سیستم تضمین کیفیت نرم‌افزار و استانداردهای متداول

علی کریمی
استادیار دانشگاه جامع امام حسین (ع)
علی طلوعی‌فر
دانشجوی کارشناسی ارشد دانشگاه جامع امام حسین (ع)

چکیده

محیط توسعه و نگهداری نرم‌افزار برای ایجاد سامانه‌های نرم‌افزاری بدون خطا، به‌طور مستقیم تحت تاثیر مولفه‌های تضمین کیفیت نرم‌افزار قرار می‌گیرد. ماهیت منحصربه‌فرد تولید نرم‌افزار در مقایسه با سایر محصولات صنعتی، نیاز به رویکرد و ابزارهای کیفی متفاوتی را برای توسعه و نگهداری نرم‌افزار ایجاد می‌کند. از طرفی، تغییرات مداوم در زمینه کسب‌وکار و توسعه نرم‌افزار، درک چگونگی پاسخ‌گویی تضمین کیفیت نرم‌افزار را مهم می‌سازد. تضمین کیفیت نرم‌افزار امری حیاتی است که برای موفقیت پروژه‌های نرم‌افزاری ضروری می‌باشد. سیستم تضمین کیفیت نرم‌افزار بیانگر یک معماری تضمین کیفیت نرم‌افزار است که در قالب شش کلاس قابل دسته‌بندی است. معماری تضمین کیفیت نرم‌افزار، رویکردی جهت مدیریت و بهبود کیفیت در طول چرخه حیات پروژه نرم‌افزاری است. اگرچه طیف گسترده‌ای از مدل‌های تضمین کیفیت نرم‌افزار در دسترس هستند، اما شش مولفه سیستم تضمین کیفیت نرم‌افزار که پایه و اساس کیفیت نرم‌افزار را تشکیل می‌دهند، به همراه ۴ استاندارد پرکاربرد جهانی، در این مقاله مورد بحث قرار خواهند گرفت. هر یک از این مولفه‌ها به مقابله با منشاء خطاهای نرم‌افزاری مختلف می‌پردازند تا سطح قابل قبولی از کیفیت نرم‌افزار فراهم شود. این مقاله به توسعه‌دهندگان، مدیران پروژه و علاقه‌مندان به تضمین کیفیت نرم‌افزار کمک می‌کند تا با بهره‌گیری از مولفه‌های معماری تضمین کیفیت نرم‌افزار و استفاده از استانداردها، کیفیت نرم‌افزار را افزایش داده و با بهبود بهره‌وری، جلب اعتماد مشتریان و افزایش رقابت‌پذیری در سازمان‌ها، موفقیت پروژه‌های خود را تضمین نمایند.

واژه‌های کلیدی: سیستم تضمین کیفیت نرم‌افزار، توسعه نرم‌افزار، استانداردهای تضمین کیفیت نرم‌افزار، معماری تضمین کیفیت نرم‌افزار.

۱- مقدمه

انقلاب صنعتی چهارم بدون شک چالش‌های قابل‌توجهی را برای توسعه نرم‌افزارهای «سنتی» ایجاد خواهد کرد. این امر در نتیجه ماهیت غیر قابل پیش‌بینی رفتار سامانه‌های نرم‌افزاری، فقدان کنترل متمرکز، عدم مقیاس‌پذیری، عدم تحمل‌پذیری خطا و پایین بودن قابلیت اطمینان سامانه‌های نرم‌افزاری اتفاق می‌افتد. از سوی دیگر، بخش عمده‌ای از این مشکلات ممکن است به‌عنوان فرصت‌هایی برای افزایش و گسترش فرایندهای آزمون و توسعه نرم‌افزار تلقی شوند (Bhanushali, 2023).

چالش اصلی در مورد تضمین کیفیت است؛ زیرا به قابلیت اطمینان و اتکاپذیری نرم‌افزار مربوط می‌شود. هم آزمونگرها و هم متخصصان تضمین کیفیت، اغلب از تضمین کیفیت به‌عنوان بخش مهمی از فرایند توسعه در صنعت نرم‌افزار استفاده می‌کنند (Bhanushali, 2023).

تضمین کیفیت نرم‌افزار[1] بخش مهم و جدایی‌ناپذیر از فرایند توسعه نرم‌افزار محسوب می‌شود. در طول تکامل روش‌های توسعه نرم‌افزار، عوامل متعددی به اهمیت نقش تضمین کیفیت نرم‌افزار کمک کرده‌اند. افزایش اتصال برنامه‌های کاربردی به اینترنت مهم است، زیرا نیازمند توجه بیشتر به تجزیه‌وتحلیل محصولات برای ویژگی‌های امنیتی و عملکرد آن است. عامل دوم، افزایش تقاضا برای انتشار نرم‌افزارهای مختلف، نیاز به بهینه‌سازی فرایندهای اعتبارسنجی و تأیید را افزایش می‌دهد تا محصول سریع‌تر و باکیفیت بالاتر تحویل داده شود (Silva Farias et al., 2024).

سازمان‌ها توجه زیادی به توسعه نرم‌افزار دارند؛ زیرا محصولات با کیفیت بالا نیازهای مشتریان را برآورده می‌کنند، از شهرت آنها محافظت می‌کنند و عملیات تجاری را بهبود می‌بخشند (Wong et al., 2022).

SQA شامل فعالیت‌های مختلفی است که در طول فرایند توسعه نرم‌افزار انجام می‌شود. هنگامی که این فعالیت‌ها به‌خوبی انجام شوند، نتایج مطلوبی در راستای تامین نیازمندی‌ها و انتظارات کاربران حاصل خواهد شد. همچنین، انجام دقیق فعالیت‌های تضمین کیفیت نرم‌افزار، مزایای مختلفی از جمله کاهش هزینه‌های توسعه و آزمون، بهبود کیفیت و بهره‌وری سامانه‌های نرم‌افزاری و افزایش رضایتمندی مشتریان را در پی دارد (Zhao et al., 2021).

فعالیت‌های تضمین کیفیت نرم‌افزار؛ شامل برنامه‌ریزی، تعریف و اجرای استانداردهای کیفیت، ممیزی و بازبینی[2]، آزمون و گزارش‌دهی درباره نتایج می‌باشد. اعتبارسنجی نیازمندی‌ها با استفاده از عناصر مهمی نظیر نمونه‌سازی، بازرسی مبتنی بر دانش، بازرسی مبتنی بر آزمون، مدل‌سازی و ارزیابی به‌عنوان سنگ‌بنایی برای SQA در نظر گرفته می‌شود؛ (Atoum et al., 2021).

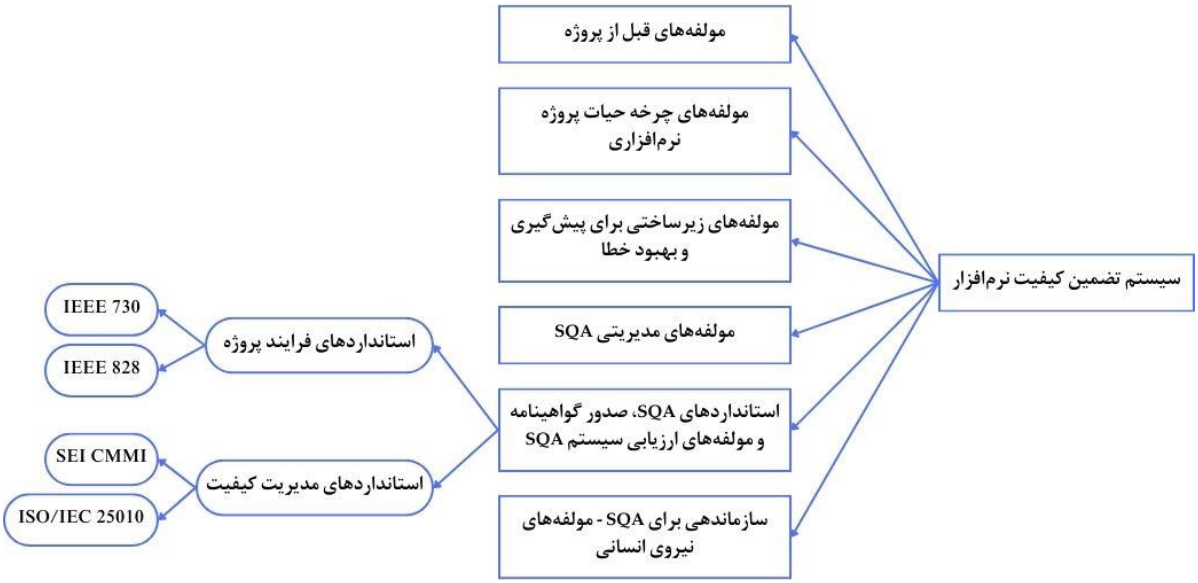
استانداردهای مهندسی نرم‌افزار منابع دانش مدون هستند. مطالعات نشان داده‌اند که استفاده از استانداردها مزایایی مانند قابلیت همکاری محصول، افزایش بهره‌وری، افزایش سهم بازار و بهبود تعامل با ذینفعان نظیر شرکت‌ها، سازمان‌های دولتی و عموم مردم را به همراه دارد. استانداردها و اسناد فنی مرتبط را می‌توان نوعی انتقال فناوری در نظر گرفت و در صورت انتخاب و استفاده صحیح از استانداردها، باید تأثیرات اقتصادی در سازمان داشته باشند (Laporte et al., 2023).

با توجه به مطالعات انجام‌شده در حوزه سیستم تضمین کیفیت نرم‌افزار و استانداردهای متداول این حوزه با بررسی مقالات پیشین، آرایه‌شناسی[3] مولفه‌های معماری تضمین کیفیت نرم‌افزار در شش دسته به‌صورت شکل ۱ ارائه می‌شود. در این شکل، شش مولفه مذکور به همراه زیربخش‌های هرکدام، جداگانه بررسی شده‌اند.

