# مروری بر تکنیک‌های اولویت‌بندی نیازمندی‌های نرم‌افزار

**آذین سادات پیشداد**
کارشناس ارشد دانشکده کامپیوتر دانشگاه آزاد اسلامی واحد دولت آباد

**گلناز آقایی قزوینی**
استادیار دانشکده کامپیوتر دانشگاه آزاد اسلامی واحد دولت آباد

**بابک نیک مرد**
استادیار دانشکده کامپیوتر دانشگاه آزاد اسلامی واحد دولت آباد

## چکیده

در توسعه یک نرم‌افزار با تعدادی از کارها و نیازمندی‌ها روبرو هستیم که ترتیب انجام هر فعالیت نیاز به اولویت‌بندی داشته تا بتوان مدیریتی بر انتشار نرم‌افزار، کنترل بودجه، زمان‌بندی و مصرف منابع داشته باشیم. پیچدگی فرایند اولویت‌بندی، اهمیت استفاده از تکنیک‌ها و فریمورک‌های توسعه یافته را توجیه کرده و بهینه‌سازی این روش‌ها در بهبود مهندسی نیازمندیها را لازم دانسته است. در این تحقیق مروری بر مجموعه‌ای از تکنیک‌های اولویت‌بندی نیازمندی‌های نرم‌افزار به صورت دسته‌بندی شده جمع‌آوری گردیده است و مزایا و معایب هر یک با توجه به محدودیت‌هایی همچون سهولت استفاده، اثربخشی، مقیاس‌پذیری، کارایی، جذابیت، مصرف زمان، قابلیت اطمینان، تحمل خطا و پیچیدگی مورد بررسی قرار گرفته است. دسته‌بندی تکنیک‌ها شامل تکنیک‌های مبتنی بر مقیاس اسمی، مقیاس نسبت و مقیاس ترتیبی، تکنیک‌های مبتنی بر روش‌های کیفی و کمی که به سه زیرمجموعه روش‌های مبتنی بر نقشه برداری، امتیازدهی و بازی تقسیم می‌شوند، تکنیک‌های مبتنی بر منطق فازی، تکنیک‌های مبتنی بر الگوریتم بهینه‌سازی و تکنیک‌های مبتنی بر یادگیری ماشین می‌باشند. این مطالعه سعی کرده با بررسی ۵۷ مقاله منتشر شده از سال ۲۰۱۴ تا ۲۰۲۳، تکنیک‌های موجود را برای انتخاب بهترین تصمیم در اولویت‌بندی نیازمندی‌های نرم‌افزاری جهت دریافت نتیجه بهینه معرفی کند.

**واژگان کلیدی:** اولویت‌بندی نیازمندی‌ها، فریمورک‌های اولویت‌بندی، نیازمندی‌های نرم‌افزار

## مقدمه

اولویت‌بندی نیازهای نرم‌افزار (SRP[1]) اصولی است که نرم‌افزار دردست توسعه را قادر می‌سازد تا مطابق انتظار عمل کند. اولویت‌بندی نیازمندی‌ها به عنوان یک فرآیند تصمیم‌گیری چند معیاره پیچیده در نظر گرفته می‌شود که قبل از طراحی معماری یا کدنویسی اجرا شده و مزایای زیادی به منظور پیاده‌سازی یک سیستم نرم‌افزاری با الزامات ترجیحی دارد. الزامات توسط ذینفعان اصلی، که می‌تواند توسعه‌دهنده نرم‌افزار یا مشتری باشد اولویت‌بندی شده که یک محصول نرم‌افزاری تولید شود. توسعه‌دهنده نرم‌افزار الزامات را بر اساس نیازهای بازار اولویت‌بندی و مشتری این کار را بر اساس نیازهای تجاری خود انجام می‌دهد. در نتیجه، به منظور تعیین اولویت الزامات، معیارهایی شامل: (۱) منافع، (۲) هزینه‌ها، (۳) خطرات و ریسک، (۴) جریمه‌ها، (۵) زمینه کسب‌وکار، و (۶) زمینه فنی و ویژگی‌های الزامات در نظر گرفته شده و رتبه‌بندی نیازها صورت می‌گیرد (Riegel et al., 2015). اولویت‌بندی نیازمندی‌ها شامل رتبه‌بندی نیازمندی‌های نرم‌افزار به ترتیبی خاص است که زمان انتظار برای الزامات پیش‌نیاز بر زمان‌بندی پروژه تأثیر نامطلوبی نگذارد و وابستگی متقابل ایجاد نکند، از طرفی ریسک تاخیر با تخصیص اولویت کم به الزامات حیاتی بالا نرود، در نتیجه قانون تعادل برای جلوگیری از تأخیرهای غیرضروری و در عین حال حصول اطمینان از تحویل به موقع، برقرار شود ( Yaseen, Ibrahim, et al., 2019). در توسعه نرم‌افزار، اکثریت قریب به اتفاق وظایف، وابستگی اجباری ندارند و مدیر پروژه باید تصمیم بگیرد کدام کار ابتدا تکمیل شود. اولویت‌بندی مستمر و مناسب وظایف (در اصطلاح چابک) به یک عامل موفقیت حیاتی برای هر پروژه توسعه نرم‌افزار تبدیل می‌شود، زیرا تضمین می‌کند که اهداف حیاتی شرکت در کانون توجه قرار دارند و می‌توان آنها را برآورده کرد. (Bugayenko et al., 2023).

الزامات عملکردی نرم‌افزار در توسعه سیستم‌های مبتنی بر مؤلفه CBD[2]، با تأکید بر چابکی و تحویل به موقع محصول با کیفیت بالا بسیار مهم بوده که با اولویتی مناسب برای استفاده مجدد و ساختاربندی صحیح در یک محیط تکامل نرم‌افزار، پیوسته با چالش‌هایی مانند ناقص بودن، ناسازگاری، پیکربندی‌های مبهم مقابله می‌کند (Ali et al., 2021; Nguyen et al., 2019) . یک فرآیند تثبیت شده برای کشف و اولویت‌بندی محتوای نقشه راه محصول به روش صحیح برای موفقیت یک شرکت بسیار مهم است. با این حال، اکثر شرکت ها موارد نقشه راه محصول خود را بر اساس نظرات کارشناسان یا مدیریت اولویت‌بندی می‌کنند. به علت نامشخص بودن معیار تصمیم‌گیری، مشکلاتی فراهم خواهد شد و سازمان با چالش انتخاب مرتبط‌ترین ویژگی‌ها مواجه می‌شود( Münch et al., 2019). علاوه بر این، افزایش پویایی بازار، فناوری‌های در حال تکامل سریع و تغییر سریع رفتار مشتری، انجام فرآیند اولویت‌بندی را پیچیده می‌کند. بنابراین، بسیاری از شرکت‌ها در تلاش برای یافتن و ایجاد تکنیک‌های مناسب و کنترل شده برای اولویت‌بندی نقشه راه محصول خود هستند. به‌ویژه شرکت‌هایی که در تجارت نرم‌افزاری فعالیت می‌کنند، با چالش افزایش پویایی بازار که مستلزم عدم قطعیت‌های بالایی است، مواجه هستند. این وضعیت بر الزامات فرآیند اولویت‌بندی تأثیر می‌گذارد و بنابراین تکنیک‌هایی مورد نیاز است که بتوان آنها را به طور کارآمد و مؤثر اجرا کرد(Trieflinger et al., 2021).

مهندسی نیازمندی‌های اولیه و برنامه‌ریزی انتشار اولیه بیشتر بر روی رویکردهای اولویت‌بندی مبتنی بر ابزار متمرکز است در حالی که برنامه‌ریزی حداقل محصول قابل دوام و برنامه‌ریزی انتشار یکپارچه بر رویکردهای اولویت‌بندی مبتنی بر بهینه‌سازی تمرکز دارد. در سطح فنی، اولویت‌بندی مبتنی بر بهینه‌سازی اغلب مبتنی بر یک رویکرد ترکیبی است که در آن شناسایی و تجمیع ترجیحات

[1] Software Requirement prioritization
[2] Component Base Development

ذینفعان توسط تجزیه و تحلیل سودمندی پشتیبانی می‌شود و بهینه‌سازی بر اساس استدلال محدودیت انجام می‌شود. رویکردهای مبتنی بر سودمندی بر تجزیه و تحلیل الزامات داده شده با توجه به مجموعه ای از ابعاد علاقه و کمتر بر بهینه سازی خودکار تمرکز می کنند، ابتدا، الزامات فردی با توجه به ابعاد بهره (به عنوان مثال، سطح ریسک یک نیاز و ارتباط تجاری یک نیاز) ارزیابی می‌شود، سپس مطلوبیت مورد نیاز بر اساس مجموع مقادیر مطلوبیت خاص بهره تعیین شده که ابعاد بهره را می‌توان با یک وزن مرتبط کرد (Huang, 2011; Ninaus et al., 2014). زمانی که با مجموعه‌ای از نیازها سروکار داریم، فرآیندهای اولویت‌بندی دستی بسیار پرهزینه می‌شوند (Felfernig, 2021). روش‌های فعلی اولویت‌بندی با محدودیت‌هایی مواجه هستند، زیرا تکنیک‌های اولویت‌بندی فعلی برای الزامات عملکردی به جای اولویت‌بندی نیازمندی‌ها بر اساس وابستگی‌های داخلی یک نیاز به نیازهای دیگر، بر پاسخ‌های ذینفعان تکیه می‌کنند. علاوه بر این، نیاز به طبقه‌بندی الزامات بر اساس اهمیت آنها وجود دارد، یعنی چقدر برای سایر الزامات مورد نیاز هستند یا به سایر الزامات وابسته هستند (Yaseen, Mustapha, et al., 2019) ساختار باقیمانده این مقاله به شرح زیر است: بخش ۲ روش تحقیق را توصیف می‌کند، یعنی روند مرور سیستماتیک، سوالات تحقیق و استراتژی جستجو را بیان کرده، بخش ۳ تکنیک‌های اولویت‌بندی نیازمندی‌های نرم‌افزار را مورد بررسی قرار می‌دهد و بخش ۴ در نهایت نتیجه‌گیری می‌کند.

## روش تحقیق

این مطالعه، مروری سیستماتیک بوده و با در نظر گرفتن سوالاتی به اهداف پژوهشی پاسخ می‌دهد. با استفاده از کلمه‌کلیدی‌های مرتبط، مقالات مناسب در موضوع تکنیک‌های اولویت‌بندی نیازمندی‌های نرم‌افزار را انتخاب کرده و دسته‌بندی می‌کند. روش تحقیق به صورت شکل ۱ می‌باشد.


**شکل-۱: روش تحقیق**

### سوالات تحقیق

در این تحقیق سه پرسش پژوهشی، با توجه به مطالعات منتشر شده از ۲۰۱۴ تا ۲۰۲۳ به منظور جمع‌آوری و دسته‌بندی تکنیک‌های اولویت‌بندی نیازمندیهای نرم‌افزار و محدودیت‌های هر روش برای کاهش تلاش‌های تصمیم‌گیری، در نتیجه به حداقل رساندن خطاها و تسریع در اجرای نیازمندی‌ها بیان شده است.

RQ1: چه تکنیک‌های اولویت‌بندی برای نیازمندی‌های نرم‌افزار وجود دارد؟

RQ2: چگونه تکنیک‌ها را به صورت دسته‌بندی شده تقسیم کنیم؟

RQ3: چه تکنیک‌هایی به صورت ترکیبی از روش‌ها ایجاد شده‌اند؟

RQ4: چه محدودیت‌هایی در هر تکنیک وجود دارد؟

**استراتژی جستجو**

با توجه به سوالات پژوهش، کلمات کلیدی با پیوندهای AND و OR بولی در کتابخانه‌های دیجیتال به جای جستجوی ساده ترکیب شده و از رشته جستجوی زیر در عنوان، چکیده و کلمات کلیدی استفاده شده است:

Software Requirements AND requirement engineering AND Multi-objective optimization AND Prioritization AND Method OR Technique

این رشته به همراه محدودیت‌هایی: (i) محدودیت بر اساس نوع منبع در دسترس (یعنی مقالات کنفرانس و مقالات مجلات)، (ii) محدودیت بر اساس سال انتشار، از ۲۰۱۴ و (iii) محدودیت حوزه موضوعی، یعنی علوم کامپیوتر از sciencedirect ، ieeexplore ، springer تکمیل شد. پس از آن، با استفاده از فرآیند گلوله برفی[3] ، مراجع هر مطالعه انتخاب شده، برای شناسایی مطالعات مهمی که ممکن است در طول فرآیند جستجوی اولیه از قلم افتاده باشند، جستجو شد. در ابتدا عنوان و سپس چکیده مطالعه شد و هر مقاله‌ای که به موضوع بحث مرتبط نبود یا قادر به پاسخگویی به سؤالات تحقیق بیان شده نبود، از فهرست مطالعات حذف شد.

**معیارهای ورود و خروج**

در این پژوهش از معیارهای ورود (ICs[4]) و معیارهای خروج (ECs[5]) زیر استفاده کردیم. معیارهای ورود عبارتند از:

IC1: مقاله مستقیماً به موضوع بررسی ما مربوط می‌شود.

IC2: مقاله به سوالات تحقیق می‌پردازد.

