هفتمین کنفرانس بین‌المللی
مهندسی قابلیت اطمینان و ایمنی

روش‌های اندازه‌گیری قابلیت اطمینان نرم افزار

مهرداد برون[۱]

۱- پژوهشگر مستقل، لیسانس فناوری اطلاعات، mehrdadboroon95@gmail.com

چکیده

قابلیت اطمینان نرم افزار یکی از فاکتورهای مهم در تعیین کیفیت نرمافزار محسوب میشود. تا کنون مدلهای زیادی جهت کمّی کردن و اندازه-گیری این فاکتور مهم و حیاتی معرفی شدهاست. اکثر این مدلها پیچیده بوده و بیشتر در محیطهای آکادمیک و آزمایشگاهی مورد استفاده قرار میگیرند. در این مقاله علاوه بر ارائه دستهبندیهای مختلفی از این مدلها، مفروضات و ویژگیهای هر یک از آنها مشخص میشود؛ به نحوی که با دادن نظم به آنها میتوان به سمت راحتتر کردن استفاده از آنها پیش رفت. از بین مدلهای ارائه شده، مدلهای رشد قابلیت اطمینان نرمافزار بیشترین مقالات و پژوهشها را به خود اختصاص دادهاند و ما نیز در این مقاله آنها را بیشتر از دیگر مدلها مورد بحث قرار میدهیم. هدف از این مقاله، ارائهی یک بازبینی ازروشها و مدلهای اندازه-گیری قابلیت اطمینان نرمافزار میباشد؛ به ویژه آن دسته از مدلهایی است که در فازهای نهایی چرخهی حیات نرمافزار، قابلیت اطمینان را تخمین میزنند (که به آنها مدلهای تخمین قابلیت اطمینان نرمافزارنیز گفته میشود). سپس در پایان الگوریتمی برای انتخاب مدل مناسب ارائه میگردد.

واژه‌های کلیدی: انتخاب مدل، پروفایل عملیاتی، قابلیت اطمینان، مدلهای رشد قابلیت اطمینان نرم‌افزار، مدلهای قابلیت اطمینان نرم افزار.

مقدمه

در حال حاضر نرم افزار به عنوان یکی از اصلیترین بخش-های یک سیستم محسوب میشود و یکی از کلیدیترین نقشها را در سیستم ایفا میکند؛ به طوریکه این نقش روز به روز در حالا پررنگ شدن میباشد. یکی از فاکتورهای مهم هر نرمافزاری، کیفیت آن نرمافزار است و همچنین از مهمترین فاکتورهای کیفیت، قابلیت اطمینان آن نرمافزار میباشد [1- 3].

بر اساس استاندارد IEEE 729 که در سال 1991 ارائه شد، قابلیت اطمینان نرمافزار عبارتست از احتمال بدون شکست بودن عملکرد سیستم برای یک بازهی زمانی معین و در یک محیط معین. اگر قابلیت اطمینان را با$R(t)$نمایش دهیم، بر اساس این تعریف، $R(t)$ یک سیستم، تابعی است از زمان که به صورت یک احتمال شرطی تعریف میشود که سیستم وظایف خود را در بازهی زمانی$[t_0,t_1]$به درستی انجام خواهد داد، با این شرط که در زمانt_0سیستم وظایف خود را به درستی انجام میدادهاست.

مهندسی قابلیت اطمینان نرمافزار[1]یکی از مقولههایی است که به آن توجه ویژهای شدهاست. بر اساس یک مطالعهی قاعده-مند [4]، قابلیت اطمینان نرمافزار یکی از موضوعات به شدت فعال و پویا در مهندسی نرمافزار میباشد و تا کنون مقالات زیادی در این حوزه نگارش شدهاست. همچنین بر اساس این تحقیق قاعدهمند، بیشترین کارهای صورت گرفته در این زمینه مربوط میشود به ارائهی یک مدل جدید برای اندازهگیری قابلیت اطمینان و یا ارتقای مدلهای ارائه شدهی قبلی (63% درصد از کارهای انجام شده در این حوزه).