بخش باقیمانده مقاله به شرح زیر تنظیم شده است. بخش دوم، مفهوم تضمین کیفیت و بخش سوم تضمین کیفیت نرم‌افزار را مورد بحث قرار می‌دهد. در بخش چهارم، شش مؤلفه سیستم SQA به همراه بررسی زیربخش‌های هر کدام، ارائه خواهد شد. در نهایت، بخش پنجم شامل نتیجه‌گیری است.

---

[1] Software Quality Assurance (SQA)
[2] Reviewing and auditing
[3] Taxonomy



شکل ۱ - آرایه‌شناسی معماری تضمین کیفیت نرم‌افزار

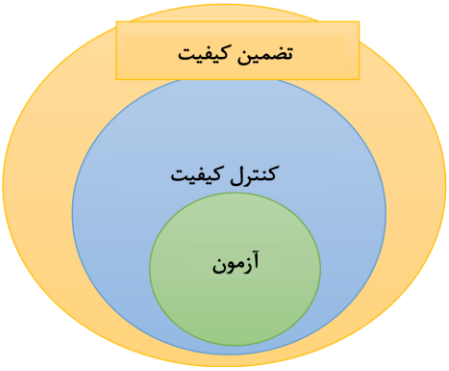
۲- تضمین کیفیت

استاندارد ISO 9000 تضمین کیفیت را که گاهی اوقات QA[1] نامیده می‌شود، به‌عنوان بخشی از مدیریت کیفیت تعریف می‌کند که مسئول اطمینان از برآورده‌شدن نیازمندی‌های رفع نقص است. تثبیت اینکه برنامه به طور مؤثر خواسته‌های مخاطبان هدف را برآورده می‌کند، هدف تضمین کیفیت است. در هر مرحله از چرخه حیات نرم‌افزار، فنون تضمین کیفیت برای تضمین سطح بالای توسعه برنامه، طراحی شده است. این اقدامات اغلب قبل از توسعه برنامه انجام می‌شود و در حین ساخت ادامه می‌یابد. ایجاد و اجرای رویه‌ها و استانداردهایی باهدف بهبود چرخه حیات توسعه و همچنین حصول اطمینان از پیروی از این رویه‌ها، از وظایف

[1] Quality Assurance (QA)

تضمین کیفیت است. هدف اصلی تضمین کیفیت، ریشه‌کن‌کردن خطاها در هر مرحله از توسعه نرم‌افزار است و درعین‌حال اطمینان از اینکه محصول نهایی همواره در حال بهبود است (Bhanushali, 2023).

برخلاف تضمین کیفیت که بر اطمینان از ایجاد نرم‌افزار باکیفیت تمرکز دارد، کنترل کیفیت فعالیتی است که پس از تولید یک برنامه، کیفیت آن را جمع‌آوری و بررسی می‌کند؛ بنابراین آزمون، جزء سیستم کنترل کیفیت است و کنترل کیفیت بخشی از سیستم تضمین کیفیت محسوب می‌شود. رابطه بین آزمون، کنترل کیفیت و تضمین کیفیت در شکل ۲ نشان داده شده است. ایجاد دستورالعمل‌ها و استانداردها، کنترل کیفیت و انتخاب ابزار مناسب، همه فعالیت‌های مرتبط با تضمین کیفیت هستند (Bhanushali, 2023).



شکل ۲ - رابطه آزمون، کنترل کیفیت و تضمین کیفیت (Bhanushali, 2023)

طبق استاندارد ISO 9126، کیفیت نرم‌افزار شامل طیف گسترده‌ای از ویژگی‌ها است که به توانایی آن در برآوردن نیازهای صریح یا ضمنی همه طرف‌ها مربوط می‌شود. برتری کلی یک محصول نرم‌افزاری تحت‌تأثیر تعدادی از عوامل یا ابعادی است که کیفیت نرم‌افزار را تشکیل می‌دهند (Korchenko et al., 2021):

- رویه‌های فناوری مورداستفاده در توسعه نرم‌افزار، تأثیر قابل‌توجهی بر کیفیت محصول نهایی دارد.
- کیفیت داخلی نرم‌افزار به ویژگی‌های آن اشاره دارد که بدون توجه به نحوه عملکرد یک برنامه کاربردی در عمل، وجود دارد.
- رفتار و عملکرد نرم‌افزار به‌عنوان کیفیت خارجی شناخته می‌شود.
- قابلیت استفاده و کارایی نرم‌افزار در سناریوهای خاص کاربر، ممکن است بسته به نحوه عملکرد آن در شرایط مختلف ارزیابی شود.

۳- تضمین کیفیت نرم‌افزار

کیفیت نرم‌افزار به مدت طولانی یک نگرانی مهم در توسعه نرم‌افزار بوده و برای ارزیابی موفقیت محصولات نرم‌افزاری استفاده شده است (Al-Qutaish, 2010). کیفیت پایین در سیستم‌های حیاتی و بی‌درنگ می‌تواند منجر به ضرر مالی، ناتوانی دائمی، شکست در انجام ماموریت یا حوادث ناگوار شود (Jamwal, 2010).

ISO 8402:1986 کیفیت نرم‌افزار را این‌گونه تعریف می‌کند: «مجموعه ویژگی‌ها و مشخصات یک محصول یا خدمت که بر توانایی آن در برآوردن نیازهای مشخص یا ضمنی تأثیر می‌گذارد». ISO 9001:2008 این دیدگاه را به شرح زیر گسترش داد:« کیفیت چیزی را می‌توان با مقایسه مجموعه‌ای از ویژگی‌های ذاتی با مجموعه‌ای از نیازمندی‌ها تعریف کرد. کیفیت بالا در صورتی حاصل می‌شود که ویژگی‌های ذاتی همه نیازمندی‌ها را برآورده کنند. اگر آن ویژگی‌ها همه نیازمندی‌ها را برآورده نکنند، سطح کیفیتِ

پایین یا ضعیف حاصل می‌شود» (Ndukwe et al., 2023). چندین مدل کیفیت و ویژگی برای اندازه‌گیری کیفیت نرم‌افزار پیشنهاد شده است (Kumar & Gupta, 2017; Miguel et al., 2014; Thapar et al., 2012).

تضمین کیفیت مطابق تعریف ISO 24765 عبارت است از (Laporte & April, 2018):

۱) یک الگوی برنامه‌ریزی‌شده و سیستماتیک از کلیه اقدامات لازم برای ایجاد اطمینان کافی مبنی بر انطباق یک کالا یا محصول با الزامات فنی تعیین‌شده.

۲) مجموعه‌ای از فعالیت‌ها که برای ارزیابی فرایند توسعه یا تولید محصولات طراحی شده‌اند.

۳) فعالیت‌های برنامه‌ریزی شده و سیستماتیک که در چارچوب سیستم کیفیت اجرا شده و به میزان لازم به نمایش درآمده‌اند تا اطمینان کافی ارائه دهند که یک موجودیت نیازهای کیفیت را برآورده خواهد کرد.

کیفیت نرم‌افزار دارای هشت ویژگی سطح بالا مطابق با استاندارد ISO/IEC 25010 است که هر یک از آنها از مجموعه‌ای از ویژگی‌های فرعی مرتبط تشکیل شده است که در شکل ۳ نشان داده شده است (Nyári & Kerti, 2021).

شکل ۳ – آرایه‌شناسی ویژگی‌های کیفیت نرم‌افزار بر اساس استاندارد ISO/IEC 25010 (Nyári & Kerti, 2021)

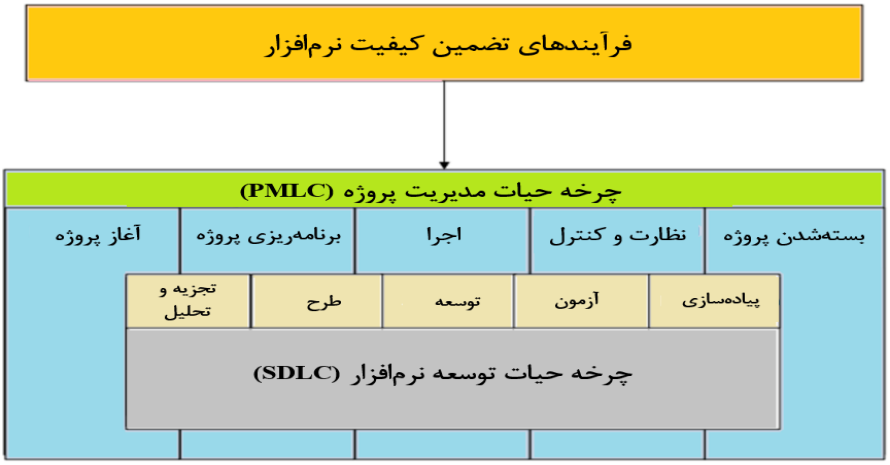
بر اساس تعریف ارائه‌شده توسط IEEE Std. 730-2014، تضمین کیفیت نرم‌افزار یک الگوی برنامه‌ریزی‌شده و سیستماتیک از تمام اقدامات لازم برای ایجاد اطمینان کافی در مورد مطابقت کالا یا محصول با الزامات فنی تعیین‌شده است. تضمین کیفیت[1] باید در طول تمام فعالیت‌های توسعه‌ای به‌کار گرفته شود تا در اعلام اینکه محصول نیازهای مشخص‌شده را برآورده می‌کند، سطح بالایی از اطمینان فراهم شود (Karmakar et al., 2023).

یک تعریف جایگزین از SQA، به‌عنوان گروهی از خدمات یا تلاش‌هایی است که اعتماد و اطمینان را تقویت می‌کند. کاهش ریسک، ارزیابی کنترل داخلی و بهبود کیفیت اهداف اصلی خدمات تضمینی هستند. کنترل‌ها قدرت جلوگیری از مشکلات پروژه را قبل از وقوع دارند. کنترل‌ها به‌عنوان یک سیستم کامل از رویه‌ها، روش‌ها یا قوانینی در نظر گرفته می‌شوند که برای جلوگیری و شناسایی نمونه‌های شکست پروژه نرم‌افزاری قبل از وقوع در نظر گرفته می‌شوند. یک فرایند تضمین کیفیت نرم‌افزار باید دارای سازوکارهای کنترلی برای تضمین انطباق با خط‌مشی‌ها، رویه‌ها و استانداردهای سازمان باشد. فرایند SQA باید با موفقیت با مراحل متعدد چرخه حیات مرتبط شود. در نتیجه، برای SQA دشوار است که در اواخر چرخه حیات به‌عنوان آخرین تلاش برای بهبود کیفیت قبل از تکمیل فعالیت توسعه انجام شود (Bhanushali, 2023).