IC3: مقاله در یک مجله یا کنفرانس معتبر منتشر شده است.

IC4: مقاله به زبان انگلیسی است.

IC5: مقاله برای دانلود در دسترس است.

معیارهای خروج عبارتند از:

[3] snowballing process
[4] Inclusion criteria
[5] Exclusion criteria

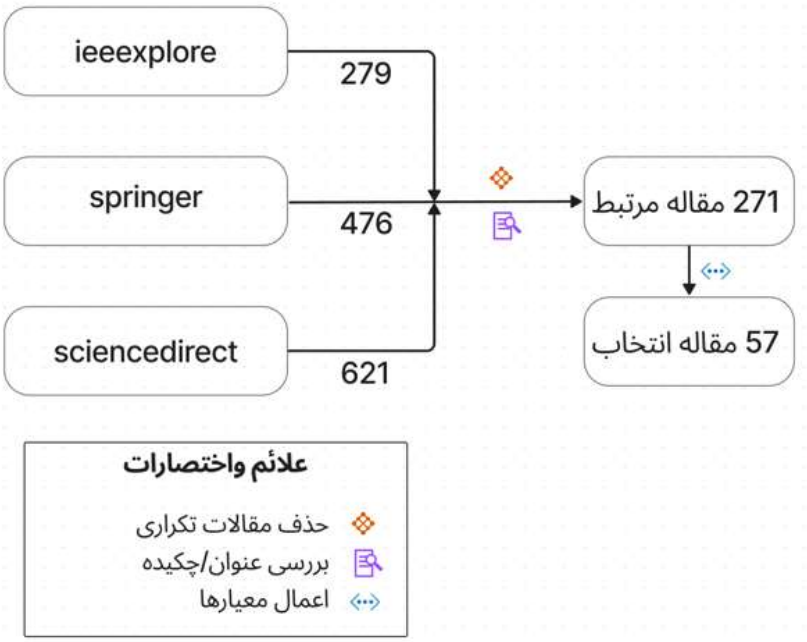
EC1: مقاله در مورد یک موضوع جانبی که با موضوع اصلی مقاله مرتبط است صحبت می‌کند.

EC2: مقاله در مورد روشی صحبت می‌کند که در دسته‌بندی خاصی قرار ندارد.

EC3: مقاله مورد بررسی به زبانی غیر از انگلیسی می‌باشد.

**اجرای فرآیند استخراج داده ها**

شکل۲ یک نمای کلی از فرآیند تحقیق را ارائه می‌دهد که از آن برای حذف تدریجی مطالعات استفاده کردیم. در نهایت ۵۷ مطالعه انتخاب شدند.




**شکل-۲: اجرای فرآیند استخراج داده ها**

**تهدید اعتبار**

در این بخش، چند تهدید شناخته شده برای اعتبار نتایج این مطالعه را شناسایی کردیم.

۱. جمع آوری ناقص داده‌ها تهدیدی برای این مقاله مروری است. که البته با بررسی دستی برای جلوگیری از حذف نادرست مطالعات مطلوب و مشخص کردن کامل معیارهای انتخابی که اهداف تحقیق در نظر داشته، تا حدودی از این تهدید پیشگیری کرده‌ایم.

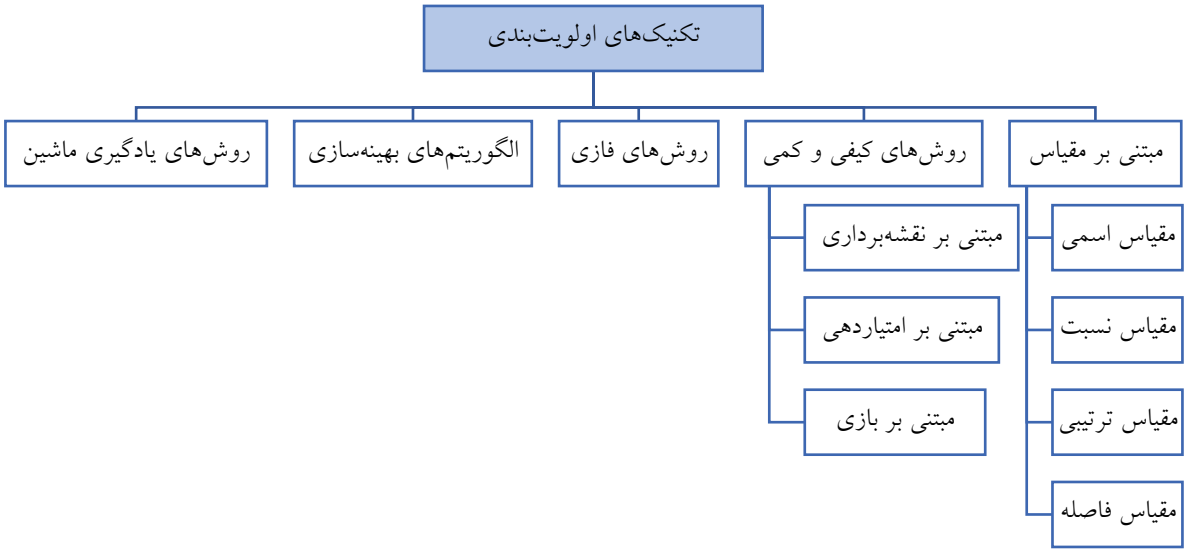
۲. استفاده از معیارهای کیفیت مطابق با دستورالعمل‌های SLR در انتخاب پژوهش مناسب در این تحقیق گنجانده نشده است. که البته تعداد کم مقالات انتخابی و مطالعه کامل آنها، کیفیت تحقیق را کاهش نمی‌دهد.

**تکنیک‌های اولویت‌بندی نیازمندی‌های نرم‌افزار**

پروژه‌های نرم‌افزاری در مقیاس‌های کم، متوسط، بزرگ و فوق العاده بزرگ توسعه پیدا می‌کنند. مجموعه‌های مقیاس کم، تعداد نیاز کمتر از صد مورد دارند، مجموعه‌های مقیاس متوسط را می‌توان با صدها نیاز، مجموعه‌های مقیاس بزرگ را با هزاران نیاز، و سیستم‌هایی با مقیاس فوق العاده بزرگ پروژه‌هایی با صدها هزار ذینفع و نیازمندی هستند ( Somohano-Murrieta et al. ,2020). هر چه تعداد ویژگی‌ها و نیازها افزایش پیدا می‌کند، اهمیت ترتیب انجام هر کدام نیز افزایش یافته و اولویت‌بندی با استفاده از تکنیک‌های مناسب معنا می‌یابد. پیش از ارائه تکنیک‌ها و چارچوب‌های اولویت‌بندی، سه مرحله: آماده‌سازی الزامات توسط رهبر تیم، اجرا تصمیم‌گیری الزامات و در اخر ارائه نتایج اجرا برای ذینفعان پروژه انجام می‌شده است، اما با گستردگی نیازها، مناسب بودن و سودمندی یک تکنیک اولویت‌بندی به نوع بلوغ محصول و همچنین جنبه‌های سازمانی (مانند طرز فکر کارکنان یا فرهنگ سازمان) بستگی دارد، که از کدام تکنیک با محدودیت‌های کمتر می‌توان استفاده کرد ( Hudaib et al., 2018; Trieflinger et al., 2021).

یک تکنیک موثر اولویت‌بندی نیازمندی‌ها باید معیارهای مختلفی را برای پاسخگویی به ذینفعان در برگیرد. دقت کافی برای اطمینان از همسویی نتایج با نیازهای ذینفعان حتی در پروژه‌هایی با مقیاس بزرگ و در نتیجه مقیاس‌پذیری تکنیک بسیار مهم است. کارایی ضروری است، با تاکید بر اهمیت به حداقل رساندن مقایسه برای راه‌حل‌های سریع‌تر، در حالی که به طور موثر از منابع استفاده می‌شود زیرا ارزش هر نیاز از هزینه پیاده‌سازی آن بیشتر بوده و با مقایسه‌های زوجی می‌توان برای تعیین هزینه نسبی هر الزام وزن‌دهی انجام داد. هر تکنیک مجموعه‌های متعددی از الزامات رتبه‌بندی تولید می‌کند و امکان انتخابی انعطاف‌پذیر از راه‌حل‌های پیشنهادی را فراهم می‌سازد. یک ساختار ایده‌آل به منظور سادگی در استفاده و اولویت‌بندی خودکار از محاسبات کمتر پیچیده برای تولید نتیجه نهایی جهت جلب اعتماد و توجه کاربر استفاده می‌کند، همچنین به منظور جلوگیری از افزونگی برای افزایش قابلیت اطمینان سیستم الزامات اضافه را نادیده می‌گیرد. به طور کلی، یک تکنیک ایده‌آل به طور یکپارچه: دقت، مقیاس پذیری، کارایی، مقرون به صرفه بودن، مجموعه‌ای از راه حل‌ها، سهولت استفاده را ادغام می‌کند، و افزونگی را برای اولویت‌بندی نیازمندی‌های جامع برطرف می‌کند.

این مطالعه تکنیک‌ها را به پنج دسته: تکنیک‌های مبتنی بر منطق فازی، الگوریتم بهینه‌سازی، یادگیری ماشین، مقیاس پروژه و روش‌های کیفی و کمی تقسیم‌بندی کرده است و همراه با محدودیت‌هایشان بیان می‌کند، به طور مثال: تکنیک‌های اولویت‌بندی نیازمندی‌ها مبتنی بر بهینه‌سازی بر اساس الگوریتم‌ها در مقایسه با تکنیک‌های مبتنی بر منطق فازی و یادگیری ماشین، دقیق‌ترین، مقرون‌به‌صرفه و آسان‌ترین هستند، در صورتی که کمترین مقیاس‌پذیری، کمترین کارآمدی و کمترین اثربخشی را در مدیریت افزونگی دارند. تکنیک‌های اولویت‌بندی نیازمندی‌ها مبتنی بر منطق فازی دارای روش های مقیاس‌پذیر و مقرون به صرفه می‌باشند، اما دقت و راه‌حل بهینه‌ای ندارند. تکنیک‌های مبتنی بر یادگیری ماشین نتایج استثنایی در مقیاس‌پذیری، راه‌حل بهینه، کارایی و مدیریت افزونگی نشان داده‌اند( Ahmad et al., 2017a; Alrezaamiri et al., 2020; Mougouei et al., 2019; Quiroz et al., 2007) .




**شکل-۳: دسته‌بندی تکنیک‌های اولویت‌بندی**

**تکنیک‌های مبتنی بر مقیاس**

تکنیک‌های مبتنی بر مقیاس به دسته‌های مقیاس اسمی، ترتیبی، فاصله و نسبت تقسیم‌بندی می‌شوند( Achimugu et al., 2014; Hudaib et al., 2018; Vestola, 2010) .

مقیاس اسمی: مکانیسم‌های اولویت‌بندی مقیاس اسمی کلاس‌هایی را تولید می‌کنند که می‌توان اشیاء را در آن‌ها دسته‌بندی کرد. این بدان معناست که نیازمندی‌ها بر اساس اهمیتشان به دسته‌هایی با اولویت یکسان طبقه‌بندی می‌شوند. انتظار می‌رود که هر نیاز در هر زمان فقط به یک گروه یا زیرگروه تعلق داشته باشد. نیاز اولیه بر اساس بالاترین فرکانس در تمام نیازمندی‌ها تعیین می‌شود و حالت و مربع نیازهای رتبه‌بندی شده را محاسبه می‌کند.

مقیاس نسبت: مقیاس نسبت که قابل دوام تلقی می‌شود، نیازمندی‌ها را سفارش داده، فواصل نسبی را تعیین می‌کند و نسبت‌های بین نیازها را فراهم می‌کند. انجام کلیه اشکال محاسبات آماری از جمله میانگین هندسی، میانگین هارمونیک، ضریب تغییرات و همچنین استفاده از الگوریتم ها در فرآیند اولویت بندی را در نظر می‌گیرد.

مقیاس ترتیبی: این مقیاس با ارائه اطلاعاتی در مورد ترتیب نیازمندی‌ها، مقیاس اسمی را تکمیل می‌کند. اطلاعاتی در مورد رتبه‌بندی نیازهای مختلف ارائه کرده اما نشان نمی‌دهد که تا چه حد یک نیاز مهم‌تر از دیگری است و لیست رتبه‌بندی شده را تولید می‌کند. بر خلاف تکنیک‌های مقیاس نسبت، مقیاس ترتیبی نمی‌تواند به این سوال پاسخ دهند که چقدر یکی از الزامات در مقایسه با دیگری مهم است؟ در نتیجه فقط می‌تواند آن یک نیاز را بیان کند ولی کمی نمی‌کند.

مقیاس فاصله: این مقیاس اطلاعاتی در مورد اندازه فواصل بین مجموعه‌ای از الزامات را داشته و اختلاف بین الزامات را محاسبه می‌کند. مانند مقیاس ترتیبی، نظم را حفظ کرده و می تواند میانگین، انحراف معیار، همبستگی، رگرسیون، تحلیل واریانس را محاسبه کند و با تجمیع وزن‌ها و استفاده از الگوریتم استدلال شواهد داخلی (IER) نیازمندی‌ها را اولویت‌بندی کند. از این معیار برای پروژه‌هایی با مقیاس کوچک استفاده شده زیرا مقیاس‌پذیری مناسبی ندارد.

