic Review of Requirement Engineering Practices in Agile Model. Int. J. Mech. Eng. **2019**, 10, 671–687. Available online: http://eprints.lmu.edu.ng/3119/1/Agile%20model_Okesola%20IJMET.pdf

[27] Muhammad, A.; Siddique, A.; Mubasher, M.; Aldweesh, A.; Naveed, Q.N. Prioritizing Non-Functional Requirements in Agile Process Using Multi Criteria Decision Making Analysis. IEEE Access **2023**.

[28] Harvie, D.P.; Agah, A. Targeted Scrum: Applying Mission Command to Agile Software Development. IEEE Trans. Softw. **2016**.

[29] Alsaqaf,W.; Daneva, M.;Wieringa, R. Quality requirements challenges in the context of large-scale distributed agile: An empirical study. Inf. Soft. Technol. **2019**.

[30] Hess, A.; Diebold, P.; Seyff, N. Understanding information needs of agile teams to improve requirements communication. J. Ind. Inf. Integr. **2019**.

[31] Firdaus, A.; Ghani, I.; Abg Jawawi, D.N.;Wan Kadir,W.M.N. Non functional requirements (NFRs) traceability metamodel for agile development. J. Teknol. **2015**.

[32] Alrezaamiri, H.; Ebrahimnejad, A.; Motameni, H. Software requirement optimization using a fuzzy artificial chemical reaction optimization algorithm. Soft Comput. **2019**.

[33] Drury-Grogan, M.L.; Conboy, K.; Acton, T. Examining decision characteristics & challenges for agile software development. J. Syst. Softw. **2017**.

[34] Borrego, G.; Morán, A.L.; Palacio, R.R.; Vizcaíno, A.; García, F.O. Towards a reduction in architectural knowledge vaporization during agile global software development. Inf. Softw. Technol. **2019**.

[35] Martínez-García, J.R.; Castillo-Barrera, F.-E.; Palacio, R.R.; Borrego, G.; Cuevas-Tello, J.C. Ontology for knowledge condensation to support expertise location in the code phase during software development process. IET Softw. **2020**.

[36] Heikkilä, V.T.; Paasivaara, M.; Lasssenius, C.; Damian, D.; Engblom, C. Managing the requirements flow from strategy to release in large-scale agile development: A case study at Ericsson. Empir. Softw. Eng. **2017**.

[37] Rojas, L.A.; Macías, J.A. Toward collisions produced in requirements ranking: A qualitative approach and experimental study. J. Syst. Softw. **2019**.

[38] Mishra, D.; Mishra, A. Complex software project development: Agile methods adoption. J. Softw. Maint. **2011**.

[39] Behutiye,W.; Rodriguez, P.; Oivo, M. Quality Requirement Documentation Guidelines for Agile Software Development. IEEE Access **2022**.

[40] Wagner, S.; Fernández, D.M.; Kalinowski, M.; Felderer, M. Agile Requirements Engineering in Practice: Status Quo and Critical Problems. CLEI Electron. J. **2018**.

[41] Al-Ta'ani, R.H.; Razali, R. Process Model for Systematic Requirements Prioritisation Process in an Agile Software Development Environment Based on 5S Approach: Empirical Study. J. Theor. Appl. Inf. Technol. **2017**.

[42] Martins, H.F.; de Oliveira, A.C.; Canedo, E.D.; Kosloski, R.A.D.; Paldês, R.A.; Oliveira, E.C. Design thinking: Challenges for software requirements elicitation. Information **2019**, 10.

[43] Mathrani, A.; Wickramasinghe, S.; Jayamaha, N.P. An evaluation of documentation requirements for ISO 9001 compliance in scrum projects. TQM J. **2022**.

[44] Madampe, K.; Hoda, R.; Grundy, J.A. Faceted Taxonomy of Requirements Changes in Agile Contexts. IEEE Trans. Softw. **2022**.