روش SQA چرخه حیات خاص خود را دارد و هر مرحله از پروژه توسعه نرم‌افزار را ارزیابی می‌کند. در اکثر مواقع، این امر مستلزم برنامه‌ریزی پرزحمت و همچنین جمع‌آوری اطلاعات در قالب چندین عکس از آثار گذشته است. شما ممکن است چیزهایی مانند

[1] Quality Assurance (QA)

طرح SQA، طرح مدیریت پروژه یا معیارهای پروژه را به‌عنوان مثال، در مخزن اسناد پروژه پیدا کنید. SQA یک رویکرد به‌جای یک فرایند واحد است که اطمینان و اعتبار را در تمام فعالیت‌های مدیریت پروژه و چرخه حیات توسعه فراهم می‌کند. شکل ۴ فرایند SQA را نشان می‌دهد و توضیح می‌دهد که چگونه مراحل مدیریت پروژه و توسعه را یکپارچه می‌کند (Bhanushali, 2023).



شکل ۴ – فرآیند تضمین کیفیت نرم‌افزار همراه با PMLC و SDLC (Bhanushali, 2023)

در مجموع هشت مرحله جداگانه در چرخه حیات مدیریت پروژه[1] یکپارچه و چرخه حیات توسعه سیستم[2] وجود دارد که شامل برنامه‌ریزی، تجزیه‌وتحلیل، طراحی، توسعه، پیاده‌سازی و آزمون می‌باشد. یک روش تضمین کیفیت نرم‌افزار با هر مرحله از چرخه حیات توسعه نرم‌افزار مطابقت دارد. برنامه‌ریزی SQA شامل؛ تضمین نیازمندی‌ها، تضمین طراحی، تضمین توسعه، تضمین پیاده‌سازی و تضمین آزمون برخی از رویه‌هایی هستند که در این دسته قرار می‌گیرند. یک حلقه بازخورد که مربوط به مرحله PMLC یا SDLC است برای هر مرحله SQA وجود دارد. هدف اصلی حلقه بازخورد، ارائه بازخورد درباره نقاط ضعف یا مشکلاتی است که در طول روند تضمین کیفیت نرم‌افزار به‌منظور تسهیل در بهبود مستمر، یافت می‌شود (Bhanushali, 2023).

٤- مولفه‌های سیستم تضمین کیفیت نرم‌افزار

سیستم SQA، بیانگر یک معماری SQA است که در قالب شش کلاس، قابل دسته‌بندی است. یک سیستم SQA همیشه طیف گسترده‌ای از اجزای SQA را ترکیب می‌کند که همه آنها برای به چالش کشیدن منابع متعدد خطاهای نرم‌افزاری و دستیابی به سطح قابل قبولی از کیفیت نرم‌افزار به کار می‌روند. وظیفه SQA در زمینه تضمین کیفیت به دلیل ویژگی‌های خاص نرم‌افزار منحصربه‌فرد است. علاوه‌براین، محیطی که در آن توسعه و نگهداری نرم‌افزار انجام می‌شود، به‌طور مستقیم بر مولفه‌های SQA تأثیر می‌گذارد (Galin, 2018).

به‌طور ضمنی، SQA یک مرحله نهایی توسعه نیست، بلکه یک سازوکار ارزیابی مداوم است که در طول نگهداری و توسعه بعدی ادامه می‌یابد. مولفه‌های سیستم SQA را می‌توان به شش کلاس طبقه‌بندی کرد (SenthilMurugan & Prakasam, 2014).

۴-۱- مولفه‌های قبل از پروژه

مولفه‌های این بخش به بهبود مراحل مقدماتی قبل از شروع و استفاده از پروژه، می‌پردازند و عبارت‌اند از:

[1] Project management life cycle (PMLC)
[2] Software development life cycle (SDLC)

- بازبینی قرارداد[1]
- طرح‌های توسعه و کیفیت[2]

این فعالیت‌ها برای حصول اطمینان از تعریف مناسب تعهدات پروژه انجام می‌شود. این مؤلفه‌ها اساساً برای اطمینان حاصل‌کردن از تعریف صحیح تعهدات پروژه‌های انجام‌شده با درنظرگرفتن منابع موردنیاز، برنامه زمان‌بندی و بودجه است. در این مرحله طرح توسعه پروژه و طرح کیفیت نیز تدوین می‌شود. استاندارد IEEE 730 به حداقل مستندات برای طرح نیاز دارد که شامل توصیف نیازمندی‌های نرم‌افزار[3]، توصیف طراحی نرم‌افزار[4]، برنامه‌های تأیید و اعتبارسنجی و گزارش نتیجه تأیید و گزارش نتایج اعتبارسنجی می‌باشد (Kamarudin, 2012).

۴-۲- مولفه‌های چرخه حیات پروژه نرم‌افزاری

چرخه حیات پروژه از دو مرحله کلی شامل؛ مرحله چرخه حیات توسعه و مرحله عملیات - نگهداری تشکیل می‌شود. مولفه‌های چرخه حیات توسعه، خطاهای طراحی و کدنویسی را پیدا می‌کنند و به موارد بازبینی/مرور، نظرات خبرگان و متخصصین و آزمون نرم‌افزار تقسیم‌بندی می‌شوند. مرحله عملیات - نگهداری شامل مولفه‌های ویژه نگهداری است که عمدتاً برای بهبود عملکرد نگهداری نرم‌افزار استفاده می‌شوند. در برخی موارد، فعالیت‌هایی برای اطمینان از کیفیت مشارکت‌کنندگان خارجی در توسعه و نگهداری نیز باید مورد توجه قرار گیرد (Kamarudin, 2012).

دومین مؤلفه به ارزیابی و افزایش فعالیت‌های چرخه حیات پروژه اختصاص دارد و شامل پنج مولفه اصلی است. این مولفه‌ها، برای بهبود فعالیت‌های فرایند توسعه نرم‌افزار به‌کار گرفته می‌شوند. چندین مولفه SQA، در نقاط مختلف وارد چرخه حیات توسعه نرم‌افزار می‌شوند. استفاده از آنها باید قبل از شروع پروژه برنامه‌ریزی شود. پنج مولفه عبارت‌اند از (Galin, 2018):

- بازبینی‌ها[5]
  - بازبینی‌های رسمی طراحی[6] (DRs)
  - بازبینی‌های همتایان[7]
- نظرات خبرگان و متخصصین[8]
- آزمون نرم‌افزار[9]
- نگهداری نرم‌افزار[10]
- اطمینان از کیفیت کار پیمانکاران فرعی

---

[1] Contract review
[2] Development and quality plans
[3] Software Requirement Description (SDR)
[4] Software Design Decription (SDD)
[5] Reviews
[6] Formal design reviews
[7] Peer reviews
[8] Expert opinions
[9] Software testing
[10] Software maintenance

۳-۴- مولفه‌های زیرساختی برای پیشگیری و بهبود خطا

اهداف اصلی این مؤلفه‌ها که مبتنی بر تجربه SQA انباشته‌شده در سراسر سازمان است، ازبین‌بردن خطاها و یا حداقل کاهش نرخ خطاها می‌باشد. این هدف با استفاده از ابزارهایی مانند رویه‌ها، ابزارهای پشتیبانی مانند الگوها و چک‌لیست‌ها، دستورالعمل‌های آموزشی، اقدامات پیشگیرانه، مدیریت پیکربندی و کنترل اسناد به دست می‌آید (Kamarudin, 2012).

اهداف زیرساختی SQA به‌منظور بهبود بهره‌وری نرم‌افزار است. این مؤلفه‌ها، طیف وسیعی از پروژه‌ها و خدمات نگهداری نرم‌افزار را پوشش می‌دهند. در سالیان اخیر، شاهد رشد و توسعه ابزارهای کامپیوتری خودکار برای کاربردهای این مؤلفه‌ها بوده‌ایم. مولفه‌های زیرساختی SQA، شامل شش مورد زیر است (Galin, 2018):

- رویه‌ها و دستورالعمل‌های کار[1]
- قالب‌ها و چک‌لیست‌ها[2]
- آموزش کارکنان، بازآموزی و صدورگواهینامه[3]
- اقدامات پیشگیرانه و اصلاحی[4]
- مدیریت پیکربندی[5]
- کنترل مستندسازی[6]

۴-۴- مولفه‌های مدیریتی SQA

هدف اصلی این مؤلفه‌ها کنترل فعالیت‌های توسعه و نگهداری است. مداخله زودهنگام مدیریت برای جلوگیری یا به‌حداقل‌رساندن شکست‌های زمان‌بندی و بودجه مهم است. ابزارهای این مؤلفه، شامل سه مورد زیر است (Kamarudin, 2012).

این گروه از مولفه‌ها چندین هدف را اقناع می‌کنند (Galin, 2018):

- ✓ عمدتاً به دنبال کنترل فعالیت‌های توسعه و نگهداری نرم‌افزار هستند.
- ✓ اقدامات اولیه پشتیبانی مدیریتی به‌منظور جلوگیری یا کاهش شکست‌های زمانی، بودجه‌ای و پیامدهای ناشی از آنها در پروژه را انجام می‌دهند.