معیارهای متفاوتی برای پیشبینی و یا تخمینقابلیت اطمینان یک نرمافزار وجود دارد. در مدلهایی که در فازهای اولیه چرخهی حیات نرمافزار برای پیشبینی قابلیت اطمینان نرمافزار استفاده میشود (مدلهای پیشبینی که در قسمت 5-1) (2- در مورد آنها توضیح داده خواهد شد)، از معیارهایی نظیرنوع کاربرد نرمافزار، محیط توسعه و خصیصههای نرمافزاراستفاده میشود[2، 5]. این در حالی است که در مدلهایی که در فازهای نهایی چرخهی تولید نرم افزار استفاده میشوند (مدلهای تخمینی که در قسمت (2-5-2) به آنها پرداخته خواهد شد) از دادههای شکست و همچنین اشکالات باقیمانده[2]در نرمافزار، برای اندازهگیری قابلیت اطمینان استفاده می‌کنند [1، 2، 5، 6].

دستهبندی مدلهای موجود بـر اسـاس معیارهـای مختلـف باعث میشود که بتوان با توجه به سطحی از قابلیت اطمینان کـه برای نرمافزار مطلوب است، از کاراترین و مناسبترین مدل موجود استفاده نمود.
در واقع این دستهبندیها یک دید روشن از روش-های موجود ارائه خواهد دادکه در نتیجه میتوان به ویژگـیهـا و مفروضات مشترک خاص هر دسته پی برد و در عمل نیز بتوان از این مدلها با توجه به ویژگیهای نرمافزار و نیز میـزان بودجـهی تعریف شده به نحو کارایی استفاده کرد.
با توجه به هدف این مقاله که ارائهی یک بازبینی از مدلهـا اندازهگیری قابلیت اطمینان نرمافزار است، در ابتدا در بخش 2 به معرفی چند نمونه از پر استنادترین و پرکاربردترین دستهبنـدی-های موجود خواهیم پرداخت و در بخش 3 نحوهی ایجاد پروفایل عملیاتی را شرح خواهیم داد، در بخـش 4 نیـز بـا الهـامگیـری از تحقیقات قبلی، الگوریتمی جهت انتخاب مدل مناسب ارائه مـی-دهیم. در بخش پایانی نیز بـه بیـان نتیجـهگیـری و چـالشهـا و همچنین مسائل بازی که در این حوزه وجود دارد میپردازیم.

دستهبندی مدلهای اندازهگیری قابلیت اطمینان نرم-افزار

مدلهای اندازهگیری قابلیت اطمینان نـرمافـزار تـا کنـون از منظرهای مختلفی دستهبندی شـدهانـد. در ادامـه چنـد مـورد از دستهبندیهای پرکاربرد مورد بررسی قرار خواهد گرفت.

دستهبندی موسا و اکیوموتو

یکی از اولین دسته بندیهای ارائـه شـده بـرای مـدلهـای قابلیت اطمینان نرمافزار، توسـط موسـا و اکیوموتـو ارائـه شـده-است[1]. این ردهبندی، بر اساس 5فـاکتورِحوزهی زمـان (زمـان تقویمی[3] و یا زمان اجرا در واحد پردازش مرکزی[4])، دسته[5](تعداد شکستهای محدود و یا نامحدود)، نوع توزیع (توزیع پواسن و یـا توزیع دو جملهای)، کلاس (این فاکتور فقط مخصوص مدلهـای دستهی تعداد شکستهای محدود اسـت و بـر مبنـای آن شـکل تابعی شدت شکست بر حسب زمان بیـان مـیشـود) و خـانواده (این فاکتور فقط مخصوص مدلهای دستهی تعداد شکستهـای نامحدود است و بر مبنـای آن شـکل تـابعی شـدت شکسـت بـر حسب تعداد شکستهای مورد انتظار بیـان مـیشـود)مـیباشـد. اهمیت این دستهبندی به خاطر این است که جزء اولـین دسـته-بندیهاست و مطالعات و پژوهشهـایی کـه بعـداً در ایـن حـوزه انجام شدهاست از آن باز خورد زیادی گرفتهاند [1].