**جدول-۱: تکنیک‌های مبتنی بر مقیاس**

| مقیاس | تکنیک | توضیحات |
|---|---|---|
| مقیاس اسمی | Top ten (Lehtola et al., 2004) | در این تکنیک، از ذینفعان خواسته می‌شود تا ده نیاز برتر را از مجموعه نیازمندی‌ها انتخاب کنند اما امکان تناقض یا ابهام به دلیل نبود وزن در فرآیند رتبه‌بندی وجود دارد. |
| | Requirements triage (Babar et al., 2011) | این تکنیک با آموزش و متقاعد کردن ذینفعان برای درک هر یک از الزامات قبل از شروع رتبه‌بندی، اولویت نیازمندی‌ها را تعیین کرده و سپس آنها را مرتب می‌کند. این تکنیک مبتنی بر منطق فازی برای عملکرد کارآمد در مجموعه های بزرگ بوده اما مستعد خطا، اتکا به متخصص و زمان بر است. |
| | MosCow (Ahmad et al., 2017b; Hatton, 2007) | این تکنیک از روش توسعه نرم افزار پویا (DSDM) سرچشمه گرفته و نیازمندی‌ها را بر اساس اهمیت آن‌ها به گروه‌های «باید داشته باشد»، «می‌تواند داشته باشد» و «نخواهد داشته باشد» دسته‌بندی می‌کند. این رویکرد در یک محیط فازی گسترش یافته و همه الزامات در یک گروه اولویت دارای اولویت مشابه هستند، بدون اینکه اطلاعات اضافی یک نیاز را به عنوان اولویت بالاتر یا پایین‌تر از نیاز دیگری در همان گروه نشان دهد. از یک سو روشی ساده، سطح متوسط از قابلیت اطمینان و سطح بالایی از تحمل خطا را دارد و از سوی دیگر دارای پیچیدگی متوسط، سرعت کم و دارای مشکل در مقیاس‌پذیری برای پروژه‌های متوسط و بزرگ است. |
| مقیاس نسبت | VOP[6] - (Thakurta, 2013) | در این تکنیک، الزامات با ارزش‌های تجاری شناسایی شده و بر اساس رتبه‌بندی اولویت‌بندی می‌شوند، روشی ساده، موثر و با نتایج صحیح است، اما وابستگی نیازمندی‌ها را در روش محاسبات نادیده می‌گیرد و برای پروژه های بزرگ مناسب نیست. |
| | TOPSIS[7] (Misaghian et al., 2019) | یک چارچوب تجزیه و تحلیل برای اولویت‌بندی گزینه‌ها بر اساس مجموعه‌ای از معیارها، شامل امتیازدهی به هر گزینه در برابر معیارهایی که می‌توانند در مقیاس مطلق یا نسبی باشند. این الگوریتم بر اساس مدل فضای برداری محاسباتی بوده، که برای انجام اولویت‌بندی سلسله‌مراتبی از نرمال‌سازی ریاضی استفاده می‌کند، و روشی برای انجام تحلیل تصمیم چندمعیاره (MCDA) برای امتیازدهی به مجموعه‌ای از گزینه‌ها در برابر مجموعه‌ای از معیارها می‌باشد، اما در مدیریت پیش‌نیازها ضعیف است. |
| | AHP[8] (J. Karlsson et al., 1998) | این تکنیک معروف ترین و پرکاربردترین تکنیک اولویت‌بندی نیازمندی‌ها بوده و به اولویت‌بندی مجدد در توسعه نرم افزار چابک می‌پردازد. وابستگی‌ها را قبل از اولویت‌بندی شناسایی کرده و از ماتریس مقایسه زوجی برای محاسبه اهمیت نسبی هر نیاز استفاده می‌کند. این روش به دلیل افزایش قابل توجه زمان پردازش در تعداد نیازهای زیاد مقیاس‌پذیر نیست و پیچیدگی $O(n^2)$ را دارد، همچنین استفاده آسانی ندارد، پیچیدگی بالا با سرعت کم، نتایج غیر قابل اعتماد، سطح پایینی در تحمل خطا از مشکلات این روش می‌باشد.<br>در روش Hierarchy AHP نیز الزامات به ترتیب سلسله‌مراتبی بر اساس امتیازات انباشته هر نیاز در بین ذینفعان مربوطه اولویت‌بندی می‌شوند. |

[6] Value oriented prioritization
[7] Technique for ordering from similarity to ideal solution
[8] Analytic hierarchy process

| | | |
|---|---|---|
| هدف این تکنیک به حداقل رساندن افزونگی در تصمیم‌گیری برای پرروژه‌های کوچک تا متوسط است، به ویژه هنگامی که تصمیمات نسبتاً ثابت هستند. چالش اصلی که به آن پرداخته می‌شود، کاهش تعداد مقایسه های مورد نیاز است، و از یک مقیاس وزنی برای رتبه‌بندی الزامات به صورت سلسله مراتبی استفاده می‌کند. در این روش پیچیدگی کم و سهولت استفاده در کنار سرعت قابل قبول را داراست، اما ممکن است نتایج کمتر قابل اعتمادی را ارائه دهد، به ویژه از نظر تحمل خطا. | Minimal spanning tree (J. Karlsson et al., 1998) | |
| در این روش، به ذینفعان یک اسکناس ۱۰۰ دلاری داده می‌شود تا در مورد نیازهای استخراج شده هزینه کنند، پس از آن، کل پول هزینه شده برای هر نیاز بر تعداد کل سهامداران برای اولویت‌بندی نیازمندی‌ها تقسیم می‌شود. از مزایا، به راحتی استفاده با پیچیدگی کم و سرعت بالا در پروژه‌هایی با اندازه کوچک و متوسط است و معایبی همچون سطح اطمینان و تحمل خطای متوسط است. این روش از نظر سرعت اجرا نسبت به AHP برتر است تا زمانی که فرآیند در نظر گرفتن مقادیر فازی نباشد (Sharif et al., 2014). برای حل وابستگی‌های متقابل ناشی از نیازها یک مدل آماری بهبود یافته به نام CoDA[10] ارائه شد(Chatzipetrou et al,2010.) در روش HCV[11] **نیز** اولویت‌بندی در هر سطح سلسله‌مراتبی انجام شده و همه الزامات به طور همزمان اولویت‌بندی نمی‌شوند. روشی آسان و به عنوان راه‌حلی برای مسائل مقیاس‌پذیری با رای‌گیری تجمعی (CV) معرفی شد .(Berander et al., 2006) | CV[9] 100$ (Leffingwell et al., 2000) | |
| این تکنیک از رویکرد یادگیری ماشین، مبتنی بر وب، برای کاهش میزان اطلاعات مورد نیاز از ذینفعان برای دستیابی به رتبه‌بندی با درجه کیفی معین استفاده کرده و فعالیت‌ها را در طول اولویت‌بندی، به منظور ایجاد نظم کاهش می‌دهد. این رویکرد در بدترین سناریو بهتر از AHP عمل می‌کند اما مشکلاتی در مقیاس‌پذیری و هماهنگی بین ذینفعان مختلف دارد. در دسته تکنیک های یادگیری‌ماشین نیز قرار دارد. | CBRank[12](Perini et al., 2012) | |
| این تکنیک از روش بهینه‌سازی تکراری با الگوریتم ژنتیک به منظور رتبه‌بندی نیازمندی‌های نرم‌افزار در چندین تکرار (جایگشت)، برای به حداکثر رساندن سود وزنی نسبت به همه ذینفعان مختلف استفاده کرده و برنامه‌ریزی مداوم توسعه نرم‌افزار را تشویق می‌کند. با در نظر گرفتن وابستگی‌های تجاری در بین الزامات، این روش از نظر دقت نتایج نسبت به AHP و TOPSIS به خوبی عمل کرده اما دارای پیچیدگی محاسباتی و مشکل در ارائه راه‌حل برای مسائل غیرمحدب دارد. این تکنیک جز روش‌های مبتنی بر الگوریتم‌های بهینه‌سازی نیز می‌باشد. | EVOLVE(Thakurta, 2013) | |
| الگوریتم ژنتیک تعاملی برای اولویت‌بندی نیازهای هوشمند، محدودیت‌های وابستگی‌ها و اولویت‌های قبلی را با اطلاعات افزوده ادغام کرده، و از روش استخراج ترجیحات زوجی به عنوان روشی موفق برای به‌دست آوردن داده‌های مرتبط از کاربر استفاده می‌کند. در مقایسه IGA با تکنیک‌های اولویت‌بندی تعاملی پیشرفته و فرآیند تحلیل سلسله مراتبی ناقص بر روی یک سیستم نرم‌افزار واقعی، IGA از دیگر روش‌های اولویت‌بندی تعاملی پیشرفته برتری دارد اما فقط تا حدی خودکار است و برای عملکرد FITNESS FUNCTION به ورودی انسانی نیاز دارد، که آن را مستعد خطا و نیازمند راه‌حل‌های دقیق‌تر می‌کند. | IGA[13] (Tonella et al., 2010) | |
| این روش، مشابه تکنیک انتساب اعداد، نیازمندی‌ها را به گروه‌هایی با اولویت بالا، متوسط و پایین دسته‌بندی می‌کند، اما این کار به طور مکرر انجام می‌شود، یعنی در گروه‌هایی که بیش از یک نیاز دارند، زیرگروه‌ها به صورت بازگشتی تکرار شده، تا زمانی که در هر زیرگروه فقط یک نیاز وجود داشته باشد. این تکنیک، کند و استفاده از آن سخت می‌باشد و از | Priority groups (J. Karlsson et al., 1998) | **مقیاس ترتیبی** |

[9] Cumulative Voting
[10] Compositional Data Analysis
[11] Hierarchical Cumulative Voting
[12] Case based ranking
[13] Interactive Genetic Algorithm

| | |
|---|---|
| نظر سهولت استفاده، قابلیت اطمینان و تحمل خطا کمترین امتیاز را کسب کرده و در مقایسه با سایر تکنیک‌ها (AHP، درخت پوشا حداقل، جستجوی دودویی، مرتب‌سازی حبابی)، به عنوان بدترین رویکرد رتبه‌بندی در نظر گرفته می‌شود. | |
| این تکنیک الزامات را بر اساس مراحل زیر اولویت‌بندی می‌کند: (الف) آماده سازی و ترتیب الزامات در یک بردار، (ب) اجرای مقایسه الزامات و (ج) مرتب سازی الزامات به ترتیب رتبه‌بندی آنها از پایین به بالا. این روش مشابه AHP ، از مقایسه زوجی استفاده کرده، و دارای معایبی همچون پیچیدگی بالا، سرعت کم، غیرقابل اطمینان با تحمل خطای پایین بوده و برای تعداد زیادی از نیازمندی ها مقیاس پذیر نیست. | Bubble sort (J. Karlsson et al., 1998) |
| این تکنیک تمام الزامات استخراج شده را تجزیه و در یک نظم سلسله‌مراتبی رتبه‌بندی می‌کند. با ایجاد یک ساختار درختی که در آن گره‌ها الزامات را نشان می‌دهند، و زیردرخت‌های چپ و راست به ترتیب اولویت‌های پایین‌تر و بالاتر را نشان می‌دهند، تمام الزامات را مشخص کرده، درخت جستجوی دودویی با مقایسه هر نیاز با گره بالایی انتخاب شده و در مرحله ارائه لیست اولویت‌بندی را ایجاد می‌کند. با وجود مقیاس‌پذیر نبودن، دارای استفاده آسان، نتایج قابل اعتماد و سطح بالایی از تحمل خطا با پیچیدگی O(n log n) دارد. | BST[14] (Bebensee et al., 2010; Duan et al., 2009) |
| این تکنیک الزامات را بر اساس مزایای درک شده آنها رتبه‌بندی می‌کند، شبیه به رویکرد درخت جستجوی دودویی (BST) است. این روش در مقایسه با AHP از نظر سهولت استفاده، مصرف زمان، قابلیت اطمینان و تحمل خطا نسبتاً ضعیف بوده و مقیاس‌پذیر نیست. | BPL[15] (J. Karlsson et al., 1998) |
| این تکنیک شامل الگوریتمی است که قادر به جستجوی نیازمندی‌ها در گره‌ها و به حداقل رساندن تعداد مقایسه‌های مورد نیاز برای تعیین اهمیت نسبی با استفاده از مقیاس وزنی به منظور اولویت‌بندی است. نیازمندی‌ها در یک ساختار درختی متعادل که به درخت B معروف است ذخیره و بازیابی می‌شوند، سپس یک درخت باینری بر اساس دنباله تشکیل می‌شود. مکانیسم‌های مرتب‌سازی برای اولویت‌بندی آسان نیازمندی‌ها به ترتیب صعودی یا نزولی، با مقایسه کمتری نسبت به AHPدر زمان کم استفاده می‌شود، اما با توجه به ساختار درخت دودویی، استفاده از آن دشوارتر و به خوبی مقیاس نمی‌شود، همچنین، روش خودکار نیست و کاربر باید ورودی را وارد کند. | B-Tree[16] (Beg et al., 2009) |
| این تکنیک به موقعیتی مربوط می‌شود که در آن مشتریان نیازمندی‌ها را به سه کلاس دسته‌بندی می‌کنند: ضروری، مشروط و اختیاری. این فرآیند بر اساس دو معیار است: ارزش تجاری توسط مشتریان و ریسک فنی که توسط توسعه دهندگان قضاوت می شود. | GP[17] (Duan et al., 2009) |
| این تکنیک دارای فرآیند چهار مرحله‌ای شامل ترجمه نیازهای مشتری، شناسایی روابط بین ویژگی‌های کیفی، اولویت‌بندی فرآیندهای تولید و ایجاد یک برنامه عملیاتی برای تضمین کیفیت است. از ماتریس‌ها برای نشان دادن زمان انتظارات مشتری و نحوه برآورده شدن این انتظارات توسط توسعه‌دهندگان استفاده می‌کند و معمولا برای زیرسیستم‌های کوچک اعمال شده، ناسازگاری را برآورده نمی‌کند و مقیاس‌پذیر نیست. | QFD[18] (Avesani et al., 2005) |
| این تکنیک اولویت‌بندی سه دیدگاه مهم در تعریف محصول را شامل می‌شود: تجارت، کاربران و فناوری | Ranking based on product definition (Racheva et al., 2010) |