به‌طورکلی، مولفه‌های مدیریتی SQA از موارد زیر پشتیبانی می‌کنند:

- ✓ کنترل مدیریتی پروژه‌های توسعه نرم‌افزار
- ✓ کنترل مدیریتی خدمات نگهداری نرم‌افزار

مولفه‌های کنترلی شامل موارد زیر است:

- کنترل پیشرفت پروژه[7] (از جمله قرارداد نگهداری)
- سنجه‌های کیفیت نرم‌افزار[8]

---

[1] Procedures and work instructions
[2] Templates and checklists
[3] Staff training, retraining, and certification
[4] Preventive and corrective actions
[5] Configuration management
[6] Documentation control
[7] Project progress control
[8] Software quality metrics

- هزینه‌های کیفیت نرم‌افزار[1]

۴-۵- استانداردهای SQA، صدور گواهینامه و مولفه‌های ارزیابی سیستم SQA

امروزه، کیفیت سیستم‌های نرم‌افزاری برای شرکت‌ها در ارائه خدمات و محصولات باکیفیت بسیار مهم است. بااین‌حال، تعداد زیادی از پروژه‌های نرم‌افزاری هنوز با شکست مواجه می‌شوند. برای افزایش احتمال موفقیت پروژه‌ها، استفاده از استانداردهای کیفیت نرم‌افزار برای هدایت فرایند توسعه، مناسب و ضروری است (Silega et al., 2023).

معرفی استانداردهای ISO در حوزه کیفیت نرم‌افزار، علاوه بر صرفه‌جویی در هزینه، مزایای دیگری را نیز پس از مدت معینی به همراه دارد. به‌عنوان مثال، مزایای سازمان از اجرای استاندارد ISO 9001 و صدور گواهینامه سیستم را می‌توان به مزایای داخلی و خارجی تقسیم نمود (Aarts & Vos, 2001; Zimon, 2017a, 2017b; Zimon & Dellana, 2020). مزایای داخلی ممکن است شامل؛ اطمینان از پایداری و تکرارپذیری فرآیندها، تضمین و حفظ حاکمیت سازمانی از طریق رویه‌های عملیاتیِ واضح تعریف‌شده و مسئولیت‌ها و اختیارات مرتبط با آن‌ها، اطمینان از داده‌ها و اطلاعاتِ بدون ابهامِ استخراج‌شده از نظارت و اندازه‌گیری محصولات و خدمات و همچنین فرآیندهایی برای استفاده در مدیریت سازمان و بهبود مستمر آن باشد (Zimon et al., 2024).

یکی از شش کلاس در دسته‌بندی معماری SQA عبارت است از: استانداردهای SQA، صدور گواهینامه و مولفه‌های ارزیابی سیستم SQA. این مولفه‌ها، استانداردهای مدیریتی و حرفه‌ای بین‌المللی را در سازمان پیاده‌سازی و اجرا می‌کنند. اهداف اصلی این گروه از مولفه‌ها عبارت‌اند از (Galin, 2018):

۱) استفاده از دانش حرفه‌ای و تخصصی بین‌المللی؛

۲) بهبود همکاری با سیستم‌های کیفیت سازمان‌های دیگر؛

۳) سنجش و ارزیابی حرفه‌ای اهداف و دستاوردهای سیستم‌های کیفیت سازمان.

استانداردهای موجود به دو زیر کلاس طبقه‌بندی می‌شوند که هر دو ممکن است موردنیاز مشتریان باشند:

۱) استانداردهای فرایند پروژه؛[2]

۲) استانداردهای مدیریت کیفیت.[3]

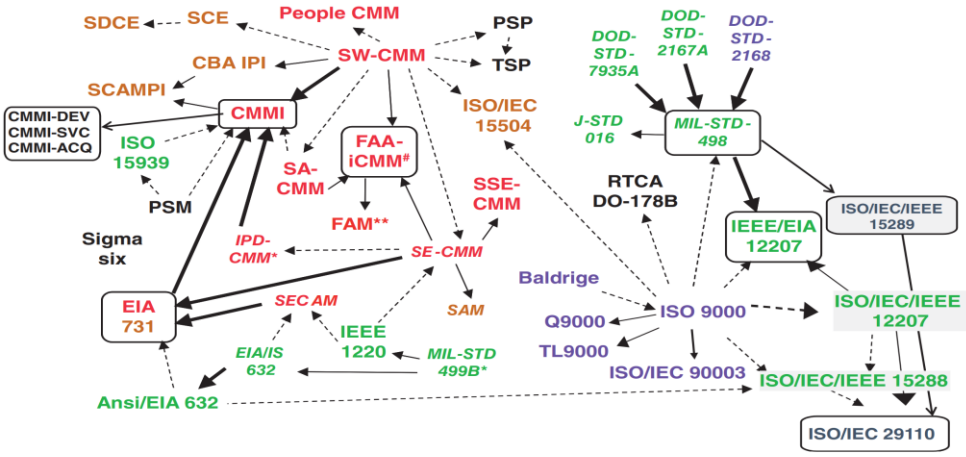
توسعه استانداردهای SQA توسط چندین مؤسسه استاندارد ملی و بین‌المللی انجام شده است؛ سازمان‌های حرفه‌ای و صنعت‌محور که منابع قابل‌توجهی را در توسعه و به‌روزرسانی پروژه‌های استاندارد سرمایه‌گذاری می‌کنند. مؤسسات و سازمان‌های زیر از برجسته‌ترین توسعه‌دهندگان استانداردهای SQA و مهندسی نرم‌افزار هستند و در این زمینه شهرت و جایگاه بین‌المللی کسب کرده‌اند (Galin, 2018):

- IEEE (مؤسسه مهندسین برق و الکترونیک) انجمن کامپیوتر؛
- ISO (سازمان بین‌المللی استاندارد)؛
- ANSI (مؤسسه استاندارد ملی آمریکا[4])؛
- IEC (کمیسیون بین‌المللی الکتروتکنیکی)؛
- EIA (اتحادیه صنایع الکترونیک[1]).

---

[1] Software quality costs
[2] Project Process Standards
[3] Quality management standards
[4] American National Standards Institute (ANSI)

سیر تحول و توسعه استانداردها و مدل‌ها در شکل ۵ نشان داده شده است.



شکل ۵ – توسعه استانداردها و مدل‌ها (Laporte & April, 2018)

خصوصیات دو کلاس استاندارد در جدول ۱ خلاصه شده است.

جدول ۱ – مقایسه کلاس‌های استاندارد (Galin, 2018)

مشخصات	استانداردهای فرایند پروژه	استانداردهای مدیریت کیفیت
واحد هدف	یک تیم پروژه توسعه و/یا تعمیر و نگهداری نرم‌افزار.	مدیریت توسعه و/یا نگهداری نرم‌افزار و واحدهای خاص SQA.
تمرکز اصلی	روش‌شناسی برای انجام پروژه‌های توسعه و نگهداری نرم‌افزار.	سازماندهی سیستم‌ها، زیرساخت‌ها و الزامات SQA.
مقصود استاندارد	«چگونه» اجرا شود.	«چه چیزی» باید به دست آورد.
هدف استاندارد	تضمین کیفیت یک پروژه نرم‌افزاری خاص.	تضمین کیفیت نرم‌افزار تأمین‌کننده و ارزیابی قابلیت فرایند نرم‌افزاری آن.
مثال‌ها	-IEEE 730 Standard -IEEE 828	-SEI CMMI -ISO 25010

٤-٥-١- استانداردهای فرایند پروژه

استانداردهای این کلاس، بر روش‌های انجام پروژه‌های توسعه و نگهداری نرم‌افزار تمرکز دارند و کیفیت آنها را تضمین می‌کنند. به‌عبارت دیگر، بر «نحوه» اجرای یک پروژه نرم‌افزاری تمرکز دارند. این استانداردها مراحلی که باید انجام شود، نیازمندی‌های مستندسازی طراحی، محتوای اسناد طراحی، بازبینی‌های طراحی و مسائل مربوط به بازبینی، آزمون نرم‌افزار که باید انجام شود، موضوعات آزمون و سایر موارد را تعریف می‌کنند. طبیعتاً به دلیل ویژگی‌هایشان، بسیاری از استانداردها در این دسته، می‌توانند به‌عنوان کتاب‌های درسی مهندسی نرم‌افزار و تضمین کیفیت نرم‌افزار استفاده شوند (Galin, 2018).

به‌طورکلی، استانداردهای تخصصی رهنمودهای روش‌شناسانه را برای تیم توسعه‌دهنده فراهم می‌کنند. با اینکه این استانداردها توضیح می‌دهند که چه مراحلی باید برداشته شوند و چه محصولاتی باید به‌عنوان بخشی از فرایند، توسعه داده شوند، اما هنوز استفاده از فنون مشخص (مانند روش‌ها و ابزارها) برای سازمان‌ها را باز می‌گذارند. هدف اصلی استانداردهای فرایند پروژه، تعریف

[1] Electronic Industries Alliance (EIA)

چارچوب مشترکی از بهترین شیوه‌ها است. این بهترین روش‌ها، در قالب فعالیت‌ها و زیرفعالیت‌هایی است که باید به‌عنوان بخشی از یک پروژه انجام شوند (Mellegård, 2013).

۴-۵-۱-۱- استاندارد IEEE 730

استاندارد IEEE 730-2014 برای فرایندهای تضمین کیفیت نرم‌افزار شامل وظایف ارزیابی است که به‌طور خاص شامل فرایندهای تأیید و اعتبارسنجی (V&V[1]) است. در ضمیمه E از استاندارد (IEEE Std 730:2014, 2014) برای راهنمای ویژه صنعت برای استفاده از IEEE 730-2014، تعریف تأیید نرم‌افزار برای صنعت تجهیزات پزشکی بیان شده است: «آزمون نرم‌افزار یکی از بسیاری از فعالیت‌های راستی‌آزمایی است، برای تایید اینکه خروجی توسعه نرم‌افزار نیازمندی‌های ورودی آن را برآورده می‌کند» (Oberhauser, 2023).

به‌طورکلی این استاندارد، قالب و محتوای یک طرح تضمین کیفیت نرم‌افزار را ارائه می‌دهد. فهرست محتویات یک طرح تضمین کیفیت نرم‌افزار[2] مطابق با استاندارد IEEE 730 در جدول ۲ نشان داده شده است (Karmakar et al., 2023).