دستهبندی لیو

در مرجع [1]،بازبینی مفصلی از مـدلهـای تخمـین قابلیـت اطمینان نرمافزار ارائه شدهاست؛ دسته بندی ارائه شده برای این مدلها به شرح زیر است:

مدلهای ردهی زمان شکست نمایی[6] مدلهای ردهی زمان شکست گاما و ویبول[7] مدلهای شکست نامحدود[8] مدلهای بیزین[9]
بر اساس این منبع، مدلهـای رده ی زمـان شکسـت نمـایی جزء مهمترین مدلها هستند که تا آن زمان بیشـترین مقـالات و پژوهشها در مورد آنها بودهاست. معروفترین مدلهای موجـود در این دسته مدلجِی-ام [10]، مدل فرآینـد نامتجـانس پواسـن[11]، مـدل زمـان اجـرای پایـه موسـا، مـدل فـوق نمـایی[12]و مـدل اشنیدویند[13]میباشد. در این مدلها تابع شدت شکست به صورت نمایی است.
در مدلهای ردهی زمان شکست گاما و ویبول، همانطور کـه از نامشان پیداست، برای توصیف تابع شـدت شکسـت، بـه جـای توزیع نمایی از توزیع گاما و ویبول اسـتفاده مـیشـود. دو توزیـع گاما و ویبول به خاطر انعطافپذیری بـالایی کـه در مـدل کـردن شکستها فراهم میکنند، از اهمیت ویژهای برخوردارند. معروف-ترین مدلهای موجود در این دسـته عبارتنـد از: مـدل ویبـول و مدل رشد قابلیت اطمینان S شکل.
مدلهای ردهی شکست نامحدود نیز مدلهایی هسـتند کـه تعداد شکستها در آنها محدود نیست و در زمان بینهایـت تـابع متوسط تعداد شکستها در آنها به سمت بینهایت میل میکنـد. به این معنا کـه نـرمافـزار هـیچ وقـت بدون اشکال نخواهد بـود و دلیـل آن هـم مـیتوانـد پیـدا شـدن اشکالات جدید در حین فرآینـد بازیـابی و تصـحیح خطـا باشـد. معروفترین مدلهـای موجـود در ایـن دسـته عبارتنـد از: مـدل دوان[14]، مدل هندسی، مدل پواسن لگاریتمی موسا و اوکیوموتو.
آخرین دسته از مدلها مربوط میشود به معرفی مـدلهـای بیزی. در مدلهای گذشته، قابلیـت اطمینـان نـرمافـزار تنهـا در صورتی تغییر میکند که یک خطا رخ دهد و در اکثر آنها تـاثیر خطاها یکسان در نظر گرفتـه مـیشـود. در مـدلهـای بیـزی در صورتیکه هیچ شکستی گزارش نشود نیز قابلیت اطمینـان تغییـر میکند (افزایش مییابد). در واقع در این مدلها قابلیت اطمینان، تابعی است از تعداد اشکالات کشف شده و تعداد عملیـات بـدون شکست نرمافزار. همچنین در این مدلها تاثیر خطاها یکسـان در نظر گرفته نمیشود و تاثیر خطاهـا بسـیار مهـمتـر از تعـداد آن-هاست [1، 2]. مدل معـروف در ایـن دسـته مـدل رشـد قابلیـت اطمینان لیتل وود – ورال است که برای اطلاع بیشـتر مـی تـوان به[1] مراجعه نمود.

دستهبندی فام

بر اساس ایـن دسـته بنـدی ([2] و [8])، کلیـه روشهـای تخمین قابلیت اطمینان نرمافزار را میتـوان بـه دو دسـته اصـلی تقسیم کرد؛

مدلهای قطعی

در این روشها، از هیچ پدیده و اتفاق تصادفی استفاده نمی-شود و شالودهی برنامه یعنی تعداد عملگرها و عملوندهای مجزای برنامه، تعداد

خطاها و همچنین تعداد دستورات ماشین در برنامه تحلیل میشود. دو تا از معمولترین مدلهای این گروه عبارتند از:متریکهای نرمافزاری هالستید[15](مدلی برای تخمین تعداد خطاها در نرمافزار بر پایهی تعداد واحد عملگرها و عملوندها) ومتریکپیچیدگی سایکلومتیکِ مک کاب[16](مدلی برای تعیین کران بالا برای تعدادتستها در یک برنامه).