[45] Hoda, R.; Murugesan, L.K. Multi-level agile project management challenges: A self-organizing team perspective. J. Syst. Softw. **2016.**

[46] Mishra, D.; Mishra, A. Managing requirements in market-driven software project: Agile methods view. Tech. Gaz. **2010**.

[47] Chen, P.-S.; Chen, G.Y.-H.; Lien, S.-F.; Huang, W.-T. Using Scrum and unified modelling language to analyze and design an automatic course scheduling system. J. Chin. Inst. Eng. **2019**.

[48] Gupta, A.; Poels, G.; Bera, P. Using Conceptual Models in Agile Software Development: A Possible Solution to Requirements Engineering Challenges in Agile Projects. IEEE Access **2022**.

[49] Gaikwad, V.; Joeg, P. A Case Study in Requirements Engineering in Context of Agile. Int. J. Appl. Eng. **2017**.

[50] Tanveer, B.; Guzmán, L.; Engel, U.M. Effort estimation in agile software development: Case study and improvement framework. J. Softw. Evol. Process **2017**.

[51] Bhalerao, S.; Ingle, M. Incorporating Vital Factors in Agile Estimation through Algorithmic Method. Int. J. Comput. Sci. **2009.**

[52] Alsaadi, B.; Saeedi, K. Data-driven effort estimation techniques of agile user stories: A systematic literature review. Artif. Intell. **2022**.

[53] Dragicevic, S.; Celar, S.; Turic, M. Bayesian network model for task effort estimation in agile software development. J. Syst. Softw. **2017**.

[54] Usman, M.; Britto, R.; Damm, L.-O.; Börstler, J. Effort estimation in large-scale software development: An industrial case study. Inf. Softw. Technol. **2018**.

[55] Elghariani, K.; Kama, N.; Mohd Azmi, N.F.; Abu bakar, N.A. Implicit thinking knowledge injection framework for Agile requirements engineering. Int. J. Adv. Comput. Sci. Appl. **2018.**

[56] Vithana, V.N. Scrum requirements engineering practices and challenges in offshore software development. Int. J. Comput. Appl. **2015.**

[57] Kasauli, R.; Liebel, G.; Knauss, E.; Gopakumar, S.; Kanagwa, B. Requirements Engineering Challenges in large-scale agile system development. In Proceedings of the 25th

International Requirements Engineering Conference, Lisbon, Portugal, 4–8 September 2017; pp. 352–361.

[58] Saeeda, H.; Dong, J.; Wang, Y.; Abid, M.A. A proposed framework for improved software requirements elicitation process in SCRUM: Implementation by a real-life Norway-based IT project. J. Softw. Evol. Process. **2020**.

[59] Santos, P.O.; de Carvalho, M.M. Exploring the challenges and benefits for scaling agile project management to large projects: A review. Requir. Eng. **2022**.

[60] El-Najar, T.; Ahmad, I.; Alkandari, M. Easycomm: A framework and tool to solve client communication problem in Agile development. IAENG Int. J. Comput. Sci. **2019**.

[61] Medeiros, J.; Vasconcelos, A.; Silva, C.; Goulão, M. Quality of software requirements specification in agile projects: A cross-case analysis of six companies. J. Syst. Softw. **2018.**

[62] Kasauli, R.; Knauss, E.; Horkoff, J.; Liebel, G.; de Oliveira Neto, F.G. Requirements engineering challenges and practices in large-scale agile system development. J. Syst. Softw. **2021**.

[63] Levy, Y.; Stern, R.; Sturm, A.; Mordoch, A.; Bitan, Y. An impact-driven approach to predict user stories instability. Requir. Eng. **2022**.

[64] Thomas, M.; Senapathi, M. Agile requirements engineering: An empirical analysis and evidence from a tertiary education context. Issues Inf. Sci. Inf. Technol. **2019**.

[65] Muntés-Mulero, V.; Ripolles, O.; Gupta, S.; Dominiak, J.; Willeke, E.; Matthews, P.; Somosköi, B. Agile risk management for multi-cloud software development. IET Softw. **2019**.