جدول ۲ - فهرست محتویات یک SQAP مطابق با استاندارد IEEE 730 (Karmakar et al., 2023)

ردیف	فعالیت	زیربخش‌ها
۱	هدف و دامنه	
۲	تعاریف و کلمات اختصاری	
۳	اسناد مرجع	
۴	نمای کلی طرح SQA	۱.۴ سازماندهی و استقلال ۲.۴ ریسک محصول نرم افزاری ۳.۴ ابزار ۴.۴ استانداردها، رویه‌ها و قراردادها ۵.۴ تلاش، منابع و برنامه
۵	فعالیت‌ها، نتایج و وظایف	۱.۵ تضمین محصول ۱.۱.۵ ارزیابی برنامه‌ها برای انطباق ۲.۱.۵ ارزیابی محصول برای انطباق ۳.۱.۵ ارزیابی برنامه‌ها برای پذیرش ۴.۱.۵ ارزیابی چرخه حیات محصول برای انطباق ۵.۱.۵ اندازه گیری محصولات ۲.۵ تضمین فرآیند ۱.۲.۵ ارزیابی فرآیندهای چرخه حیات برای انطباق ۲.۲.۵ ارزیابی محیط‌ها برای انطباق ۳.۲.۵ ارزیابی فرآیندهای پیمانکار فرعی برای انطباق ۴.۲.۵ اندازه‌گیری فرآیندها ۵.۲.۵ ارزیابی مهارت و دانش کارکنان
۶	ملاحظات بیشتر	۱.۶ بازبینی قرارداد ۲.۶ اندازه‌گیری کیفیت ۳.۶ چشم‌پوشی‌ها و انحرافات

[1] Verificaton & Validation
[2] Software Quality Assurance Plan (SQAP)

		۴.۶ تکرار کار ۵.۶ ریسک برای انجام SQA ۶.۶ راهبرد ارتباطات ۷.۶ فرآیند عدم انطباق
۷	سوابق SQA	۱.۷ تجزیه و تحلیل، شناسایی، جمع‌آوری، ذخیره‌سازی، نگهداری و دفع ۲.۷ در دسترس‌بودن سوابق

۴-۵-۱-۲- استاندارد IEEE 828

در سال ۱۹۸۳، IEEE اولین استاندارد برای مدیریت پیکربندی نرم‌افزار[1] را منتشر کرد. استاندارد IEEE 828 برای برنامه‌های مدیریت پیکربندی نرم‌افزار که در سال‌های ۱۹۹۰، ۱۹۹۸ و ۲۰۰۵ بازنگری شد. آخرین نسخه در سال ۲۰۱۲ منتشر شد. استاندارد IEEE 828، حداقل نیازمندی‌ها را برای فرآیندهای مدیریت پیکربندی در سیستم‌ها و مهندسی نرم‌افزار تعیین می‌کند (Fahmy et al., 2020).

استاندارد IEEE 828، برای تهیه طرح مدیریت پیکربندی نرم‌افزار[2] تأیید شده است. هرگونه تغییر در نرم‌افزار باید مطابق با SCMP؛ تجزیه‌وتحلیل، مستند و تأیید شود. به‌طورکلی این استاندارد، محتوای یک طرح مدیریت پیکربندی نرم‌افزار را ارائه می‌دهد. این استاندارد (Karmakar et al., 2023) شامل موارد زیر است:

- ✓ مدیریت پیکربندی نرم‌افزار و محتویات SCMP را تعریف و توضیح می‌دهد و همچنین دستورالعمل‌هایی در مورد نحوه انجام این کار ارائه می‌دهد؛
- ✓ رابطه‌ای بین فعالیت‌های SCM و توسعه اجزای نرم‌افزار و یکپارچه‌سازی سیستم برقرار می‌کند. علاوه‌براین، فعالیت‌های کنترل تغییر پیکربندی کد و سند را شناسایی و توضیح می‌دهد؛
- ✓ استفاده از ابزارهای CASE را برای مدیریت پیکربندی توصیه می‌کند؛
- ✓ TOC را برای SCMP توصیه می‌کند.

IEEE 828، از استاندارد ISO 12207 نیز پشتیبانی می‌کند. حداقل الزامات قابل قبول برای مدیریت پیکربندی در هر دو سیستم و نرم‌افزار را بیان می‌کند. در نتیجه، برای همه کلاس‌ها اعمال می‌شود. استاندارد EEE 828، فعالیت‌های مدیریت پیکربندی را که باید در چه مرحله‌ای از چرخه حیات انجام شود و برنامه‌ریزی و منابع موردنیاز را توصیف می‌کند. مطابق استاندارد IEEE 828، هدف مدیریت پیکربندی به شرح زیر است که (Laporte & April, 2018):

- شناسایی و مستندسازی ویژگی‌های عملکردی و فیزیکی یک محصول، مولفه، خروجی یا یک سرویس؛
- کنترل هرگونه تغییری در این ویژگی‌ها؛
- ثبت و گزارش هر تغییر و وضعیت اجرای آن؛
- پشتیبانی از ممیزی محصولات، نتایج، سرویس‌ها یا مولفه‌ها برای تأیید انطباق با نیازمندی‌ها.

[1] Software Configuration Management (SCM)
[2] Software configuration management plan (SCMP)

٤-٥-٢- استانداردهای مدیریت کیفیت

استانداردهای این کلاس، بر روی توسعه نرم‌افزار سازمان و زیرساخت‌ها و الزامات مدیریت SQA تمرکز دارند و انتخاب روش‌ها و ابزارها را به سازمان واگذار می‌کنند. با رعایت استانداردهای مدیریت کیفیت، سازمان‌ها می‌توانند به‌طور پیوسته اطمینان حاصل کنند که محصولات نرم‌افزاری آنها به سطح قابل قبولی از کیفیت دست می‌یابند. برخی از مناقصه‌های توسعه نرم‌افزار فعلی، شرکت‌کنندگان را ملزم می‌کند که گواهینامه یکی از استانداردهای مدیریت کیفیت را داشته باشند (Galin, 2018).

این استانداردها راهنمای موارد زیر هستند:

- مدیریت توسعه نرم‌افزار؛
- نگهداری نرم‌افزار؛
- زیرساخت نرم‌افزار.

این استانداردها بر آنچه موردنیاز است تمرکز دارند، نه تصمیم‌گیری در مورد چگونگی دستیابی به آنها. سیستم مدیریت کیفیت، ارزیابی هدفمند از دستاوردهای سازمان را فراهم می‌سازد. سازمان‌هایی که نیازمندی‌های دستیابی به کیفیت را برآورده کرده‌اند، می‌توانند گواهینامه SQA را دنبال کنند.

استانداردهای مدیریت کیفیت، معمولاً حداقل دو هدف را دنبال می‌کنند. در مرحله اول، امکان ارزیابی بلوغ زیرساخت SQA یک سازمان؛ این معمولاً شامل صدور گواهینامه توسط یک نهاد معتبر خارجی است. چنین گواهینامه‌ای به سازنده تجهیزات اصلی[1] اجازه می‌دهد تا فرایند SQA تأمین‌کننده را ارزیابی کند و در نتیجه نشانه‌ای از توانایی سازمان برای ارائه محصولات باکیفیت بالا را ارائه می‌دهد. در مرحله دوم، استانداردهای مدیریت کیفیت را می‌توان برای راهنمایی سازمان در جهت بهبود سیستم SQA استفاده نمود (Mellegård, 2013).

۴-۵-۲-۱- استاندارد SEI CMMI

یکپارچه‌سازی مدل بلوغ قابلیت[2] برای توسعه، یک مدل مرجع است که فعالیت‌هایی را برای توسعه محصولات و خدمات پوشش می‌دهد (Sundaram & Suresh, 2023). موسسه مهندسی نرم‌افزار[3] در دانشگاه کارنگی ملون در پیتسبرگ، پنسیلوانیا، چارچوب یکپارچه‌سازی مدل بلوغ قابلیت را برای بهبود فرآیند در ایالات متحده ارائه کرده است. این مدل، مجموعه‌ای از بهترین شیوه‌ها برای بهبود نحوه عملکرد کارها است (Mona et al., 2023).

پروژه یکپارچه‌سازی CMM برای حل مشکل تکراری استفاده از چندین CMM ایجاد شده است. CMMI مدلی است که می‌تواند سه مدل منبع را ترکیب کند - اتحادیه صنایع الکترونیک/استاندارد موقت[4] ۷۳۱، مدل بلوغ قابلیت برای نرم‌افزار و مدل بلوغ قابلیت توسعه محصول یکپارچه[5]- در یک چارچوب بهبود واحد که می‌تواند رشته‌های مختلف زیادی را در خود جای دهد و انعطاف‌پذیری برای پشتیبانی از دو نمایش متفاوت (اعم از پیوسته یا مرحله‌ای) را دارد (Keshta, 2022).

---

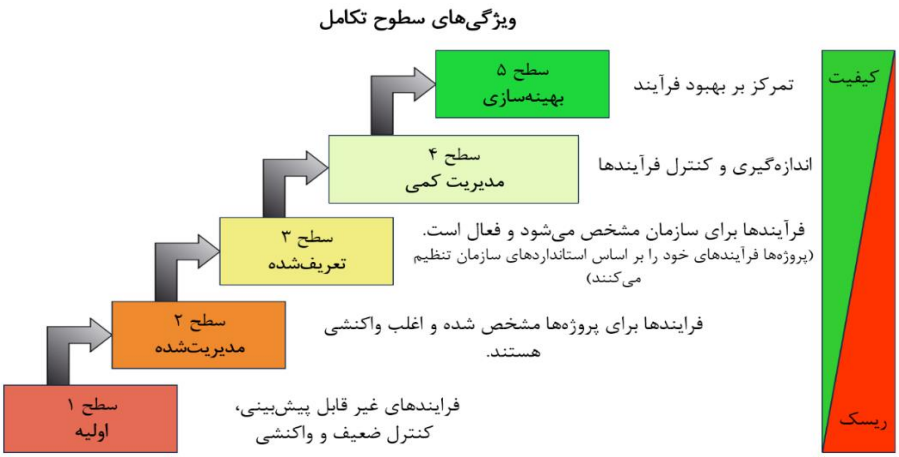
[1] Original Equipment Manufacturer (OEM)
[2] Capability Maturity Model Integration (CMMI)
[3] Software Engineering Institute (SEI)
[4] Electronic Industries Alliance/Interim Standard (EIA/IS)
[5] Integrated Product Development Capability Maturity Model (IPD-CMM)

در سال ۲۰۱۰، SEI نسخه ۱.۳ CMMI را منتشر کرد. این امر با استفاده از اطلاعات به‌دست‌آمده از تعدادی از مدل‌های معتبر و محبوب و همچنین دانش در مورد بهترین شیوه‌های توسعه نرم‌افزار و بهترین سیستم‌های سازمان‌های CMM با بلوغ بالا که بیش از ۱۰ سال از این مدل استفاده کرده بودند، ایجاد شد (Keshta, 2022).