مدل‌های احتمالی

این مدلها پدیدهی شکست را به صورت یک سری اتفاقات احتمالی با توجه به زمان یا میزان تلاش تست، شرح میدهند. مهمترین مدلهای موجود در این دسته عبارتند از: مدلهای مبتنی بر زمان بین شکستها[17]، مدل های شمارش تعداد شکستها[18]، مدلهای کاشت (تزریق) خطا یا اشکال[19]، مدلهای رشد قابلیت اطمینان[20]و سایر مدلها.

دسته‌بندی بر اساس نوع داده‌های مورد نیاز

نمونهی دیگر از دستهبندیهای موجود، مدل ها را بر اساس نوع دادههای مورد نیاز دسته بندی میکنند [3، 7]. دستهبندی مدلها به دو دستهی اصلی زیر میباشد:

مدل‌های تجربی

بر اساس[7]، مدلهای تجربی رابطه یا مجموعهای از روابط را بین معیارهای قابلیت اطمینان نرمافزار و متریکهای مناسب نرمافزار – مثل پیچیدگی برنامه – برقرار میکنند. که این متریکهای نرمافزاری خود از نتایج تجربی حاصل از اطلاعات گذشته که در دسترس هستند، استفاده میکنند. از مدلهای موجود در این دسته، عبارتند از : مدل میراندا[21]، مدل هالستید[22] و مدل اشنایدر[23].

به عنوان مثال، متریک نرمافزاری به کار گرفته شده در مدل میراندا، تعداد دستورالعملهای برنامه است و این مدل بین این متریک و تعداد شکستهای برنامه رابطه بر قرار میکند. در مدل هالستید نیز تعداد اعمال و نیز تعداد عملوندهای برنامه به عنوان متریک نرمافزاری جهت تخمین قابلیت اطمینان نرمافزار استفاده میشود. در مدل اشنایدر نیز با دانستن تعداد زیر برنامهها، تعداد کل هزاران دستورالعمل منابع کد شدهی نرمافزار و نیز میزان کل تلاش حرفهای صورت گرفته به بر حسب ماه و افراد میتوان قابلیت اطمینان را تخمین زد.

مدل‌های تحلیلی[24]

این مدلها در فازهای نهایی چرخهی حیات نرمافزار مورد استفاده قرار گرفته (فازهای تست و پیادهسازی) و با استفاده از اطلاعات شکست نرمافزار و مدل کردن آنها از طریق یک سری تئوریهای آماری/ اتفاقی[25] ، قابلیت اطمینان را برای یک نرمافزار تخمین میزنند. این دسته از مدلها خود به دو زیر گروه مدل–های پویا و مدلهای ایستا تقسیم میشوند.

مدل‌های پویا[26] یا مدل‌های رشد قابلیت اطمینان نرمافزار

در این مدلها، رفتار شکست نرمافزار وابسته به زمان است؛ به گونهای که با جلو رفتن تست و کشف اشکالات بیشتر، قابلیت اطمینانافزایش مییابد. در واقع بر اساس نوع فاکتور شکست مورد استفاده (زمان بین شکستها یا تعداد شکستها)، با پیش بردن عملیات تست، تعداد شکستهای باقیمانده یا فاصلهی زمانی بین شکستها کاهش مییابد. این دسته از مدلها بسته به نوع واحد دورهی کشف خطا، زمان اجرای نرمافزار(زمان تقویمی و یا زمان اجرا در واحد پردازش مرکزی) و یا تعداد موردهای تست اجرا شدهاست، خود به دو زیر گروه مدلهای زمان پیوسته[27] و مدلهای زمان گسسته[28] تقسیم میشوند [3].

مدل‌های ایستا[29]

در این دسته از مدلها، قابلیت اطمینان نرمافزار در فازهای ابتداییتری (مثل فاز طراحی و کدنویسی) اندازهگیری میشود. همچنین در این مدلها بر خلاف مدلهای پویا، رفتار شکست نرمافزار مستقل از زمان است. این مدلها خود به دو دستهی مدلهای مبتنی بر دامنهی شکست[30] و مدلهای مبتنی بر دامنهی داده[31]تقسیم میشوند.