[14] Binary search Tree
[15] Binary priority list
[16] Binary Tree
[17] Game planning
[18] quality functional deployment

| | | |
|---|---|---|
| این تکنیک با استفاده از ماتریس رابطه، الزامات را با ترکیب روابط همبستگی اولویت‌بندی می‌کند. | CBPA[19] (Liu et al., 2006) | |
| این تکنیک با تجمیع وزن‌ها با استفاده از الگوریتم استدلال شواهد داخلی (IER) نیازمندی‌ها را اولویت‌بندی می‌کند، اما برای پروژه‌هایی با مقیاس زیاد مناسب نیست. | RUPA[20] (Voola et al., 2012) | **مقیاس فاصله** |

### تکنیک‌های مبتنی بر روش‌های کیفی و کمی

ارائه نظرات در رابطه با اولویت‌دهی به نیازهای جمع‌آوری شده می‌تواند به صورت کیفی و یا کمی ایجاد گردد. ذینفعان داخلی و خارجی شامل: مدیران محصول، صاحبان محصول، طراحان تجربه کاربری، توسعه‌دهندگان نرم‌افزار، آزمایش‌کنندگان نرم‌افزار و مشتریان بر اساس استراتژی حاکم در تیم پیشنهادات خود را بیان می‌کنند. پژوهش (Trieflinger et al., 2021) در این خصوص سه دسته رویکرد را بیان کرده است: روش‌های مبتنی بر نقشه برداری، مبتنی بر امتیازدهی و مبتنی بر بازی.

**جدول-۲: تکنیک‌های مبتنی بر روش‌های کیفی و کمی**

| روش | تکنیک | توضیحات |
|---|---|---|
| **مبتنی بر نقشه برداری** (Pergher et al., 2013) | Story Mapping | هدف از این روش، تمرکز بر تجربه کاربر به جای نظرات داخلی تیم محصول و ذینفعان است. برای این منظور، اولین گام این است که فعالیت‌های کاربر را به ترتیب اجرای آنها توصیف کرده و آنها را در محور افقی فهرست کنیم. سپس وظایف برای هر فعالیت کاربر اضافه می‌شود و به ترتیب اهمیت آنها ( تأثیر تخمینی) مرتب می‌شود . |
| | Systemico Model | این رویکرد بر اولویت‌بندی خروجی‌ها بر اساس دو بعد اهداف کاربر،که مشتری به چه جنبه‌هایی می‌خواهد دست یابد (محور x) و تعامل کاربر، سطح تعامل بین کاربر و محصول را توصیف می‌کند (محور y) ارائه می‌دهد. |
| | Frequency and Volume of Use Model | این مدل به صورت نمودار دو محوره ایجاد می‌شود که در آن ویژگی‌های توسعه‌یافته بر اساس معیارهای «تکرار استفاده» و «تعداد کاربران» طبقه‌بندی می‌شوند. محور x نشان‌دهنده تعداد افرادی است که انتظار می‌رود از یک ویژگی استفاده کنند و محور y توصیف می‌کند که کاربران چقدر احتمال دارد یک ویژگی را به دفعات اعمال کنند؛ در نتیجه، ویژگی‌هایی که توسط همه افراد مورد استفاده قرار می‌گیرند و دارای فراوانی استفاده بسیار بالایی هستند، در اولویت قرار دارند. |
| | Kano Model (Ho et al., 2023) | ایده مدل کانو این است که رضایت مشتری به سطح عملکردی که یک ویژگی ارائه می‌دهد بستگی دارد، یعنی یک ویژگی چقدر خوب پیاده سازی شده است. با جزئیات بیشتر، این مدل یک سیستم هماهنگی شامل دو بعد "رضایت مشتری" و "عملکرد" را ارائه می‌دهد. رضایت مشتری بیان می‌کند که مشتریان چه احساسی نسبت به یک محصول دارند و توضیح عملکرد مشخص می‌کند که یک ویژگی مشخص چقدر خوب اجرا شده است. بر اساس این ابعاد، اولویت‌بندی با طبقه‌بندی ویژگی‌های بالقوه به دسته‌هایی با ترتیبی که در ابتدا ویژگی‌های «باید»، سپس ویژگی‌های «عملکرد» و پس از آن ویژگی‌های «هیجان‌انگیز» اولویت‌بندی می‌شوند. |
| | MosCow | در دسته‌بندی تکنیک‌های مقیاس اسمی بیان شده است. |

[19] Correlation based priority assessment framework
[20] Requirement uncertainty prioritization approach

| | | |
|---|---|---|
| این رویکرد ویژگی‌ها را بر اساس برآورد دو بعد «تاثیر» و «تلاش» اولویت بندی می‌کند. تأثیر به معنای ارزشی است که ویژگی برای مشتری و کسب‌وکار فراهم می‌کند، در حالی که تلاش منابعی را که برای ایجاد ویژگی نیاز دارند اندازه گیری می‌کند. سپس با توجه به افزایش و کاهش هر یک به چهار دسته تقسیم‌بندی می‌شوند: ویژگی‌های که ارزش بالایی دارند، در بالاتریت اولویت، سپس ویژگی‌هایی با ارزش بالا ولی زمانبر، در اولویت پایین‌تر ویژگی‌هایی که به راحتی قابل پیاده سازی هستند، اما تاثیر زیادی بر اهداف مشتری و کسب و کار ندارند، و در پایین‌ترین اولویت ویژگی‌هایی که به تلاش زیادی نیاز دارند و ارزش بسیار کمی ارائه می‌دهند. | Impact vs. Effort | |
| در دسته‌بندی تکنیک‌های مقیاس ترتیبی بیان شده است. | Quality Function Deployment | |
| هدف مدل امتیازدهی ICE اولویت‌بندی محصولات و ویژگی‌ها بر اساس سه عامل "تاثیر"، "اطمینان" و "سهولت" است که در مقیاس نسبی از ۱ تا ۱۰ درجه‌بندی می‌شوند. $ICE - SCORE = Impact * Confidence * Ease$ (فرمول شماره ۱) | ICE | **مبتنی بر امتیازدهی** |
| مدل RICE به مدل ICE شباهت دارد، اما دامنه فاکتور را نیز در نظر می‌گیرد. عوامل "دسترسی"، "تاثیر"، "اطمینان" به عنوان ارزش بالقوه یک محصول یا ویژگی است، در حالی که "تلاش "نشان دهنده هزینه و زمان برای اجرای آن است. RI$CE - SCORE = Impact * Confidence *$ Reach / Effort (فرمول شماره ۲) | RICE | |
| این تکنیک اولویت بندی شامل کارت امتیازی است که هدف آن رتبه‌بندی چندین ویژگی براساس معیارهای مختلف است که به یک نمره کلی محاسبه می‌شود. هر معیار به وزنی اختصاص داده می‌شود که اهمیت این معیارها را در ارتباط با دستیابی به موفقیت محصول بیان می‌کند. در مرحله بعد، هر یک از ویژگی‌های نامزد بر اساس معیارهای تعریف شده با اختصاص امتیاز بین ۱ تا ۱۰۰ مورد ارزیابی قرار می‌گیرد. هر چه امتیاز بالاتر باشد، تأثیری که ویژگی در تحقق معیار مربوطه دارد، بیشتر است. در نهایت امتیاز هر ویژگی در وزن ضرب می‌شود. | Weighted Scoring | |
| ایده این تکنیک این است که ویژگی‌هایی که ارزش بالاتری دارند، اما زمان تحویل کوتاه‌تر باید نسبت به ویژگی‌هایی که بیشتر طول می‌کشند و ارزش کمتری ارائه می‌کنند، اولویت داشته باشند. برای دستیابی به این هدف، ویژگی‌های بالقوه با تقسیم هزینه تاخیر بر طول مدت کار رتبه‌بندی می شوند. | Weighted Shortest Job First | |
| امتیازدهی فرصت بر اساس بازخورد مشتری است و هدف آن شناسایی ویژگی‌هایی است که مشتریان آن را ضروری می‌دانند اما از آن ناراضی هستند. این تکنیک از دو بعد «رضایت» و «اهمیت» برای اندازه‌گیری و رتبه‌بندی نتایج یا ویژگی‌ها استفاده می‌کند. فرآیند امتیازدهی فرصت با ایجاد فهرستی از نتایج یا ویژگی های مورد نظر برای یک محصول خاص آغاز می‌شود. سپس، هر نتیجه یا ویژگی توسط مشتریان در مقیاس ۱ تا ۱۰ نمره‌گذاری می‌شود. نتایج با بالاترین امتیاز اهمیت و کمترین رضایت، بالاترین اولویت را نشان خواهد داد. | Opportunity Scoring | |
| این مدل یک بازی نوآوری است که در آن اعضای داخلی تیم، مشتریان و سهامداران دعوت می‌شوند تا در آن شرکت کنند. اولین گام فهرست کردن تمام ویژگی‌های بالقوه و تعیین قیمت برای هر کدام است. برای ویژگی‌هایی که زمان‌بر یا حیاتی هستند قیمت بالا توصیه می‌شود که شرکت‌کنندگان پول خود را برای خرید آن ویژگی تجمیع کنند و این امر منجر به تولید لیست اولویت‌بندی شده از ویژگی‌ها می‌شود. | Buy a feature | **مبتنی بر بازی** (L. Karlsson et al., 2007) |
| این روش بر جنبه معکوس اولویت‌بندی، یعنی شناسایی ویژگی‌های کم اولویت تمرکز دارد. هر ویژگی با نماد لنگری برای قایق و رتبه‌بندی سرعت، معیاری را نشان می‌دهد که طبق نظر کاربران قایق با چه سرعتی حرکت | Speed Boat | |

| | |
|---|---|
| | کند که در مسیر موفقیت محصول قرار گیرد. این امر به اولویت‌بندی عقب‌ماندگی‌ها با توجه به نیازها و نظرات مشتریان کمک می‌کند. |
| Ian McAllister Framework | این چارچوب توسط یکی از مدیران سابق آمازون پیشنهاد شد و شامل مراحلی مانند ایجاد فهرستی از موضوعات مهم، تعریف اولویت نسبی و تخصیص منابع آنها، تولید ایده های پروژه، برآورد تأثیر و هزینه، و انتخاب پروژه‌هایی با بیشترین تأثیر و کمترین هزینه است. این چارچوب به جای فهرستی از ویژگی‌ها، به ایجاد یک طرح کمک می‌کند و نیاز به اولویت‌بندی پروژه‌های مختلف در برابر یکدیگر را از بین می‌برد. این امر بر اهمیت در نظر گرفتن تأثیر پروژه ها بر موضوعاتی مانند جذب مشتری، تعامل و فعال‌سازی تأکید می‌کند و تقسیم پروژه‌های بزرگتر به بخش های کوچکتر را تشویق می‌کند. |
| Feature Buckets | این روش ویژگی‌ها را به چهار بخش دسته‌بندی می‌کند: محرک‌های متریک (ویژگی‌هایی که به طور قابل توجهی به موفقیت محصول کمک می‌کند)، درخواست‌های مشتری، لذت (ویژگی‌های نوآورانه‌ای برای خوشحال کردن مشتریان و ایجاد موقعیت متمایز در بازار) و استراتژیک (ویژگی هایی که به دلایل استراتژیک گنجانده شده‌اند). برای موفقیت، مهم است که ویژگی‌های سه بخش اول را در یک نسخه اصلی محصول قرار دهید. |
| Assumption Mapping | شناسایی مهم‌ترین ریسک‌های یک مدل کسب‌وکار را به منظور افزایش شانس موفقیت یک ایده تجاری امکان‌پذیر می‌کند. فرآیند ایجاد یک نقشه فرضی با جمع‌آوری تمام مفروضات مهم زیربنای یک مدل کسب‌وکار شروع می‌شود که ایا مشتریان نیاز بیان شده را میخواهند؟ ایا این نیاز امکان‌پذیر است و قابلیت اجرا دارد؟ |
| The Hypotheses Prioritisation Canvas | این رویکرد از دو بعد «ارزش ادراک‌شده» و «ریسک» برای طبقه‌بندی و مقایسه مفروضات مختلف استفاده می‌کند. این مفروضات ادعاهایی در یک طرح تجاری تلقی می‌شوند که تأثیر قابل‌توجهی بر موفقیت یا شکست آن دارند. آزمایش این مفروضات می‌تواند چالش برانگیز باشد زیرا مشتریان اغلب فکر می‌کنند که می‌دانند چه می‌خواهند، اما معلوم می‌شود که اشتباه می‌کنند. |