مدل CMMI پنج سطح رشد[1] دارد که در شکل ۶ نشان داده شده است. سطح یک، پایین‌ترین سطح است که یک فرایند مدیریت نامناسب را نشان می‌دهد، درحالی‌که سطح پنج بالاترین سطح بوده و یک قابلیت مهندسی نرم‌افزار بسیار موثر و پیشرفته را نشان می‌دهد.

بیست و دو حوزه فرآیند[2] وجود دارد که به جز سطح یک، با سطوح دیگر مرتبط هستند. هر حوزه فرآیند دارای مجموعه‌ای از شیوه‌های مرتبط است که باید برای دستیابی به تعدادی از اهداف واضح تعریف‌شده، انجام شود (Keshta, 2022). برای اینکه یک شرکت توسعه نرم‌افزار به سطح خاصی از تکامل برسد، ابتدا باید تمام اهداف PA را در آن سطح خاص و همچنین هر سطح پایین‌تر، برآورده کند (Day et al., 2009).

هر سطح شامل چندین PA است، و برای هر یک از این PA، اهداف «عمومی» و «ویژه» شرح داده شده است که همه آنها متفاوت هستند. روش‌های مختلفی تحت هر هدف مشخص می‌شوند. این شیوه‌ها به سازمان کمک می‌کنند تا به‌طور کامل درک کند که چگونه می‌تواند به اهداف بلوغ خود دست یابد. آنها همچنین نمونه‌هایی از فعالیت‌هایی هستند که هر زمان که یک برنامه بهبود فرایند نرم‌افزار[3] انجام می‌شود، باید مورد توجه قرار گیرند (Keshta, 2022).



شکل ۶ - یکپارچه‌سازی مدل بلوغ قابلیت (CMMI)

CMMI یک چارچوب بهبود مهم و باارزش است که دارای PA، سطوح تعریف‌شده و اقدامات کلیدی است. بااین‌حال، CMM همانند نسل قبلی خود، نشان نمی‌دهد که شیوه‌های کلیدی چگونه باید اجرا شوند و همچنین توصیه نمی‌کند که چگونه سازمان‌ها می‌توانند راهبردهای پیاده‌سازی خود را بهبود بخشند (Clarke & O'Connor, 2013; Vivatanavorasin et al., 2006).

جدول ۳، نواحی فرآیند مرتبط با سطوح بلوغ را در نسخه مرحله‌ای از مدل توسعه CMMI نشان می‌دهد (Laporte & April, 2018).

[1] Maturity levels (MLs)
[2] process areas (PAs)
[3] Software Process Improvement (SPI)

۴-۵-۲-۲- استاندارد ISO/IEC 25010

ISO/IEC 25010 آخرین مورد از سری استاندارد ISO/IEC است که به‌عنوان ارزیابی و نیازمندی‌های کیفیت نرم‌افزار[1] شناخته می‌شود (Nuzula & Rochimah, 2023).

دیدگاه نظری SQuaRE، کیفیت سیستم‌های نرم‌افزاری را در زمینه فارنزیک (جمع‌آوری و بررسی شواهد) نرم‌افزار، ارزیابی کرده و یک مدل کیفیت در استفاده[2] ارائه می‌دهد (Sharma & Khaliq, 2024).

«کیفیت در استفاده»، شامل ویژگی‌هایی است که به نتیجه تعامل مربوط می‌شود. زمانی که یک محصول در زمینه استفاده خاصی استفاده می‌شود که برای سامانه کامل انسان - رایانه، از جمله رایانه و نرم‌افزار در حال استفاده، قابل استفاده است. «کیفیت محصول[3]»، شامل ویژگی‌هایی است که به نتیجه تعامل میان محصول و کاربر در یک زمینه خاص استفاده ارتباط دارد که قابل اعمال بر روی سامانه کامل انسان - رایانه است، شامل هم رایانه و هم نرم‌افزار در حال استفاده می‌باشد. هدف، ارزیابی ویژگی‌های کیفی واقعی محصول نرم‌افزاری است. علاوه‌براین، ازآنجایی‌که «کیفیت در استفاده» نیز ارتباط نزدیکی باکیفیت اطلاعاتی دارد که به کاربر منتقل می‌شود، مهم است که برخی از ویژگی‌های تعریف‌شده در استاندارد ISO/IEC 25012 و اقدامات مرتبطِ گزارش‌شده در استاندارد ISO/IEC 25024 در نظر گرفته شود (Santa Barletta et al., 2023).

جدول ۳ - نمایش مرحله‌ای از مدل CMMI برای توسعه (Laporte & April, 2018)

سطح	تمرکز	حوزه فرایند کلیدی	
۵ بهینه‌سازی	بهبود مستمر فرایند	مدیریت عملکرد سازمانی تجزیه‌وتحلیل و رفع علل	**کیفیت بهره‌وری**
۴ مدیریت کمی	مدیریت کمی	عملکرد فرایند سازمانی مدیریت کمی پروژه	
۳ تعریف‌شده	استانداردسازی فرایند	توسعه نیازمندی‌ها راه‌حل فنی یکپارچه‌سازی محصول تصدیق اعتبارسنجی تمرکز فرایند سازمانی تعریف فرایند سازمانی آموزش سازمانی مدیریت یکپارچه پروژه مدیریت ریسک تجزیه‌وتحلیل تصمیم و حل‌وفصل	
۲ مدیریت‌شده	مدیریت پروژه پایه	مدیریت نیازمندی‌ها برنامه‌ریزی پروژه نظارت و کنترل پروژه مدیریت قرارداد تأمین‌کننده اندازه‌گیری و تجزیه‌وتحلیل	

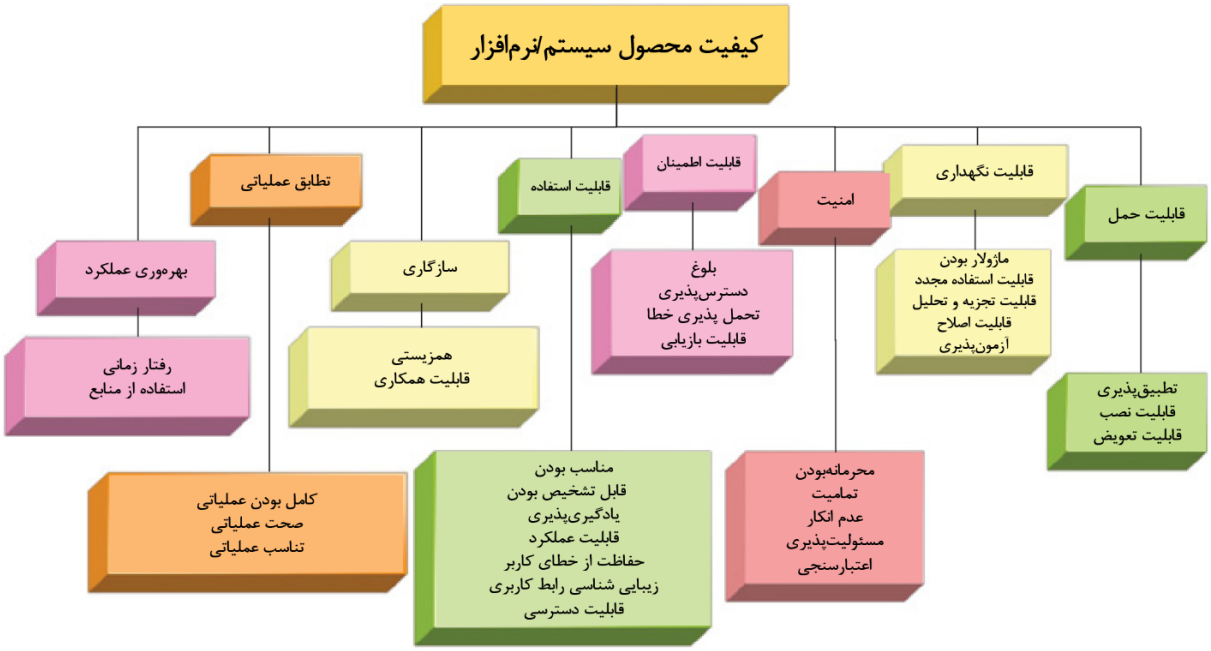
ریسک دوباره‌کاری

[1] Syste Quality Requirements and Evaluation (SQuaRE)
[2] In Us del
[3] Product quality