دسته‌بندی سنتی و کلی بر اساس زمان استفاده از مدل

یکی دیگر از دستهبندی مدلهای قابلیت اطمینان نرمافزار، بر مبنای زمان کاربرد (فاز) است [5]. بر این مبنا، مدلها به دو دستهی اصلی مدلهای پیش بینی قابلیت اطمینان و مدلهای تخمین قابلیت اطمینان تقسیم میشوند.

مدل‌های پیشبینی قابلیت اطمینان نرمافزار[32]

مدلهای پیشگویی قابلیت اطمینان نرمافزار در اولین دوره–های حیات نرمافزار –که اطلاعات خرابی موجود نیست– ایجاد میشوند و با بررسی فرآیند طراحی، قابلیتاطمینان نرمافزار را پیشگویی میکنند [3]. از آنجایی که در این مقاله بیشتر بر روی تخمین قابلیت اطمینان تاکید داریم، بیشتر از این به موضوع پیش بینی نمیپردازیم و شما را برای مطالعهی بیشتر به منبع [2] ارجاع میدهیم.

مدل‌های تخمین قابلیت اطمینان نرمافزار[33]

روشهای تخمینی، روشهایی هستند که در فازهای تست و یا اجرای یک نرم افزار– که اطلاعات خرابی موجود است– مورد استفاده قرار میگیرند. یعنی نرمافزار باید به یک سطحی از بلوغ رسیدهباشد و دیگر

نیاز نباشد تغییر جدی و بزرگی روی آن انجام شود و قابلیت تست شدن را نیز داشتهباشد. تا کنون مدلهای زیادی برای تخمین قابلیت اطمینان نرمافزار ارائهشدهاست.

بر اساس یک سری خصوصیات مشترک، این مدلها را می‌توان به چهار دستهی اصلی دستهبندی کرد که عبارتند از: مدل‌های مبتنی بر زمان بین شکستها[34]، مدلهای شمارش تعداد شکستها [35]، مدلها کاشت اشکال[36] و مدلهای مبتنی بر دامنهی ورودی[37]. این مدلها نسبت به مدلهای پیشبینی، پر استنادتر هستند و در اکثر مقالات در مورد آنها صحبت شدهاست [2، 3، 5، 7]. در ادامه در مورد هریک از این مدلها توضیح مختصری به همراه نام معروفترین آنها آوردهخواهد شد. برای مطالعهی بیشتر در مورد مدلهای نامبرده شده میتوانید به منابع[2، 3، 5، 7]مراجعه نمایید.

مدلهای مبتنی بر زمان بین شکستها، با تخمین زمان بین شکستها، قابلیت اطمینان سیستم را اندازهگیری میکنند. از معروفترین مدلهای موجود در این دسته میتوان به مدلهای جلینیسکی- موراندا، مدل شیک- ولورتون، مدل اشکالزداییغیر ایدهآل جی‌او و مدل بیزین لیتلوود - ورال اشاره کرد. در مدل‌های شمارش تعداد شکستها، معیار اندازهگیری قابلیت اطمینان، تعداد اشکالات اتفاق افتاده در یک بازهی مشخص زمانی است. مدل نمایی شومن، مدل فرآیند پواسن نامتجانس جیول- اکیوموتو، مدل زمان اجرای موسا و مدل زمان اجرای پواسن لگاریتمی موسا، از معروفترین مدلهای موجود در این دسته هستند.در مدلهای کاشت اشکال نیز اندزهگیری قابلیت اطمینان نرمافزار به این صورت خواهد بود که بعد از عمل تست، با دانستن سهم اشکالات تزریقی، تعداد کل خطاهای ذاتی قابل تخمین خواهد بود. معروفترین مدل موجود در این دسته، مدل کاشت اشکال میل[38]است. در مورد مدلهای مبتنی بر دامنهی ورودی نیز در بخش (2 - 6 - 3) توضیح لازم آورده شدهاست.