**تکنیک‌های مبتنی بر منطق فازی**

یک روش منطق فازی موثر باید نسبت سازگاری بین ۰ و ۰.۱ را حفظ کرده، تابع عضویت دائمی مثبت را تضمین کند و با کاهش تعارض بین الزامات، سازگاری ایجاد کند. چارچوب منطق فازی مقیاس پذیر است، کارایی را افزایش می‌دهد و زمان که جزء ضروری در هر پروژه‌ای است و باید به خوبی مدیریت شود را در محاسبات کاهش می‌دهد. به طور کلی، یک محدودیت زمانی، به شناسایی الزامات در مدت زمان کوتاه‌تری کمک می‌کند. سیستم منطق فازی راه‌حل‌های مقرون به صرفه‌ای را با در نظر گرفتن فاکتور هزینه در حین اولویت‌بندی ارائه می‌دهد و پیچیدگی آن $O(n)$ است که از روش‌های سنتی مانند AHP با پیچیدگی $O(n^2)$ بهتر عمل می‌کند، از طرفی با استفاده از ریاضیات برای مقابله با مسائل غیردقیق و زبانی در جهت کمی کردن نیازها استفاده می‌کند ولی لازم به ذکر است مجموعه راه‌حل‌های بهینه کمتری دارد. (Singh et al., 2021)

تکنیک‌های منطق فازی نیاز به تولید نتایج اولویت‌بندی دقیق‌تری دارند. از آنجایی که داده‌ها اغلب نادرست هستند، استنتاج فازی باید داده‌ها را برای یافتن مقادیر صحیح محاسبه کند. با جدا کردن متغیرهای قابل اعتماد، وابستگی بین متغیرها در سیستم را می‌توان از طریق استنتاج فازی مدیریت کرد. در نتیجه، منطق فازی فرآیند سفارش‌دهی را با درست‌تر و قابل اطمینان‌تر کردن آن بهبود می‌بخشد (Bisht et al., 2020).

**جدول-۳: تکنیک‌های مبتنی بر منطق فازی**

| تکنیک | توضیحات |
|---|---|
| FCM[21] (Ramzan et al., 2011) | یک تکنیک چند سطحی مبتنی بر روش میانگین C برای تخصیص نیازمندی‌ها با استفاده از تابع عضویت فازی می‌باشد، که در آن نیازمندی‌های نزدیک مرکز دارای مقادیر عضویت بالا و دور از مرکز دارای مقادیر عضویت پایین هستند، این تکنیک به دلیل مشارکت ذینفعان و متخصصان گران است. |
| neuro-fuzzy system (Momeni et al., 2014) | یک سیستم عصبی فازی که توالی ایده‌آل ویژگی‌های کیفی را برای رضایت مصرف‌کننده، از جمله صحت، قابلیت همکاری، امنیت، قابلیت درک، قابلیت ردیابی، سرعت و زمان تکمیل کار را تعیین می‌کند و محاسبات شبکه و سرعت تولید خروجی در این روش بالاتر از روش QFD می‌باشد. |
| fuzzy multi-attribute decision (Ejnioui et al., 2012) | این روش الزامات را به عنوان یک مسئله تصمیم‌گیری چند معیاره فازی که در آن از عملگر مقدار پیش‌بینی شده برای سفارش گزینه‌ها در طرح موضوع استفاده می‌شود. در این رویکرد با نشان دادن ویژگی‌های کیفی متمایز به عنوان اعداد خاکستری، با داده‌های نادرست مقابله شده و برای بررسی سناریوهای what-if با وزن‌های مختلف برای ویژگی‌های مختلف استفاده می‌شود. روشی ساده، انعطاف‌پذیر، کم هزینه و دارای کارایی بالا در زمان است. |
| a-level weighted F-preference relation+ fuzzy AHP+ binary sort tree (Sadiq et al., 2014) | روش مبتنی بر فازی که سه روش رابطه ترجیحی F وزن‌دار سطح الف، AHP فازی و روش درخت مرتب‌سازی باینری را برای اولویت‌بندی نیازمندی‌ها در فرآیند PRFGORE برای مقایسه‌های زوجی نیازمندی‌های عملکردی و غیرعملکردی ادغام می‌کند و به موضوع سازمان‌دهی نیازمندی‌ها در زمانی که ترجیحات ذینفعان بیان نمی‌شود، می‌پردازد. |
| Fuzzy Hierarchical Cumulative Voting approach (Jawale et al., 2015) | استفاده از رأی‌گیری تجمعی سلسله‌مراتبی فازی تطبیقی برای اولویت‌بندی نیازمندی‌ها و رسیدگی به رفتار پیچیده پروژه، بر پیچیدگی، ابهام و عدم قطعیت غلبه کرده و منجر به بهبود تصمیم‌گیری می‌شود. در این روش هر شیء در بالاترین سطح انتزاع به سطوح خاص‌تری تجزیه شده و از تکنیک نرمال‌سازی برای تعیین اولویت‌های نهایی استفاده می‌شود، ماژول انطباق نتایج الزامات اولویت‌بندی شده را نظارت و تجزیه و تحلیل می‌کند که آیا دقیق است یا خیر. اگر الزامات به درستی اولویت‌بندی نشده باشند، فازی HCV دوباره اعمال می‌شود. در نهایت، با استفاده از روش فازی‌سازی، اولویت‌ها غیرفازی شده و اولویت‌های نهایی را به دست می‌آورند. |
| alpha cut technique (Dabbagh et al., 2015) | تکنیک برش آلفا برای اولویت‌بندی نیازمندی‌های غیرعملکردی (NFR) برای ایجاد یک ماتریس تصمیم mxn استفاده و درجه ارتباط هر الزام غیرعملکردی در مورد هر الزام عملکردی تعیین می‌شود. در شرایط دشوار و گران روی حیاتی‌ترین نیازمندی‌های غیرعملکردی به عنوان محرک اصلی در طراحی معماری نرم‌افزار تمرکز شده و با انتخاب دستورالعمل‌های مناسب ویژگی‌های کیفی مورد نیاز یک سیستم نرم‌افزاری ساده می‌شود. |
| Phandler[22] (Babar et al., 2015) | یک سیستم خبره اولویت‌بندی نیازمندی‌های هوشمند مبتنی بر ارزش، که از شبکه عصبی برای پیش‌بینی ارزش و از روش سلسله‌مراتبی تحلیلی برای مقیاس‌پذیر کردن فرآیند اولویت‌بندی استفاده می‌کند. |
| multi-level quality-based (Mishra et al., 2016) | در این روش پنج عامل ورودی شامل هزینه، طراحی، عملکرد، زمان پاسخگویی و تعداد ذینفعان و همچنین دو متغیر خروجی، کامل بودن و قابل درک بودن، ایجاد می‌شود که طبقه‌بندی خودکار نیازمندی‌های اولویت‌دار به دسته‌های مهم، ضروری و جانبی تقسیم‌بندی شوند. |
| quantifying conflicts among requirements (Gulzar et al., 2017) | تعیین کمیت موثر تضادها بین الزامات ابزار میدانی برای مقایسه کارایی و یادگیری توابع اعضا می‌باشد که با تجزیه و تحلیل آماری میان مدیران پروژه، مهندسان نرم‌افزار، مهندسین نیازمندی‌ها، رهبران تیم و تحلیلگران نیازمندی‌ها که تیم را تشکیل می‌دهند، بازخوردها ثبت و یافته‌های نهایی نمودار می‌شود. |

[21] Fuzzy C mean
[22] Priority Handler

| | |
|---|---|
| PAPS (Mougouei et al., 2019) | یک سیستم فازی به نام PAPS برای بهبود دقت اولویت‌بندی، انتخاب و پرداختن به الزامات امنیتی است. این روش در دو مرحله پیش‌اولویت‌بندی و انتخاب (Pre-PAS) صورت گرفته که نشان‌دهنده، نمایش و بررسی الزامات امنیتی بوده و از زبان کنترل فازی (FCL) برای ایجاد قوانین استنتاج فازی استفاده می‌کند. پیچیدگی این روش O(n) است که نسبت به مرتب‌سازی حبابی $O(n^2)$ و درخت جستجوی باینری O(nlogn) مناسب‌تر است. |
| VRPR (Sadia et al., n.d.) | رویکرد اولویت‌بندی نیازمندی‌های جزئی مبتنی بر منطق فازی در این روش توسعه یافته است. فازی کردن ورودی‌ها، پیاده‌سازی عملگرهای فازی، بکارگیری روش‌های مفهومی، تجمیع خروجی‌ها و فازی‌زدایی نتایج، پنج فرآیند در سیستم استنتاج فازی هستند. رتبه‌بندی اولویت نیازمندی جزئی تنها متغیر زبانی در مرحله خروجی است که با استفاده از توابع عضویت مناسب، متغیرهای ورودی و خروجی به مجموعه‌های فازی ترجمه می‌شود. رتبه‌بندی اولویت نیاز جزئی نتیجه الگوریتم استنتاج فازی که هر چه امتیاز VRPR بیشتر باشد، نیاز جزئی مهمتر بوده است. |
| LFPP + ANN[23] (Singh et al., 2019) | در این روش، یک استراتژی ترکیبی شامل برنامه‌ریزی ترجیحی فازی لگاریتمی (LFPP) و شبکه‌های عصبی مصنوعی (ANN) را برای اولویت‌بندی نیازهای مشتری بر اساس عوامل متعدد با هزینه معقول ترکیب می‌کند. بخشی از اطلاعات توسط معیارهای متعدد وارد شده و سپس در MATLAB وزن اولویت‌های جایگزین تعیین می‌شود، علاوه بر این، شبکه عصبی ساخته شده مناسب بودن یا نامناسب بودن اقدامات تصمیم گیرنده (DM[24]) را ارزیابی می‌کند. |
| VBRP[25] (Kukreja et al., 2013) | این روش انتخاب با ارزش‌ترین الزامات برای پیاده‌سازی برای اطمینان از تحویل یک سیستم نرم‌افزاری با ارزش بالا است. در این رویکرد به معیارهای مختلف که شامل عواملی مانند سهولت استفاده، انعطاف‌پذیری، مقیاس‌پذیری، دقت و همسویی با اهداف تجاری است وزن‌دهی صورت گرفته و رتبه‌بندی می‌شوند. این فرآیند تسهیل‌کننده در جهت اولویت‌بندی از دیدگاه ذینفعان و متخصصان ارائه شده است. |

### تکنیک‌های مبتنی بر الگوریتم‌های بهینه‌سازی

بیشتر تکنیک‌های بهینه‌سازی برای نیازمندی‌های پروژه‌های کوچک طراحی شده‌اند، که با افزایش تعداد نیازمندی‌ها، سؤالاتی در مورد اثربخشی آنها ایجاد می‌شود. تکنیک‌های بهینه‌سازی بر أساس الگوریتم می‌توانند با به حداقل رساندن تعداد مقایسه‌ها یا انتخاب جمعیت تصادفی، برخلاف تکنیک‌های مرسوم، به کاهش مصرف زمان کمک کنند و همچنین نتایج دقیقی را ایجاد می‌کنند. انواع الگوریتم‌های بهینه‌سازی برای اولویت‌بندی نیازمندی‌های نرم‌افزار استفاده می‌شوند برای مثال تابع FITNESS FUNCTION بهینه‌سازی بهبودیافته و همگرایی سریع به راه‌حل بهینه را تضمین کرده تا به هدف اصلی تکنیک‌های بهینه‌سازی بر أساس الگوریتم، یعنی خودکار کردن فرآیندها و به حداقل رساندن تلاش انسانی، فراهم گردد. محدودیت‌های این روش شامل کارایی کم و مقیاس‌پذیری ضعیف می‌باشد (Durillo et al., 2009).