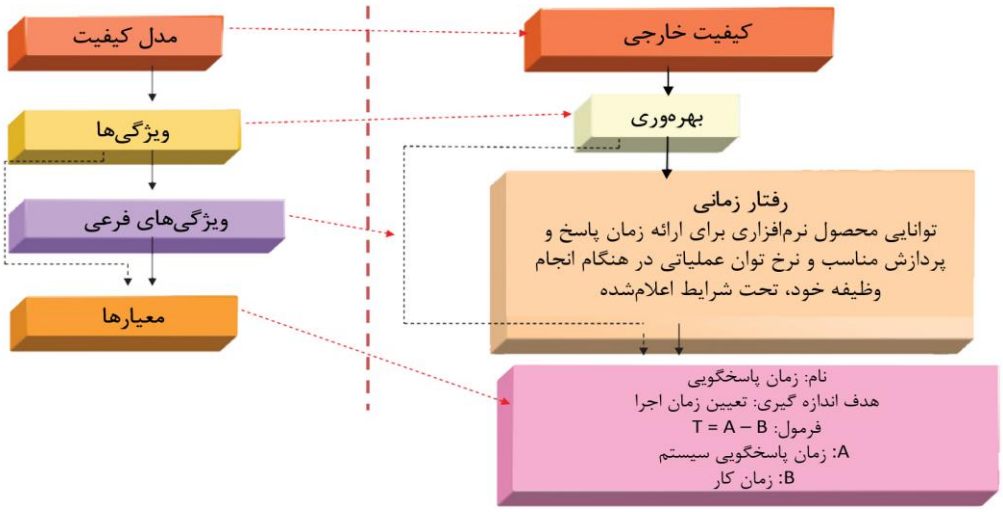
		تضمین کیفیت فرایند و محصول مدیریت پیکربندی	
۱ اولیه	-	-	

استاندارد بین‌المللی ISO/IEC 25010 مجموعه‌ای از ویژگی‌ها را پیشنهاد می‌کند که در هنگام اندازه‌گیری «کیفیت در استفاده» یک سیستم نرم‌افزاری باید در نظر گرفته شوند. استاندارد ISO/IEC 25010 دو مدل را برای اندازه‌گیری کیفیت تعریف می‌کند: استاندارد ISO 25010:2011 هشت ویژگی کیفیت را برای نرم‌افزار مشخص می‌کند، همان‌طور که در شکل ۷ نشان داده شده است. مدل کیفیت ارائه‌شده توسط ISO 25010 مشابه مدل IEEE 1061 است. یک مشخصه کیفیت را برای ارزیابی انتخاب کنید – بهره‌وری در مثال زیر (شکل ۸) استفاده شده است. سپس یک یا چند ویژگی فرعی کیفیت را برای ارزیابی انتخاب کنید (رفتار زمانی در مثال انتخاب شده است). آخرین مرحله این است که اندازه‌گیری را به وضوح مشخص کنید تا هیچ‌گونه تفسیر نادرستی از نتیجه آن وجود نداشته باشد. یک خط نقطه‌چین نشان می‌دهد که در صورت لزوم می‌توان از ویژگی‌های فرعی عبور کرد (Laporte & April, 2018).

این استاندارد ویژگی‌های مختلف کیفیت نرم‌افزار مانند قابلیت اطمینان، کارایی، امنیت و رضایت کاربر را پوشش می‌دهد. استاندارد ISO/IEC 25010:2023 شامل نه ویژگی کیفی است: مناسب‌بودن عملیات، بهره‌وری عملکرد، سازگاری، قابلیت تعامل، قابلیت اطمینان، امنیت، قابلیت نگهداری، انعطاف‌پذیری و ایمنی (Sopandi et al., 2024).



شکل ۷ – مدل کیفیت برای محصول نرم‌افزاری ISO 25010 (Laporte & April, 2018)



شکل ۸ – ارزیابی کیفیت نرم‌افزار با مدل ISO 25010 (Laporte & April, 2018)

این مدل می‌تواند برای تعیین الزامات و ارزیابی کیفیت محصولات هدف در طول چرخه حیات آنها توسط چندین ذینفع از جمله توسعه‌دهندگان، خریداران، کارکنان تضمین کیفیت و کنترل و ارزیاب‌های مستقل مورداستفاده قرار گیرد. فعالیت‌هایی در چرخه حیات محصول که می‌توانند از این مدل بهره‌مند شوند، عبارت‌اند از (ISO/IEC 25010:2023, 2023):

- ✓ استخراج و تعریف نیازمندی‌های سیستم اطلاعاتی و محصول؛
- ✓ اعتبار بخشیدن به جامعیت تعریف نیازمندی‌ها؛
- ✓ شناسایی اهداف طراحی سیستم محصول و اطلاعات و طراحی فرایند لازم برای دستیابی به کیفیت؛
- ✓ شناسایی اهداف آزمون محصول و سیستم اطلاعاتی؛
- ✓ شناسایی معیارهای کنترل کیفیت به‌عنوان بخشی از تضمین کیفیت؛
- ✓ شناسایی معیارهای پذیرش برای یک محصول و/یا یک سیستم اطلاعاتی؛
- ✓ ایجاد معیارهایی برای ویژگی‌های کیفیت محصول در حمایت از این فعالیت‌ها.

۴-۶- سازماندهی برای SQA – مولفه‌های نیروی انسانی

این مولفه‌ها شامل مدیران، کارکنان آزمون، واحد SQA و متخصصان علاقه‌مند به کیفیت نرم‌افزار می‌باشد. این افراد از طریق ابتکارات و پشتیبانی برای شناسایی انحرافات از رویه‌ها و روش SQA به کیفیت نرم‌افزار کمک می‌کنند و در نهایت اقدامات بهبود مناسب را پیشنهاد می‌کنند. می‌توان از رویکردهای مختلفی استفاده کرد؛ از جمله ایجاد واحد SQA، کمیته SQA یا انجمن SQA (Kamarudin, 2012).

مولفه‌های تضمین کیفیت نرم‌افزار نمی‌توانند در خلاء سازمانی اعمال شوند. این مؤلفه‌ها به یک پایه سازمانی نیاز دارند. این پایه سازمانی شامل مدیریت سازمان، پرسنل آزمون نرم‌افزار، واحد SQA و سایر علاقه‌مندان متخصص به کیفیت نرم‌افزار است. تمام این اشخاص به بهود کیفیت نرم‌افزار کمک می‌کنند. اهداف اصلی این پایه سازمانی به‌قرار زیر است (Galin, 2018):

- ✓ توسعه و پشتیبانی از پیاده‌سازی مولفه‌های SQA؛
- ✓ تشخیص انحرافات از رویه‌ها و متدولوژی‌های SQA؛
- ✓ پیشنهادهایی برای بهبود مولفه‌های SQA.

مولفه‌های نیروی انسانی شامل موارد زیر است:

- نقش مدیریت در SQA؛
- واحد SQA؛
- انجمن‌ها، کمیته‌ها و متولیان SQA.

۵- بحث و نتیجه‌گیری

حرکت به سمت تسلط بر تضمین کیفیت نرم‌افزار، از طریق اتخاذ بهترین شیوه‌ها و راهبردها با شناخت اهمیت فوق‌العاده آن در حوزه توسعه نرم‌افزار به اوج خود می‌رسد. این حرکت از طریق اصول بنیادی، راهبردهای ضروری و تحقق مزایای چند وجهی که پیاده‌سازی SQA قوی ارائه می‌دهد، پیموده می‌شود. نتیجه‌گیری بر نقش محوری SQA به‌عنوان یک پایه در چرخه حیات توسعه نرم‌افزار تأکید می‌کند که به‌عنوان نگهبان کیفیت محصول، قابلیت اطمینان و رضایت کاربر عمل می‌کند. با پایبندی مستمر به بهترین شیوه‌ها و راهبردهای نوآورانه، SQA به‌عنوان یک کاتالیزور برای موفقیت سازمانی و پیشرفت فناوری ظاهر می‌شود. یکی از شش کلاس در دسته‌بندی معماری SQA، عبارت است از: استانداردهای SQA، صدور گواهینامه و مولفه‌های ارزیابی سیستم SQA. این مؤلفه‌ها، استانداردهای مدیریتی و حرفه‌ای بین‌المللی را در سازمان پیاده‌سازی و اجرا می‌کنند. اهدافی که این گروه از مؤلفه‌ها دنبال می‌کنند شامل استفاده از دانش حرفه‌ای بین‌المللی، بهبود همکاری با سیستم‌های کیفیت سازمان‌های دیگر و سنجش و ارزیابی حرفه‌ای اهداف و دستاوردهای سیستم‌های کیفیت سازمان می‌باشد.

با توسعه و پیشرفت مستمر زمان، استاندارد کیفیت نرم‌افزار کمک شایانی به توسعه و بهبود مهندسی نرم‌افزار کرده و شرایط مساعدی را برای تضمین کیفیت نرم‌افزار ایجاد کرده است. نه‌تنها به استخراج روش‌های آزمون نرم‌افزار، ایجاد ابزارهای خودکار آزمون نرم‌افزار و بهبود دائمی سازوکار مدیریت فرایند آزمون نیاز داریم، بلکه توسعه مهندسی نرم‌افزار نیز باید با استانداردهای کیفیت نرم‌افزار مطابقت داشته باشد که به‌طور مداوم در جامعه امروزی در حال توسعه و به‌روزرسانی هستند. تنها در چنین سازوکار مهندسی نرم‌افزار کاملی، می‌توان نرم‌افزار باکیفیت مطلوب توسعه داد.

در این مقاله، با ارائه یک آرایه‌شناسی به بررسی مولفه‌های سیستم SQA و چهار استاندارد بین‌المللی رایج برای تضمین کیفیت نرم‌افزار پرداخته شد. ابتدا نیازمندی‌ها و اهمیت تضمین کیفیت نرم‌افزار در سامانه‌های نرم‌افزاری مورد بحث قرار گرفت. سپس در مورد هر یک از شش مولفه سیستم SQA و مزایای استفاده از استانداردهای متداول حوزه تضمین کیفیت نرم‌افزار برای رسیدن به کیفیت مطلوب، صحبت شد. آرایه‌شناسی مذکور، شامل دو دسته استانداردهای فرایند پروژه و استانداردهای مدیریت کیفیت بود. در دسته استانداردهای فرایند پروژه، استانداردهای IEEE 730 و IEEE 828 بررسی شدند. همچنین در دسته استانداردهای مدیریت کیفیت، استانداردهای SEI CMMI و ISO 25010 بررسی شدند.