[66] Behutiye,W.; Rodríguez, P.; Oivo, M.; Aaramaa, S.; Partanen, J.; Abhervé, A. Towards optimal quality requirement documentation in agile software development: A multiple case study. J. Syst. Softw. **2022**.

[67] Jain, R.; Cao, L.; Mohan, K.; Ramesh, B. Situated boundary spanning: An empirical investigation of requirements engineering practices in product family development. ACM Trans. Manag. Inf. Syst. **2014**.

[68] Bernier, C.; Dubé, L.; Roy, V. An Agile Method, a Contractual Relationship, and Distance: A Triad of Challenging Conditions for a Successful System Development Project. J. Inf. Technol. Case Appl. Res. **2012**.

[69] Urbieta, M.; Antonelli, L.; Rossi, G.; do Prado Leite, J.C.S. The impact of using a domain language for an agile requirement management. Inf. Softw. Technol. **2020**, 127, 106375.

[70] Gahyyur, S.A.K.; Razzaq, A.; Hasan, S.Z.; Ahmed, S.; Ullah, R. Evaluation for feature driven development paradigm in context of architecture design augmentation and perspective implications. Int. J. Adv. Comput. Sci. Appl. **2018**.

[71] Rosenberger, P.; Tick, H.J. Agile enhancement of critical PMBOK V6 processes. J. Mod. Proj. Manag. **2021**.

[72] Al-Zewairi, M.; Biltawi, M.; Etaiwi,W.; Shaout, A. Agile Software Development Methodologies: Survey of Surveys. J. Comput. Commun. **2017**.

[73] Amna, A.R.; Poels, G. Ambiguity in user stories: A systematic literature review. Inf. Softw. Technol. **2022**.

[74] Hoy, Z. Dataset for Agile Requirements Engineering Challenges-Solutions Updated May 2023; University of Portsmouth Research Portal: Portsmouth, UK, 2023.

[75] Alavi, M.; Carlson, P. A review of MIS research and disciplinary development. J. Manag. Inf. Syst. **1992**.

[76] Medeiros, J.; Vasconcelos, A.; Silva, C.; Goulão, M. Requirements specification for developers in agile projects: Evaluation by two industrial case studies. Inf. Softw. Technol. **2020**.

[77] Martini, A.; Bosch, J. On the interest of architectural technical debt: Uncovering the contagious debt phenomenon. J. Softw. J. Softw. Evol. Process **2017**.

[78] Lohan, G.; Conboy, K.; Lang, M. Examining customer focus in IT project management: Findings from Irish and Norwegian case studies. Scand. J. Inf. Syst. **2011**.

[79] Liebel, G.; Knauss, E. Aspects of modelling requirements in very-large agile systems engineering. J. Syst. Softw. **2023.**

[80] Canedo, E.D.; Calazans, A.T.S.; Bandeira, I.N.; Costa, P.H.T.; Masson, E.T.S. Guidelines adopted by agile teams in privacy requirements elicitation after the Brazilian general data protection law (LGPD) implementation. Requir. Eng. **2022**.

[81] Werner, C.; Li, Z.S.; Lowlind, D.; Elazhary, O.; Ernst, N.; Damian, D. Continuously Managing NFRs: Opportunities and Challenges in Practice. IEEE Trans. Softw. **2022**.

[82] Brataas, G.; Martini, A.; Hanssen, G.K.; Ræder, G. Agile elicitation of scalability requirements for open systems: A case study. J. Syst. Softw. **2021**.

[83] Oriol, M.; Martínez-Fernández, S.; Behutiye, W.; Farré, C.; Kozik, R.; Seppänen, P.; Vollmer, A.M.; Rodríguez, P.; Franch, X.; Aaramaa, S.; et al. Data-driven and tool-supported elicitation of quality requirements in agile companies. Softw. Qual. **2020.**

[84] Traini, L. Exploring Performance Assurance Practices and Challenges in Agile Software Development: An Ethnographic Study. Empir. Softw. Eng. **2022**.