**جدول-۴: تکنیک‌های مبتنی بر الگوریتم‌های بهینه‌سازی**

| تکنیک | توضیحات |
|---|---|
| Exact, Greedy, and Local Search | تکنیک جستجوی دقیق، حریصانه و محلی را برای حل مشکل انتشار بعدی (NRP) توسعه یافته و از یک رویکرد دقیق و یک روش شاخه و کران عمومی برای استخراج حدبالا و یک مدل برنامه‌نویسی عدد صحیح کامل برای حدپایین استفاده کرده است. |

[23] logarithmic fuzzy preference programming and artificial neural networks
[24] decision-maker
[25] Value-Based Requirements Prioritization

| | |
|---|---|
| (Bagnall et al., 2001) | این مدل مجموعه‌ای از الگوریتم‌های حریصانه را برای تولید سریع محدودیت‌های پایین‌تر ایجاد کرده و از ساختار همسایگی پایه برای اعمال مجموعه‌ای از استراتژی‌های جستجوی محلی مرسوم، مانند الگوریتم hill climber و الگوریتم Simulated annealing، برای ارائه راه‌حلی با کیفیت بالا و تقریباً بهینه در مدت زمان کوتاه استفاده می‌کند و از دو الگوریتم بیان شده نتیجه بهتری دارد. |
| DDP[26] (Feather et al., 2002) | ترکیب Simulated annealing و یک مدل تکراری برای انتخاب نیازمندی‌ها و مسئله بهینه‌سازی این روش مقیاس‌پذیر را توسعه داده است. برای نمونه‌برداری به طور تصادفی از فضای حل ، از یک مدل تعاملی استفاده می‌شود که با استفاده از یک ابزار مورد تجزیه و تحلیل قرار گرفته و با ایجاد مکانیسمی برای همگرایی به راه‌حل‌های نزدیک بهینه منجر به نتیجه مطلوب برای ارزیابی، برنامه‌ریزی و مدیریت ریسک می‌شود. |
| NSGA[27] (Deb et al., 2002) | الگوریتم ژنتیک مرتب‌سازی غیرمسلط به همراه MOEA، برای ایجاد مجموعه‌ای از بهترین راه‌حل‌ها توسعه یافته و از روش crowded-comparison به منظور کاهش پیچیدگی محاسباتی استفاده می‌کند. |
| Component Selection Problem (Harman et al., 2006) | مسئله انتخاب مؤلفه یک مسئله بهینه‌سازی است که به مدیران کمک می‌کند تا مؤلفه‌ها را اولویت‌بندی کنند و انتخاب‌های منطقی محصول را تعیین کنند، که اغلب به بهینه‌سازی توابع هدف چندگانه نیاز دارد و می‌تواند به عنوان یک مسئله کوله‌پشتی ۰-۱ بهینه‌سازی با محدودیت هزینه معین در نظر گرفته شود، که پیشنهاد استفاده از روش‌های اکتشافی مبتنی بر جستجو مانند منحنی Pareto را دارا است. |
| greedy and simulated annealing techniques (Baker et al., 2006) | رویکردی خودکار در مهندسی نیازمندی‌ها را برای رتبه‌بندی نیازمندی‌های نرم‌افزار با استفاده از تکنیک‌های حریصانه و شبیه‌سازی شده با فرموله کردن مسئله به دنباله‌ای از انتخاب زیرمجموعه ویژگی‌ها، ارائه می‌کند. الگوریتم simulated annealing به طور مداوم نتایج بهتری اما با زمان بیشتر در اجراهای متعدد و تنظیمات پارامتر ایجاد می‌کند. |
| bi-objective optimization problem (Saliu et al., 2007) | ابزاری برای تصمیم‌گیری برنامه‌ریزی انتشار سیستم‌های نرم‌افزاری که هم عوامل تجاری و هم عوامل اجرایی را در نظر می‌گیرد. این ابزار مسئله را به عنوان یک مسئله بهینه‌سازی دو هدفه با مبادله بین دو دیدگاه فرموله می‌کند و با راه‌حل‌های بهینه Pareto طرح‌های انتشار متناوب را ارائه می‌دهد. روش پیشنهادی برای شناسایی SD-coupling در سیستم‌های نرم‌افزاری مقیاس‌پذیر و مستقل از دانه‌بندی مولفه component granularity است. با این حال، اجزاء باید در یک سطح دانه‌بندی در همه پروژه‌ها توصیف شوند. تعامل نیازمندی‌ها، بر مطالعه و مدیریت وابستگی‌های متقابل بین نیازمندی‌ها تمرکز دارد. |
| MONRP[28] (Zhang et al., 2007) | استفاده از الگوریتم‌های تکاملی بهینه وزن‌دار و پارتو، برای حل مسئله انتشار بعدی چند هدفه در مهندسی نیازمندی‌ها می‌باشد. در نتیجه با استفاده از الگوریتم ژنتیک مرتب‌سازی غیرمسلط (NSGA-II) به‌عنوان راه‌حل مناسب برای MONRP به نگرانی‌های مقیاس‌پذیری می‌پردازد و الزامات نسخه‌های آینده را طبقه‌بندی می‌کند. |
| ACO[29])de Souza et al., 2011) | تکنیک بهینه‌سازی کلونی مورچه‌ها برای حل مشکل انتشار بعدی با نیازهای وابسته استفاده می‌شود. این روش در مقایسه با simulated annealing و evolutionary algorithms راه‌حل‌های دقیق‌تری ارائه می‌دهد و در همه موقعیت‌ها از الگوریتم ژنتیک بهتر عمل می‌کند. از آزمون Wilcoxon Rank Sum برای ارزیابی معنای آماری استفاده می‌شود. |

[26] Defect Detection and Prevention
[27] non-dominated sorting genetic algorithm
[28] Multi-Objective Next Release Problem
[29] Ant Colony Optimization

| | |
|---|---|
| در انتخاب نیازمندی‌های نرم‌افزار، الگوریتم تکامل تفاضلی (DE) در مقایسه با تکنیک‌های دیگر و از جمله NSGA-II اثربخشی مناسبی را داشته و به طور مداوم نتایج با کیفیت بالا را برای مشکل انتخاب نیازهای نرم‌افزار ارائه می‌دهد. | DE[30] (Chaves-González et al., 2015) |
| با استفاده از الگوریتم ژنتیک غنی شده (EGA) مدلی توسعه می‌یابد که اولویت‌بندی و زمان‌بندی الزامات را برای به حداقل رساندن هزینه پروژه و زمان توسعه ترکیب کرده و مدل برنامه‌ریزی خطی عدد صحیح اصلاح شده اکتشافی (RILP) که از دو معیار وابستگی‌های مورد نیاز و دوره پروژه استفاده می‌کند، تولید می‌شود. از تکنیک حداکثرسازی انتظارات (EM) برای محدود کردن کارآمد تعداد زیرمجموعه‌های مورد نیاز استفاده می‌شود. مدل Hybrid EGRILP از نظر اولویت‌بندی و زمان‌بندی نیازمندی‌های نرم‌افزار از سایر الگوریتم‌ها بهتر عمل می‌کند و در نتیجه برنامه‌ریزی انتشار نرم‌افزار کارآمدتر با حداقل تاخیر و حداکثر سود را به همراه دارد. | RILP[31] (Valsala et al., 2016) |
| این روش مشکل انتخاب نیازهای نرم‌افزار را با استفاده از سه الگوریتم الهام گرفته از کوانتوم با نام‌های QEMEA، QMDEA و MQHDE ارزیابی می‌کند. الگوریتم‌ها به همراه NSGA-II و سایر تکنیک‌ها با استفاده از معیارهای عملکردی مانند :حجم زیاد، گسترش، فاصله نسلی، ظرفیت و راه‌حل‌های بهینه پارتو و آزمون‌های آماری مانند: فرضیه ناپارامتری KruskalWallis مقایسه می شوند. | quantum algorithms (Kumari et al., 2016) |
| الگوریتم ژنتیک مرتب‌سازی غیرمسلط با رویکرد پارتو برای بهینه‌سازی چند هدفه در رتبه‌بندی نیازمندی‌های نرم‌افزار پیشنهاد شده که ترجیحات مشتریان و معیارهای هزینه را برای یافتن مجموعه راه‌حل با کیفیت بالا در یک محدودیت هزینه پیاده‌سازی خاص تنظیم می‌کند. تکنیک پیشنهادی، NSGA-IIPT، با نتایج مطالعات قبلیDEPT ، ACO، NSGA-II و GRASP مقایسه شده و مشخص شده است که در کاوش فضای جستجو از آنها بهتر عمل می‌کند. | NSGA-IIPT (Marghny et al., 2017) |
| رویکرد Jaya چند هدفه (MO-Jaya) ، یک تکنیک بهینه‌سازی چند هدفه پسینی است که می‌تواند چندین هدف را به طور همزمان مدیریت کند و بسیاری از راه‌حل‌های بهینه را در یک جلسه شبیه‌سازی ایجاد کند. با توجه به شبیه سازی‌ها و تحقیقات قبلی، الگوریتم MO-Jaya از سایر تکنیک‌ها مانندNSGA ، GA، جستجوی تکراری و BBO بهتر عمل می‌کند. یک کد کامپیوتری برای روش MO-Jaya در MATLAB R2009a ایجاد شده است. | MO- Jaya[32](Rao et al., 2017) |
| یک استراتژی جدید با استفاده از الگوریتم ژنتیک تصادفی مبتنی بر حداقل مربعات برای بهبود عملکرد مهندسان در اولویت‌بندی نیازمندی‌ها، صرفه‌جویی در زمان و تلاش، و تولید نتایج دقیق می‌باشد. | least-squares-based random genetic algorithm (Ahuja et al., 2018) |
| روشی موازی مبتنی بر مفهوم اصلی-ثانویه با استفاده از فعالیت گونه خاصی از زنبور عسل شامل: زنبورهای شاغل، تماشاگر و پیشاهنگ در سه مرحله پیشنهاد می‌کند. زنبورهای شاغل به دنبال منابع غذایی جدید در منطقه اطراف می‌گردند و در صورت یافتن مکان غذایی به کندو برگشته و زنبورهای تماشاگر را مطلع می‌کنند. به دنبال اطلاعات به دست آمده، هر زنبور ناظر در کندو به طور تصادفی یک منبع غذایی را برای انجام جستجوهای اضافی انتخاب کرده و هر چه عرضه غذا بیشتر باشد، زنبورهای ناظر بیشتر آن را انتخاب خواهند کرد. زنبورهای پیشاهنگ زنبورهای اجیر شده‌ای هستند که قبلاً منابع غذایی خود را رها کرده و در تعقیب یک منبع غذایی جدید، به طور تصادفی در منطقه حرکت می‌کنند. این الگوریتم کیفیت راه‌حل‌ها را بهبود می‌بخشد و با افزایش کیفیت خروجی، زمان اجرا را در مقایسه با روش‌های دیگر مانند NSGA II، GRASP و ACS کاهش می دهد و در MATLAB نسخه R2014b پیاده سازی شده است. | parallel method based on the main–secondary (Alrezaamiri et al., 2020) |

[30] differential evolution
[31] Revamped Integer Linear Programming
[32] multi-objective Jaya

**تکنیک‌های مبتنی بر یادگیری ماشین**

تکنیک‌های مبتنی بر یادگیری ماشین راه‌حل‌های مقیاس‌پذیری را ارائه می‌دهند. آنها می‌توانند تعداد زیادی از نیازهای نرم‌افزاری را اولویت‌بندی کرده و در سناریوهای مختلف مورد استفاده قرار دهند. تمام تکنیک‌ها را می‌توان به راحتی گسترش داد و با حذف گروه‌های تکراری نیازمندی‌ها بر مشکلات افزونگی غلبه کرد که منجر به کارایی از نظر هزینه و زمان می‌شود. ابزارهای استفاده شده در این تکنیک‌ها شامل: Score ابزاری مبتنی بر وب با رویکرد اولویت‌بندی سه مرحله‌ای، آزمون Mann-Whitney-Wilcoxon ، آزمون Shapiro-Wilk W، آزمون ANNOVA ، میانگین حسابی و انحراف معیار، ابزار SRPTackle و ابزار پرکاربرد Python می‌باشند.

در تکنیک‌های مبتنی بر یادگیری ماشین به معیار طبقه‌بندی برای رتبه‌بندی نیازها باید توجه کرد که متعلق به کلاس باینری، کلاس چندگانه یا رگرسیون هستند، از طرفی چگونگی ارتباط متغیرهای ورودی با یکدیگر در مدل طبقه‌بندی، تشخیص دقیق کلاس متغیرهای ورودی را تسهیل می‌کند. هنگامی که اولویت‌بندی نرم‌افزار از طریق یادگیری ماشین انجام می‌شود، انتخاب ویژگی بسیار مهم است و باید با دقت انجام شود زیرا مجموعه ویژگی‌ها به طبقه‌بندی نیازمندی‌ها بر اساس اهمیت و محدودیت‌ها کمک می‌کند.