دستهبندی بر اساس فاز چرخهی توسعهی نرم-افزار

در منبع [12]، یک دستهبندی بر مبنای فازهای چرخهی توسعهی نرمافزار برای مدلهای اندازهگیری قابلیت اطمینان نرم-افزار آورده شدهاست. اهمیت این دستهبندی به خاطر این است که نویسنده از آن برای معرفی الگوریتم انتخاب مدل استفاده میکند. جزئیات این الگوریتم در بخش 4 بیان خواهد شد. بر این اساس مدلهای قابلیت اطمینان نرمافزار بر مبنای فازی که می‌توانند در آن به کار گرفته شوند به 6 دسته تقسیم میشوند:

مدل‌های پیشبینی در فازهای اولیه

همانطور که از نامشان نیز مشخص است، این دسته از مدل-ها در فازهای اولیهی چرخهی توسعه نرمافزار مثل فازهای نیازمندیها، طراحی و پیادهسازی قابل استفاده هستند. در بخش (2-5-1) نیز در مورد این دسته از مدلها توضیح داده شد. معروفترین مدلهای موجود عبارتند از: مدلهای مبتنی بر فاز، مدل آزمایشگاه رم، مدل لِیغرا[39]، مدل پیش بینی موسا، مجموعه دادههای صنعتی، مجموعه دادههای تاریخی [12].

مدل‌های رشد قابلیت اطمینان نرمافزار

این دسته از مدلها در فاز تست نرمافزار قابل استفاده هستند. در بخشهای قبلی در مورد این دسته از مدلها توضیح دادهشد. همچنین در بخش (2-7-1) نیز توضیحات بیشتری در این زمینه ارائه خواهد شد.

مدل‌های مبتنی بر دامنهی ورودی

این دسته از مدلها در فاز اعتبار سنجی قابل استفاده است. در این مدلها، موردهای تست به صورت تصادفی از روی ورودیهای حاصل از پروفایل عملیاتی[40](در مورد این مفهوم در بخش 3 توضیح داده خواهد شد) که احتمال اتفاق افتادن آنها بیشتر است، بدست میآیند. سپس از روی تعداد شکستهای اتفاق افتاده،قابلیت اطمینان را تخمین میزنند. مفروضات مشترک بین این مدلها عبارتست از: توزیع پروفایل ورودی شناخته شدهاست، موردهای تست بصورت تصادفی انتخاب می‌شوند و دامنهی ورودی میتواند به کلاسهای هم ارزی تقسیم - شود. معروفترین مدلهای موجود در این دسته عبارتند از: مدل نلسون، مدل تی سوکالاس،مدل[41] وِیس و وِیوکر[42].

مدل‌های مبتنی بر معماری

این دسته از مدلها در فازهای طراحی، پیادهسازی و تست قابل استفاده هستند. در بخش (2-7-5) در مورد این مدلها توضیحاتی آورده شدهاست.

مدل‌های جعبه سیاه ترکیبی

این دسته از مدلها ویژگیهای مدلهای مبتنی بر دامنهی ورودی را با ویژگیهای مدلهای رشد قابلیت اطمینان ترکیب میکنند. این مدلها در فازهای تست و اعتبارسنجی قابل استفاده هستند. معروفترین مدل موجود در این دسته ، مدل‌های رشد قابلیت اطمینان مبتنی بر دامنهی ورودی میباشد.

مدل‌های جعبه سفید ترکیبی

این دسته از مدلها ترکیبی از ویژگیهای مدلهای جعبه سیاه و جعبه سفید را در نظر میگیرند. همچنین برای پیش بینی قابلیت اطمینان نرمافزار، ویژگیهای معماری نرمافزار را نیز در نظر میگیرند. این دسته از مدلها در فاز تست قابل استفاده هستند. معروفترین مدل موجود در این دسته عبارتست از: مدل مبتنی بر زمان/ساختار برای تخمین قابلیت اطمینان.

دسته بندی ارائه شده در منبع [4]

منبع [4]یکی از جدیدترین مطالعات انجام شده بر اساس یک چارچوب قاعده مند در مقولهی مدلسازی قابلیت اطمینان نرمافزار است. در این مقاله اطلاعات جالبی در زمینهی مدل–سازی قابلیت اطمینان و همچنین مقالات و پژوهشهای موجود در این زمینه گردآوری شدهاست. این مقاله پس از بررسی مدلها و دستهبندیهای موجود، به معرفی یک دستهبندی جدید می–پردازد. بر اساس این دستهبندی، مدلهای قابلیت اطمینان به 6 دستهی اصلی تقسیم میشوند که در ادامه آنها را به همراه شرح مختصری میآوریم.