[85] Del Sagrado, J.; Del Águila, I.M.; Orellana, F.J. Multi-objective ant colony optimization for requirements selection. Empir. Softw. Eng. **2015.**

[86] Al-Ta'ani, R.H.; Razali, R. A framework for requirements prioritisation process in an agile software development environment: Empirical study. Int. J. Adv. Sci. Eng. Inf. Technol. **2016**.

[87] Trkman, M.; Mendling, J.; Krisper, M. Using business process models to better understand the dependencies among user stories. Inf. Softw. Technol. **2016**.

[88] Trkman, M.; Mendling, J.; Trkman, P.; Krisper, M. Impact of the conceptual model's representation format on identifying and understanding user stories. Inf. Softw. Technol. **2019**.

[89] de Souza, P.L.; de Souza,W.L.; Ferreira Pires, L. ScrumOntoBDD: Agile software development based on scrum, ontologies and behaviour-driven development. J. Braz. Comput. Soc. **2021**.

[90] Raharjana, I.K.; Siahaan, D.; Fatichah, C. User Stories and Natural Language Processing: A Systematic Literature Review. IEEE Access **2021**.

[91] Usman, M.; Petersen, K.; Börstler, J.; Santos Neto, P. Developing and using checklists to improve software effort estimation: A multi-case study. J. Syst. Softw. **2018**.

[92] Rahy, S.; Bass, J.M. Managing non-functional requirements in agile software development. IET Softw. **2022**.

[93] Ernst, N.A.; Borgida, A.; Jureta, I.J.; Mylopoulos, J. Agile requirements engineering via paraconsistent reasoning. Inf. Syst. **2014**.

[94] Sarkan, H.M.; Ahmad, T.P.S.; Bakar, A.A. Using JIRA and Redmine in Requirements Development for Agile Methodology. In Proceedings of the 5th Malaysian Conference in Software Engineering, Johor Bahru, Malaysia, 13–14 December 2011.

[95] Farid, W.M.; Mitropoulos, F.J. NORMATIC: A visual tool for modeling non-functional requirements in agile processes. In Proceedings of the 2012 IEEE Southeastcon, Orlando, FL, USA, 15–18 March 2012.

[96] Farid,W.M.; Mitropoulos, F.J. Novel lightweight engineering artifacts formodeling non-functional requirements in agile processes. In Proceedings of the 2012 IEEE Southeastcon, Orlando, FL, USA, 15–18 March 2012.

[97] Anand, R.V.; Dinakaran, M. Handling stakeholder conflict by agile requirements prioritization using Apriori technique. Comput.Electr. Eng. **2017**.

[98] Mohammad, Abdulghafour, and Job Mathew Kollamana. "Causes and Mitigation Practices of Requirement Volatility in Agile Software Development." *Informatics*. Vol. 11. No. 1. MDPI, 2024.

[99] Koulecar, Neha, and Bachan Ghimire. "Agile Requirement Change Management Model for Global Software Development." *arXiv preprint arXiv:2402.14595* (2024).

[100] Bambazek, Peter, Thomas Hofer, and Iris Groher. "Scrum Sustainability Poker: Assessing the Sustainability Effects of User Stories in Agile Software Development." *REFSQ Workshops*. 2024.

[101] Barros, Leonor, Carlos Tam, and Joao Varajao. "Agile software development projects–Unveiling the human-related critical success factors." *Information and Software Technology* 170 (2024): 107432.

[102] Hoy, Zoe, and Mark Xu. "Agile Software Requirements Engineering Challenges-Solutions—A Conceptual Framework from Systematic Literature Review." *Information* 14.6 (2023): 322.

[103] Herwanto, Guntur Budi, Gerald Quirchmayr, and A. Min Tjoa. "Leveraging NLP Techniques for Privacy Requirements Engineering in User Stories." *IEEE Access* (2024).