در حالی که تکنیک‌های یادگیری ماشینی کمتر دستی هستند و بنابراین کمتر مستعد خطاهای انسانی بوده و هنوز جایی برای بهبود دقت نتایج وجود دارد. برخی از تکنیک‌ها، تا حدی خودکار هستند ولی احتمال خطا را افزایش می‌دهند (Anwar et al., 2023)

**جدول-۵: تکنیک‌های مبتنی بر یادگیری ماشین**

| تکنیک | توضیحات |
|---|---|
| comparative analysis of two options (Avesani et al., 2003) | یک فرآیند استخراج مبتنی بر مورد برای برنامه‌ریزی ارزیابی رتبه، با استفاده از روش زوجی برای استخراج ترجیحات مبتنی بر تحلیل مقایسه‌ای می‌باشد که در دو مرحله توسعه یافته: توسعه استراتژی انتخاب مورد و تقریب ترجیحات مورد، با هدف آسان‌تر کردن مقایسه زوجی و کاهش زمان استخراج. نتایج تجربی بین دقت رتبه و تلاش استخراج را نشان می‌دهد. |
| case selection method and approximating case (Avesani et al., 2005) | تکنیک استخراج اولویت جهت حل مشکل مقیاس‌پذیری بر اساس بررسی دو فرآیند خودکار برای توسعه روش انتخاب مورد و تقریب ترجیحات مورد با استفاده از رویکرد زوجی است. نتایج تجربی یک مبادله موفقیت‌آمیز بین دقت اولویت پیش‌بینی شده و تلاش کاربر را نشان می‌دهد. |
| SPK[33] (Duan et al., 2009) | تکنیکی برای اولویت‌بندی نیازمندی‌ها بر اساس اهداف تجاری، علایق و مسائل بین‌بخشی مانند الزامات امنیتی و عملکرد ذینفعان می‌باشد. محدودیت‌های این روش مستقیماً با محدودیت‌های الگوریتم‌های طبقه‌بندی، ردیابی و خوشه‌بندی مرتبط است که همگی مبتنی بر داده‌کاوی احتمالی و تکنیک‌های بازیابی اطلاعات هستند و از این رو نمی‌توانند دقت مناسبی داشته باشند. |

[33] Spherical Kmeans

| | |
|---|---|
| Clustering/evolutionary-based (Achimugu et al., 2015) | الگوریتمی ترکیبی از وزن‌های ترجیحی نیاز که از ارزیابی‌های ذینفعان برای اولویت‌بندی نیازمندی‌ها استفاده می‌کند. هدف آن پرداختن به مقیاس‌پذیری، تغییر رتبه و پیچیدگی محاسباتی است. برای خوشه‌بندی نیازمندی‌ها و ارزیابی اهمیت نسبی آنها، از یک تکنیک ترکیبی مبتنی بر تکامل تفاضلی و الگوریتم k-means استفاده می‌شود و به توسعه‌دهندگان نرم‌افزار در برنامه‌ریزی انتشار و کاهش تضادها کمک می‌کند. از مجموعه داده‌های معیار RALIC ، که شامل الزامات با وزن نسبی سهامداران است، برای تأیید این تکنیک استفاده می‌شود. |
| DRANK (Shao et al., 2017) | تکنیکی اولویت‌بندی که از درخت ویژگی‌های ارزیابی اولویت‌بندی و RankBoost برای محاسبه اولویت‌بندی نیازهای ذهنی بر اساس ترجیحات ذینفعان استفاده می‌کند. این روش از نظر دقت و قابلیت اطمینان از روش‌های CBRank، EVOLVE و AHP بهتر عمل می‌کند و با در نظر گرفتن وابستگی‌های نیاز، قابلیت اعتماد، دقت و زمان کمتر، اولویت نهایی را بهبود می‌بخشد. ابزارهای مختلفی مانند TAOM4E، OpenOME، RE-Tools و GR-Tool کمک کننده هستند. |
| CDBR (Gupta et al., 2022) | رویکرد اولویت‌بندی نیازمندی‌های مشارکتی مبتنی بر وابستگی نیمه خودکار (CDBR) به وسیله الگوریتم یادگیری ماشین برای کاهش اختلافات و بهبود تقریب بر مقیاس‌پذیری، دقت، وابستگی‌های نیازمندی‌ها و ارتباطات ذینفعان و توسعه‌دهنده تمرکز دارد. این روش از تکنیک بهینه‌سازی ازدحام ذرات PSO[34] برای کاهش اختلاف نظر بین ذینفعان و رتبه‌بندی توسعه‌دهندگان استفاده می‌کند. اولویت توسعه‌دهنده با استفاده از یک ماتریس وابستگی محاسبه می‌شود که به طور تصادفی ایجاد شده و در عین حال چگالی ماتریس را در نظر می‌گیرد. هر چه چگالی بیشتر، وابستگی‌های سیستم بیشتر و از این رو پیچیده‌تر است. دقت یافته‌ها با مقایسه تفاوت در عدم توافق بین فهرست‌های اولویت CDBR-AHP و CDBR-IGA با استفاده از تکنیک آزمون آنالیز واریانس (ANOVA) تعیین می‌شود. کسری اختلاف بین CDBR-AHP و CDBRIGA صحت یافته‌ها را تعیین می‌کند. از نظر کارایی و زمان پردازش از روش‌های AHP و IGA بهتر است. |
| SRPTackle (Hujainah et al., 2021) | این روش، از تکنیک WSM برای فرمول‌بندی RPV، الگوریتم‌های خوشه‌بندی (K-means و K-means++) و BST استفاده می‌کند و مسائل مربوط به تکنیک‌های اولویت‌بندی نیازمندی‌های موجود مانند مصرف زمان، مقیاس‌پذیری، نتایج دقیق‌تر در زمان کمتر نسبت به سایر رویکردها و جلوگیری از اتکای بیش از حد به مداخله متخصص و حل محدودیت‌های تعریف شده، می‌پردازد. از مجموعه داده‌های معیار RALIC برای ارزیابی استفاده شده است. |
| gradient-boosted tree (Bollumpally et al., 2019) | این تکنیک از روش XGBoost استفاده می‌کند که یک پیاده‌سازی درختی با گرادیان تقویت شده برای کاهش بیش‌برازش و حفظ عملکرد بالا است. اولین تکنیک (A) شامل انجام طبقه‌بندی باینری بر اساس ROC-AUC در کار و تخصیص امتیاز اولویت به هر فرآیند بر اساس حداکثر احتمال پیش‌بینی است. تکنیک جایگزین (B) مستلزم جمع‌آوری ویژگی‌ها پس از پردازش آنها برای ایجاد ارتباط یک به یک بین ویژگی‌ها و فرآیندها است. مدل انتخاب شده احتمال تکمیل وظایف را با استفاده از تقویت درخت تصمیم‌گیری گرادیان با خروجی لجستیک پیش بینی می‌کند. |

[34] Particle Swarm Optimization

## نتیجه‌گیری

در این پژوهش مروری بر روش‌های اولویت‌بندی نیازمندی‌های نرم‌افزار انجام شد که روش‌های پیشنهادی و روش‌های بهبود یافته را به صورت دسته‌بندی شده از سال ۲۰۱۴ تا ۲۰۲۳ ارائه می‌دهد. هر روش تجزیه و تحلیل شده و با توجه به معیارهایی مانند (۱) منافع، (۲) هزینه ها، (۳) خطرات، (۴) جریمه ها ، (۵) زمینه کسب‌وکار، و (۶) زمینه فنی و ویژگی‌های الزامات در پروژه‌های مختلف می‌توانند استفاده شوند. پاسخ‌های سوالات طرح شده به شرح زیر می‌باشد:

RQ1: چه تکنیک‌های اولویت‌بندی برای نیازمندی‌های نرم‌افزار وجود دارد؟

تکنیک‌های چند معیاره برای تصمیم‌گیری در اولویت‌بندی نیازمندیها تعداد زیادی هستند که می‌بایست با توجه به ساختار مدیریت پروژه و محدودیت‌های اعمال شده در آن، انتخاب شوند. مشکل اصلی در بیشتر این تکنیک‌ها نبود معیاری دقیق برای رتبه‌بندی می‌باشد، و معمولا از روی احساس، تصمیم‌گیری توسط ذینفعان اعمال می‌شود، در نتیجه تکنیک‌های جدید در مسیر بهبود روش‌های قدیمی با ریاضیات و الگوریتم‌ها، نظریه‌های مختلف، هوش‌مصنوعی و یادگیری ماشین ترکیب شده و ساختاری بهینه ایجاد کرده‌اند. این تکنیک‌ها در جداول ۱، ۲، ۳، ۴ و ۵ قابل مشاهده هستند.

RQ2: چگونه تکنیک‌ها را به صورت دسته‌بندی شده تقسیم کنیم؟

هر تکنیک با توجه به رویکردی که در اولویت‌بندی در نظر دارد در گروه‌بندی مشخصی قرار می‌گیرد، ممکن است یک تکنیک در چند گروه قابل دستیابی باشد. ویژگی‌های تکنیک‌ها در هر گروه شبیه به هم بوده اما دارای مزایا و معایب مختلفی هستند که باعث می‌شوند با توجه به استراتژی مدیریتی در پروژه‌های مختلفی استفاده شوند. دسته‌بندی که در مقالات مختلف اشاره شده شامل، تکنیک‌های مبتنی بر مقیاس اسمی، نسبت، فاصله و ترتیبی، تکنیک‌های مبتنی بر روش‌های کیفی و کمی که به سه زیرمجموعه روش‌های مبتنی بر نقشه برداری، امتیازدهی و بازی تقسیم می شوند، تکنیک‌های مبتنی بر منطق فازی، تکنیک‌های مبتنی بر الگوریتم بهینه‌سازی و تکنیک‌های مبتنی بر یادگیری ماشین می‌باشند.

RQ3: چه تکنیک‌هایی به صورت ترکیبی از روش‌ها ایجاد شده‌اند؟

تکنیک‌هایی که در دسته روش‌های فازی و الگوریتم‌های بهینه‌سازی قرار دارند بیشتر به صورت ترکیبی بوده و نقاط ضعف هر روش را با نقاط قوت روشی دیگر پر می‌کنند. معمولا محبوبیت تعدادی از روش‌ها، دلیلی برای ترکیب با روش‌های دیگر ایجاد می‌کند تا قابلیت استفاده از آن گسترش یابد.

RQ4: چه محدودیت‌هایی در هر تکنیک وجود دارد؟

جزئیات مربوط به هر تکنیک در جداول اشاره گردیده و به منظور تحقیقات آتی میتوان در مسیر توسعه روشی جدید با توجه به ویژگی‌های هر تکنیک اقدام کرد. این محدودیت‌ها شامل مواردی همچون سهولت استفاده، اثربخشی، مقیاس‌پذیری، کارایی، جذابیت، مصرف زمان، قابلیت اطمینان تحمل‌پذیری خطا و پیچیدگی می‌باشند.

## منابع

Achimugu, P., & Selamat, A. (2015). A hybridized approach for prioritizing software requirements based on K-means and evolutionary algorithms. *Computational Intelligence Applications in Modeling and Control*, 73–93.

Achimugu, P., Selamat, A., Ibrahim, R., & Mahrin, M. N. (2014). A systematic literature review of software requirements prioritization research. *Information and Software Technology*, *56*(6), 568–585. doi: https://doi.org/10.1016/j.infsof.2014.02.001

Ahmad, K. S., Ahmad, N., Tahir, H., & Khan, S. (2017a). Fuzzy_MoSCoW: A fuzzy based MoSCoW method for the prioritization of software requirements. *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, 433–437.

Ahmad, K. S., Ahmad, N., Tahir, H., & Khan, S. (2017b). Fuzzy_MoSCoW: A fuzzy based MoSCoW method for the prioritization of software requirements. *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, 433–437.

Ahuja, H., Sujata, & Batra, U. (2018). Performance enhancement in requirement prioritization by using least-squares-based random genetic algorithm. *Innovations in Computational Intelligence: Best Selected Papers of the Third International Conference on REDSET 2016*, 251–263.

Ali, A., Hafeez, Y., Ali, S., Hussain, S., Yang, S., Malik, A. J., & Abbasi, A. A. (2021). A data mining technique to improve configuration prioritization framework for component-based systems: an empirical study. *Information Technology and Control*, *50*(3), 424–442.

Alrezaamiri, H., Ebrahimnejad, A., & Motameni, H. (2020). Parallel multi-objective artificial bee colony algorithm for software requirement optimization. *Requirements Engineering*, *25*, 363–380.

Anwar, R., & Bashir, M. B. (2023). A Systematic Literature Review of AI-Based Software Requirements Prioritization Techniques. *IEEE Access*, *11*, 143815–143860. doi: 10.1109/ACCESS.2023.3343252

Avesani, P., Bazzanella, C., Perini, A., & Susi, A. (2005). Facing scalability issues in requirements prioritization with machine learning techniques. *13th IEEE International Conference on Requirements Engineering (RE'05)*, 297–305.

Avesani, P., Ferrari, S., & Susi, A. (2003). Case-based ranking for decision support systems. *Case-Based Reasoning Research and Development: 5th International Conference on Case-Based Reasoning, ICCBR 2003 Trondheim, Norway, June 23–26, 2003 Proceedings 5*, 35–49.

Babar, M. I., Ghazali, M., Jawawi, D. N. A., Shamsuddin, S. M., & Ibrahim, N. (2015). PHandler: an expert system for a scalable software requirements prioritization process. *Knowledge-Based Systems*, *84*, 179–202.

Babar, M. I., Ramzan, M., & Ghayyur, S. A. K. (2011). Challenges and future trends in software requirements prioritization. *International Conference on Computer Networks and Information Technology*, 319–324.

Bagnall, A. J., Rayward-Smith, V. J., & Whittley, I. M. (2001). The next release problem. *Information and Software Technology*, *43*(14), 883–890.

Baker, P., Harman, M., Steinhofel, K., & Skaliotis, A. (2006). Search based approaches to component selection and prioritization for the next release problem. *2006 22nd IEEE International Conference on Software Maintenance*, 176–185.

Bebensee, T., van de Weerd, I., & Brinkkemper, S. (2010). Binary priority list for prioritizing software requirements. *Requirements Engineering: Foundation for Software Quality: 16th International Working Conference, REFSQ 2010, Essen, Germany, June 30–July 2, 2010. Proceedings 16*, 67–78.

Beg, M. R., Verma, R. P., & Joshi, A. (2009). Reduction in number of comparisons for requirement prioritization using B-Tree. *2009 IEEE International Advance Computing Conference*, 340–344.

Berander, P., & Jönsson, P. (2006). Hierarchical cumulative voting (hcv)—prioritization of requirements in hierarchies. *International Journal of Software Engineering and Knowledge Engineering*, *16*(06), 819–849.