مدل‌های رشد قابلیت اطمینان نرم‌افزار

در بخش (2-4-2-1) توضیح مختصری در مورد این مدلها داده شد. همچنین بر اساس منابع [2، 3، 4، 6]باید این نکته را در نظر گرفت که این دسته از مدلها به نرمافزار به صورت کلی و به اصطلاح جعبه سیاه نگاه میکنند و ساختار داخلی و معماری اجزاء نرمافزار را در نظر نمیگیرند. همچنین بر اساس منبع [2]، نمودار تعداد اشکالات کشف شده بر حسب زمان تست برای این مدلها میتواند مقعری شکل[43] و یا شکل حرف S[44]باشد.در جدول (1) انواع این مدلها بر اساس این رفتار آورده شدهاست که وجه مشترک آنها مجانبی بودن رفتار نمودار و در نتیجه محدود و متناهی بودن تعداد خطاهاست (زیرا در این نوع مدلها با گذر زمان تعداد خطاهای باقیمانده کاهش مییابد).

بر اساس آمارهایی که در مورد کارهای موجود در مدلسازی قابلیت اطمینان نرمافزار در منبع [4] گردآوری شدهاست، مدل–های رشد قابلیت اطمینان نرمافزار از سایر مدلهای دیگر پر استنادتر هستند و اکثر مقالات این حوزه را به خود اختصاص دادهاند. برای مثال از بین مدلهایی که در این دستهبندی ارائه شده، 47% از مقالات و کارهای انجام شده به این مدلها اختصاص دارد. لذا این دسته از مدلها از اهمیت ویژهای برخوردار هستند. برای مطالعهی بیشتر در این زمینه به منابع [3، 6]مراجعه فرمایید.

جدول 1: انواع مدلهای رشد قابلیت اطمینان نرمافزار

نام مدل	نوع مدل	(متوسط تعداد شکستها در فاصلهی زمانی t)
Geol-Okumoto	مقعر	×
G–O S-shaped	Sشکل	×
Hossain Dahia	مقعر	×
Yamada exponential	مقعر	×
Yamada-Raleigh	S شکل	×
Weibull	مقعر	×
Kapur et al.	مقعر	×

روش‌های بیزین

در بخش (2- 2) در مورد برخی از مدلهای این دسته توضیحاتی دادهشدهاست. بر اساس آمار موجود در منبع [4]، بعد از مدلهای رشد قابلیت اطمینان، این دسته از مدلها 6% از کارها و پژوهشهای موجود در زمینهی مدلسازی قابلیت اطمینان را به خود اختصاص دادهاند. برای مطالعهی بیشتر در این زمینه منبع [9] مراجعه فرمایید.

مدلهای مبتنی بر تست

مدلهای مبتنی بر تست، تنها فاکتورهایی نظیر تعداد شکستهای اتفاق افتاده در یک بازهی زمانی مشخص و یا متوسط زمان بین شکستها را در نظر نمیگیرند؛ بلکه فاکتور دیگری نظیر میزان پوشش تست[45] را نیز به فاکتورهای گفته شده اضافه میکنند [10]. در واقع مدلهای مبتنی بر تست چگونگی استفاده از سیستم توسط کاربر را نیز در نظر میگیرند و موردهای تست را بر اساس پروفایل عملیاتی[46]سیستم انتخاب میکنند [4].

مدلهای قابلیت اطمینان نرمافزار که از معیار پوشش تست نیز استفاده میکنند، در سالها اخیر به طور گستردهای مورد مطالعه قرار گرفتهاند. یکی از چالشهایی که در این زمینه وجود دارد این است که

این مدلها بنا به نوع نرمافزار و بستر مورد استفاده، از متریکهای مختلفی (نظیر تعداد بلاکها، انشعابها) برای اندازهگیری میزان پوشش تست استفاده میکنند. حال آنکه نبود یک روش واحد که از همهی معیارهای گفته شده استفاده کند، احساس میشود. در واقع نیاز است تا این معیارها به طور یکپارچه و با هم برای مدلهای قابلیت اطمینان مبتنی بر میزان پوشش تست به کار گرفته شوند. در منبع [10] که در سال 2010 منتشر شدهاست، روشی بر این مبنا ارائه شدهاست.