به‌طورکلی، استفاده از سیستم تضمین کیفیت نرم‌افزار به همراه استانداردهای متداول این حوزه، به سازمان‌ها کمک می‌کند تا فرایندهای خود را استانداردسازی کنند، کیفیت نرم‌افزار را بهبود دهند، اعتماد مشتریان را بالا ببرند، رقابت‌پذیری خود را افزایش دهند و فرایندهای خود را بهبود بخشند. این مزایا در نهایت بهبود عملکرد سازمان و ارائه نرم‌افزارهای باکیفیت و قابل اعتماد به مشتریان را ممکن می‌سازند. لازم به ذکر است که هر سازمان باید به نیازها و شرایط خاص خود توجه کند و استانداردهای مناسبی را برای تضمین کیفیت نرم‌افزار خود انتخاب کند. همچنین، مدیران و تیم‌های توسعه نرم‌افزار باید به‌طور مداوم مولفه‌های سیستم SQA و استانداردها را مورد بررسی و ارزیابی قرار داده و در صورت نیاز، تغییرات لازم را ایجاد کنند.

٦- مراجع

Aarts, F. M., & Vos, E. (2001). The impact of ISO registration on New Zealand firms' performance: a financial perspective. *The TQM magazine*, *13*(3), 180-191.

Al-Qutaish, R. E. (2010). Quality models in software engineering literature: an analytical and comparative study. *Journal of American Science*, *6*(3), 166-175.

Atoum, I., Baklizi, M. K., Alsmadi, I., Otoom, A. A., Alhersh, T., Ababneh, J., Almalki, J., & Alshahrani, S. M. (2021). Challenges of software requirements quality assurance and validation: A systematic literature review. *Ieee Access*, *9*, 137613-137634.

Bhanushali, A. (2023). Ensuring Software Quality Through Effective Quality Assurance Testing: Best Practices and Case Studies. *International Journal of Advances in Scientific Research and Engineering*, *26*(1).

Clarke, P., & O'Connor, R. V. (2013). An empirical examination of the extent of software process improvement in software SMEs. *Journal of Software: Evolution and Process*, *25*(9), 981-998.

Day, B., Ke-Zun, S. C., Lovelock, L., & Lutteroth, C. (2009). Climbing the ladder: CMMI level 3. 2009 IEEE International Enterprise Distributed Object Computing Conference,

Fahmy, S., Deraman, A., Yahaya, J., Nasir, A., & Shamsudin, N. (2020). The evolution of software configuration management. *International Journal*, *9*(1.3).

Galin, D. (2018). *Software quality: concepts and practice*. John Wiley & Sons.

IEEE Std 730:2014. (2014). IEEE standard for software quality assurance processes. In (pp. 1–138). (Revision of IEEE Std 730-2002).

ISO/IEC 25010:2023. (2023). Systems and software engineering-Systems and software Quality Requirements and Evaluation (SQuaRE)-Product quality model. In (pp. 1-22). International Organization for Standardization (ISO).

Jamwal, D. (2010). Analysis of software quality models for organizations. *International Journal of Latest Trends in Computing*, *1*(2), 19-23.

Kamarudin, S. a. (2012). Architecting Software Quality: A Foundation For Software Productivity. proceedings intl conf information system business competitiveness,

Karmakar, G., Wakankar, A., Kabra, A., & Pandya, P. (2023). *Development of Safety-Critical Systems: Architecture and Software*. Springer Nature.

Keshta, I. (2022). A model for defining project lifecycle phases: Implementation of CMMI level 2 specific practice. *Journal of King Saud University-Computer and Information Sciences*, *34*(2), 398-407.

Korchenko, A., Breslavskyi, V., Yevseiev, S., Zhumangalieva, N., Zvarych, A., Kurchenko, O., Laptiev, O., & Tkachuk, S. (2021). Development of a method for constructing linguistic standards for multi-criteria assessment of honeypot efficiency.

Kumar, A., & Gupta, D. (2017). Paradigm shift from conventional software quality models to web based quality models. *International Journal of Hybrid Intelligent Systems*, *14*(3), 167-179.

Laporte, C. Y., & April, A. (2018). *Software quality assurance*. John Wiley & Sons.

Laporte, C. Y., Verret, G., & Muñoz, M. (2023). A software project that partially failed: A small organization that ignored the management and technical practices of software standards. *Computer*, *56*(5), 138-144.

Mellegård, N. (2013). *Improving Defect Management in Automotive Software Development, LiDeC—A Light-weight Defect Classification Scheme*. Chalmers Tekniska Hogskola (Sweden).

Miguel, J. P., Mauricio, D., & Rodríguez, G. (2014). A review of software quality models for the evaluation of software products. *arXiv preprint arXiv:1412.2977*.

Mona, J., Al-Sagheer, R. H. A., & Alghazali, S. M. (2023). Software Quality Assurance Models and Application to Defect Prediction Techniques. *International Journal of Intelligent Systems and Applications in Engineering*, *11*(1), 169–178-169–178.

Ndukwe, I. G., Licorish, S. A., Tahir, A., & MacDonell, S. G. (2023). How have views on software quality differed over time? Research and practice viewpoints. *Journal of systems and software*, *195*, 111524.

Nuzula, M. I. F., & Rochimah, S. (2023). Evaluation of Service Quality in Human Resource Information Systems Using the ISO/IEC 25010. 2023 International Seminar on Application for Technology of Information and Communication (iSemantic),

Nyári, N., & Kerti, A. (2021). Review of software quality related iso standards. *Biztonságtudományi Szemle*, *3*(2), 61-72.

Oberhauser, R. (2023). VR-V&V: Immersive Verification and Validation Support for Traceability Exemplified with ReqIF, ArchiMate, and Test Coverage. *International Journal on Advances in Systems and Measurements*, *16*(3 & 4), 103-115.

Santa Barletta, V., Caivano, D., Colizzi, L., Dimauro, G., & Piattini, M. (2023). Clinical-chatbot AHP evaluation based on "quality in use" of ISO/IEC 25010. *International Journal of Medical Informatics*, *170*, 104951.

SenthilMurugan, C., & Prakasam, S. (2014). IMPLEMENTING QUALITY PROCESS IN DEVELOPING PHASES. *Indian Journal of Communications Technology and Electronics (IJCTE)*, *2*(2), 75-80.

Sharma, S. K., & Khaliq, M. (2024). Design and development of software quality forensics framework and model. *Multidisciplinary Science Journal*, *6*(7), 2024111-2024111.

Silega, N., Castro Aguilar, G. F., Alcívar, I. M., Faggioni, K. M., Rogozov, Y. I., & Lapshin, V. S. (2023). An ontology-based approach to support the knowledge management of software quality standards. *Enfoque UTE*, *14*(3), 49-56.

Silva Farias, R., Ahmed, I., & Santana de Almeida, E. (2024). What Makes a Great Software Quality Assurance Engineer? *arXiv e-prints*, arXiv: 2401.13623.

Sopandi, A., Yahaya, N. A., & Subiyakto, A. (2024). Developing the Readiness and Success Model of Information System Implementation in the Indonesian Equestrian Industry. *Journal of Applied Data Sciences*, *5*(1), 133-145.

Sundaram, S. K., & Suresh, M. (2023). Partial CMMI V2. 0 Assessment Using Multi-grade Fuzzy for Healthcare and Insurance Segment in Software Services. In *Intelligent Manufacturing and Energy Sustainability: Proceedings of ICIMES 2022* (pp. 401-412). Springer.

Thapar, S. S., Singh, P., & Rani, S. (2012). Challenges to development of standard software quality model. *International Journal of Computer Applications*, *49*(10).

Vivatanavorasin, C., Prompoon, N., & Surarerks, A. (2006). A process model design and tool development for supplier agreement management of CMMI: Capability Level 2. 2006 13th Asia Pacific Software Engineering Conference (APSEC'06),

Wong, W. Y., Sam, T. H., Too, C. W., & Pok, W. F. (2022). Software quality assurance plan: Setting quality assurance checkpoints within the project life cycle and system development life cycle. 2022 IEEE 18th International Colloquium on Signal Processing & Applications (CSPA),

Zhao, Y., Hu, Y., & Gong, J. (2021). Research on International Standardization of Software Quality and Software Testing. 2021 IEEE/ACIS 20th International Fall Conference on Computer and Information Science (ICIS Fall),

Zimon, D. (2017a). The influence of quality management systems for improvement of logistics supply in Poland. *Oeconomia Copernicana*, *8*(4), 643-655.

Zimon, D. (2017b). Quality Management Systems' impact on the functioning of distribution channels in the FMCG market. *Calitatea*, *18*(156), 52.

Zimon, D., & Dellana, S. (2020). A longitudinal exploratory study of ISO 9001 certification abandonment in small- and medium-sized enterprises. *International Journal of Quality & Reliability Management*, *37*(1), 53-67.

Zimon, G., Habib, A. M., & Haluza, D. (2024). Does the quality management system affect working capital management efficiency? Evidence from Polish firms. *Cogent Business & Management*, *11*(1), 2292787.

Software quality assurance system and common standards

Ali Karimi
Assistant Professor of Imam Hossein Comprehensive University

Ali Tolui far[1]
MSc. Student of Imam Hossein Comprehensive University

1-1-

2-1- Abstract

The software development and maintenance environment is directly affected by software quality assurance components to create error-free software systems. The unique nature of Software production compared to other industrial products creates the need for different quality approach and tools for software development and maintenance. On the other hand, constant changes in business context and software development make it important to understand how software quality assurance should respond. Software quality assurance is a critical issue that is necessary for the success of software projects. The software quality assurance system expresses an Software quality assurance architecture that can be classified into six classes. The Software quality assurance architecture is an approach to manage and improve quality throughout the software project lifecycle. Although a wide range of software quality assurance models are available, but six software quality assurance system components which form the foundation for software quality system, along with four widely used global standards, will be discussed in this paper. Each of these components deal with different software error sources to provide an acceptable level of software quality. This article helps developers, project managers and software quality assurance enthusiasts to increase software quality by taking advantage of software quality assurance architecture components and using standards and ensure the success of their projects by improving productivity, attracting customers trust and increasing competitiveness in organizations.

3-1- Keywords: Software quality assurance system, Software development, Software quality assurance standards, Software quality assurance architecture.

1-Corresponding Author