Bisht, A., & Kushwaha, M. (2020). Parameter Optimization of Software Requirement by Using Fuzzy Algebra. *Int. J. Res. Develop. Appl. Sci. Eng*, *20*(1).

Bollumpally, N. R., Evans, A. C., Gleave, S. W., Gromadzki, A. R., & Learmonth, G. (2019). A Machine Learning Approach to Workflow Prioritization. *2019 Systems and Information Engineering Design Symposium (SIEDS)*, 1–5.

Bugayenko, Y., Bakare, A., Cheverda, A., Farina, M., Kruglov, A., Plaksin, Y., Pedrycz, W., & Succi, G. (2023). Prioritizing tasks in software development: A systematic literature review. *Plos One*, *18*(4), e0283838.

Chatzipetrou, P., Angelis, L., Rovegård, P., & Wohlin, C. (2010). Prioritization of Issues and Requirements by Cumulative Voting: A Compositional Data Analysis Framework. *2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications*, 361–370. doi: 10.1109/SEAA.2010.35

Chaves-González, J. M., & Pérez-Toledano, M. A. (2015). Differential evolution with Pareto tournament for the multi-objective next release problem. *Applied Mathematics and Computation*, *252*, 1–13.

Dabbagh, M., & Lee, S. P. (2015). An approach for prioritizing NFRs according to their relationship with FRs. *Lecture Notes on Software Engineering*, *3*(1), 1–5.

de Souza, J. T., Maia, C. L. B., Ferreira, T. do N., Carmo, R. A. F. do, & Brasil, M. M. A. (2011). An ant colony optimization approach to the software release planning with dependent requirements. *Search Based Software Engineering: Third International Symposium, SSBSE 2011, Szeged, Hungary, September 10-12, 2011. Proceedings 3*, 142–157.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Duan, C., Laurent, P., Cleland-Huang, J., & Kwiatkowski, C. (2009). Towards automated requirements prioritization and triage. *Requirements Engineering*, *14*, 73–89.

Durillo, J. J., Zhang, Y., Alba, E., & Nebro, A. J. (2009). A study of the multi-objective next release problem. *2009 1st International Symposium on Search Based Software Engineering*, 49–58.

Ejnioui, A., Otero, C. E., & Qureshi, A. A. (2012). Software requirement prioritization using fuzzy multi-attribute decision making. *2012 IEEE Conference on Open Systems*, 1–6.

Feather, M. S., & Menzies, T. (2002). Converging on the optimal attainment of requirements. *Proceedings IEEE Joint International Conference on Requirements Engineering*, 263–270.

Felfernig, A. (2021). *AI Techniques for Software Requirements Prioritization*.

Gulzar, K., Sang, J., Ramzan, M., & Kashif, M. (2017). Fuzzy approach to prioritize usability requirements conflicts: An experimental evaluation. *IEEE Access*, *5*, 13570–13577.

Gupta, A., & Gupta, C. (2022). CDBR: A semi-automated collaborative execute-before-after dependency-based requirement prioritization approach. *Journal of King Saud University-Computer and Information Sciences*, *34*(2), 421–432.

Harman, M., Skaliotis, A., Steinhöfel, K., & Baker, P. (2006). Search–based approaches to the component selection and prioritization problem. *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, 1951–1952.

Hatton, S. (2007). Early prioritisation of goals. *Advances in Conceptual Modeling–Foundations and Applications: ER 2007 Workshops CMLSA, FP-UML, ONISW, QoIS, RIGiM, SeCoGIS, Auckland, New Zealand, November 5-9, 2007. Proceedings 26*, 235–244.

Ho, S.-C., & Chuang, W.-L. (2023). Identifying and prioritizing the critical quality attributes for business-to-business cross-border electronic commerce platforms. *Electronic Commerce Research and Applications*, *58*, 101239. doi: https://doi.org/10.1016/j.elerap.2023.101239

Huang, S. (2011). Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods. *Electronic Commerce Research and Applications*, *10*(4), 398–407.

Hudaib, A., Masadeh, R., Haj Qasem, M., & Alzaqebah, A. (2018). Requirements Prioritization Techniques Comparison. *Modern Applied Science*, *12*. doi: 10.5539/mas.v12n2p62

Hujainah, F., Bakar, R. B. A., Nasser, A. B., Al-haimi, B., & Zamli, K. Z. (2021). SRPTackle: A semi-automated requirements prioritisation technique for scalable requirements of software system projects. *Information and Software Technology*, *131*, 106501.

Jawale, B., & Bhole, A. T. (2015). Adaptive fuzzy hierarchical cumulative voting: a novel approach toward requirement prioritization. *International Journal of Research in Engineering and Technology*, *4*(5), 365–370.

Karlsson, J., Wohlin, C., & Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. *Information and Software Technology*, *39*(14–15), 939–947.

Karlsson, L., Thelin, T., Regnell, B., Berander, P., & Wohlin, C. (2007). Pair-wise comparisons versus planning game partitioning—experiments on requirements prioritisation techniques. *Empirical Software Engineering*, *12*(1), 3–33. doi: 10.1007/s10664-006-7240-4

Kukreja, N., Payyavula, S. S., Boehm, B., & Padmanabhuni, S. (2013). Value-Based Requirements Prioritization: Usage Experiences. *Procedia Computer Science*, *16*, 806–813. doi: https://doi.org/10.1016/j.procs.2013.01.084

Kumari, A. C., & Srinivas, K. (2016). Comparing the performance of quantum-inspired evolutionary algorithms for the solution of software requirements selection problem. *Information and Software Technology*, *76*, 31–64.

Leffingwell, D., & Widrig, D. (2000). *Managing software requirements: a unified approach*. Addison-Wesley Professional.

Lehtola, L., Kauppinen, M., & Kujala, S. (2004). Requirements prioritization challenges in practice. *International Conference on Product Focused Software Process Improvement*, 497–508.

Liu, X. F., Sun, Y., Veera, C. S., Kyoya, Y., & Noguchi, K. (2006). Priority assessment of software process requirements from multiple perspectives. *Journal of Systems and Software*, *79*(11), 1649–1660.

Marghny, M. H., El-Hawary, H. M., & Dukhan, W. H. (2017). An effective method of systems requirement optimization based on genetic algorithms. *Inf Sci Lett*, *6*(1), 15–28.

Misaghian, N., Motameni, H., & Rabbani, M. (2019). Prioritizing interdependent software requirements using tensor and fuzzy graphs. *Turkish Journal of Electrical Engineering and Computer Sciences*, *27*(4), 2697–2717.

Mishra, N., Khanum, M. A., & Agrawal, K. (2016). Approach to prioritize the requirements using fuzzy logic. *ACEIT Conference Proceeding*.

Momeni, H., Motameni, H., & Larimi, M. (2014). A neuro-fuzzy based approach to software quality requirements prioritization. *Int J Appl Inf Syst*, *7*, 15–20.

Mougouei, D., Powers, D. M. W., & Mougouei, E. (2019). A fuzzy framework for prioritization and partial selection of security requirements in software projects. *Journal of Intelligent & Fuzzy Systems*, *37*(2), 2671–2686.

Münch, J., Trieflinger, S., & Lang, D. (2019). What's hot in product roadmapping? Key practices and success factors. *Product-Focused Software Process Improvement: 20th International Conference, PROFES 2019, Barcelona, Spain, November 27–29, 2019, Proceedings 20*, 401–416.

Nguyen, S., Nguyen, H., Tran, N., Tran, H., & Nguyen, T. (2019). Feature-interaction aware configuration prioritization for configurable code. *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 489–501.

Ninaus, G., Felfernig, A., Stettinger, M., Reiterer, S., Leitner, G., Weninger, L., & Schanil, W. (2014). INTELLIREQ: intelligent techniques for software requirements engineering. In ECAI 2014 (pp. 1161–1166). IOS Press.

Pergher, M., & Rossi, B. (2013). Requirements prioritization in software engineering: A systematic mapping study. *2013 3rd International Workshop on Empirical Requirements Engineering (EmpiRE)*, 40–44. doi: 10.1109/EmpiRE.2013.6615215

Perini, A., Susi, A., & Avesani, P. (2012). A machine learning approach to software requirements prioritization. *IEEE Transactions on Software Engineering*, *39*(4), 445–461.

Quiroz, J. C., Louis, S. J., Shankar, A., & Dascalu, S. M. (2007). Interactive genetic algorithms for user interface design. *2007 IEEE Congress on Evolutionary Computation*, 1366–1373.

Racheva, Z., Daneva, M., Herrmann, A., & Wieringa, R. J. (2010). A conceptual model and process for client-driven agile requirements prioritization. *2010 Fourth International Conference on Research Challenges in Information Science (RCIS)*, 287–298.

Ramzan, M., Jaffar, M. A., & Shahid, A. A. (2011). Value based intelligent requirement prioritization (VIRP): expert driven fuzzy logic based prioritization technique. *International Journal Of Innovative Computing, Information And Control*, *7*(3), 1017–1038.

Rao, R. V., Rai, D. P., & Balic, J. (2017). A multi-objective algorithm for optimization of modern machining processes. *Engineering Applications of Artificial Intelligence*, *61*, 103–125.

Riegel, N., & Doerr, J. (2015). A Systematic Literature Review of Requirements Prioritization Criteria. S. A. Fricker & K. Schneider (Eds.), Requirements Engineering: Foundation for Software Quality (pp. 300–317). Cham: Springer International Publishing.

Sadia, H., Abbas, S. Q., & Faisal, M. (n.d.). *VOLATILE REQUIREMENT PRIORITIZATION: A FUZZY BASED APPROACH*.

Sadiq, M., & Jain, S. K. (2014). Applying fuzzy preference relation for requirements prioritization in goal oriented requirements elicitation process. *International Journal of System Assurance Engineering and Management*, *5*, 711–723.

Saliu, M. O., & Ruhe, G. (2007). Bi-objective release planning for evolving software systems. *Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, 105–114.

Shao, F., Peng, R., Lai, H., & Wang, B. (2017). DRank: A semi-automated requirements prioritization method based on preferences and dependencies. *Journal of Systems and Software*, *126*, 141–156.

Sharif, N., Zafar, K., & Zyad, W. (2014). Optimization of requirement prioritization using computational intelligence technique. *2014 International Conference on Robotics and Emerging Allied Technologies in Engineering (ICREATE)*, 228–234.

Singh, Y. V., Kumar, B., & Chand, S. (2021). Requirements Prioritization Using Logarithmic Fuzzy Trapezoidal Approach (LFTA). *Innovations in Information and Communication Technologies (IICT-2020) Proceedings of International Conference on ICRIHE-2020, Delhi, India: IICT-2020*, 309–318.

Singh, Y. V., Kumar, B., Chand, S., & Sharma, D. (2019). A hybrid approach for requirements prioritization using logarithmic fuzzy trapezoidal approach (LFTA) and artificial neural network (ANN). *Futuristic Trends in Network and Communication Technologies: First International Conference, FTNCT 2018, Solan, India, February 9–10, 2018, Revised Selected Papers 1*, 350–364.

Somohano-Murrieta, J. C. B., Ocharán-Hernández, J. O., Sánchez-García, A. J., & de los Ángeles Arenas-Valdés, M. (2020). Requirements Prioritization Techniques in the last decade: A Systematic Literature Review. *2020 8th International Conference in Software Engineering Research and Innovation (CONISOFT)*, 11–20. doi: 10.1109/CONISOFT50191.2020.00013

Thakurta, R. (2013). A framework for prioritization of quality requirements for inclusion in a software project. *Software Quality Journal*, *21*, 573–597.

Tonella, P., Susi, A., & Palma, F. (2010). Using interactive GA for requirements prioritization. *2nd International Symposium on Search Based Software Engineering*, 57–66.

Trieflinger, S., Münch, J., Bogazköy, E., Eißler, P., Schneider, J., & Roling, B. (2021). How to Prioritize Your Product Roadmap When Everything Feels Important: A Grey Literature Review. *2021 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, 1–9. doi: 10.1109/ICE/ITMC52061.2021.9570243

Valsala, S., & Nair, A. R. (2016). Requirement Prioritization and Scheduling in Software Release Planning Using Hybrid Enriched Genetic Revamped Integer Linear Programming Model. *Research Journal of Applied Sciences, Engineering and Technology*, *12*(3), 347–354.

Vestola, M. (2010). A comparison of nine basic techniques for requirements prioritization. *Helsinki University of Technology*, 1–8.

Voola, P., & Babu, A. V. (2012). Requirements uncertainty prioritization approach: a novel approach for requirements prioritization. *Softw Eng Int J (SEIJ)*, *2*(2), 37–49.

Yaseen, M., Ibrahim, N., & Mustapha, A. (2019). Requirements Prioritization and using Iteration Model for Successful Implementation of Requirements. *International Journal of Advanced Computer Science and Applications*, *10*. doi: 10.14569/IJACSA.2019.0100115

Yaseen, M., Mustapha, A., & Ibrahim, N. (2019). Prioritization of Software Functional Requirements: Spanning Tree based Approach. *International Journal of Advanced Computer Science and Applications*, *10*. doi: 10.14569/IJACSA.2019.0100767

Zhang, Y., Harman, M., & Mansouri, S. A. (2007). The multi-objective next release problem. *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, 1129–1137.