مدلهای مبتنی بر هوش مصنوعی

مدلهایی که در آنها از تکنیکهای هوش مصنوعی مانند یادگیری ماشین، یادگیری ایستا، الگوریتمهای حریصانه، الگوریتم ژنتیک، منطق فازی و دیگر مفاهیم هوش مصنوعیاستفاده می‌شود، در این دستهبندی قرار میگیرندالبته. اکثراًمدلهای موجود در این دسته از مفهوم و تکنیک شبکههای عصبی استفاده میکنند.

مدلهای مبتنی بر ویژگیهای ایستا و معماری نرمافزار

این دسته از مدلها بر مبنای ملاحظات معماری و ویژگی‌های ایستای نرمافزار شکل گرفتهاند. در واقع این مدلها بین معیارهای مهندسی نرمافزار و قابلیت اطمینان نرمافزار رابطه بر قرار میکنند. این دسته از مدلها در فازهای اولیهی چرخهی حیات نرمافزار به کار گرفته میشوند، از این رو همانطور که قبلاً نیز در مورد مدلهای پیشبینی قابلیت اطمینان نرمافزار گفته شد، باعث صرفه جویی در هزینه میگردند.

مدلهای رشد قابلیت اطمینان نرمافزار در اصطلاح مدلهای جعبه سیاه نامیده میشوند؛ زیرا به نرمافزار به حالت کلی نگاه میکنند و سیستم را به حالت جزء به جزء در نظر نمیگیرند. در واقع مشکل اینگونه از سیستمها همین مطلب است. زیرا تنها رفتار سیستم را نسبت به محیط خارجی مدل میکنند و رفتار داخلی سیستم را در نظر نمیگیرند.در واقع اینگونه مدلها برای اندازهگیری قابلیت اطمینان یک سیستم بزرگ مبتنی بر جزء[47] مناسب نیستند [2]. بنابراین مدلهای مبتنی بر معماری، که برای اندازهگیری قابلیت اطمینان نرمافزار معماری اجزاء نرمافزار

را نیز در نظر میگیرند، برای مدل کردن قابلیت اطمینان سیستمهای گفته شده مناسب هستند. به مدلهای مبتنی بر معماری در اصطلاح مدلهای جعبه سفید نیز میگویند.

در منبع [11]، مهمترین مدلهای موجود در این دسته آورده شدهاست.براساس این منبع، مدلهای معماری به دو دستهی کلی مدلهای کمّی و مدلهای کیفی طبقهبندی می‌شوند. مدلهای کمّی که با اعداد و ارقام سر و کار دارند خود به سه دستهی اصلی مدلهای مبتنی بر حالت، مبتنی بر مسیر و مدلهای افزوده تقسیم میشوند. مدلهای کیفی بر خلاف مدل‌هایکمّی که با اعداد و ارقام سر و کار دارند، از طریق دانش[48]به تحلیل قابلیت اطمینان میپردازنداین. دانش معمولاً خاصِ سیستم تحت بررسی میباشد و میتواند عینی و بر گرفته از مستندات بوده و یا ضمنی و بر گرفته از فکر و نظرات کارشناسان باشد (که در این صورت به دلیل وابستگی بالا به انسان، مستعد خطاست). همچنین این دانش میتواند انتزاعی و کلی باشد یا مربوط به دامنه و کاربرد خاصی باشدالبته. اخیراً تمایل به سمت مستند کردن دانشهای کلی است؛ مثلاً از طریق شناسایی و استفاده از سبکها و الگوهای طراحی. البته در مورد سبکها و الگوهایی که تمرکز بر روی ویژگی قابلیت اطمینان دارند، با کمبود قابل ملاحظهای مواجه هستیم[11].

نکتهای که در مورد مدل های قابلیت اطمینان وجود دارد، این است که هیچ کدام از مدلهایی که تا کنون معرفی شدهاند به تنهایی برای پیشبینی قابلیت اطمینان کافی نیستند همچنین در صنعت به ندرت از آنها استفاده میشود [11]؛ از این رو هیچ استنباطی مبنی بر اینکه این روشها به اندازهی کافی کامل و مناسب هستند وجود ندارد. روشها و مدلهایی که در این زمینه وجود دارند بر روی تحلیل و آنالیز سیستم از طریق محاسباتمثلاً(متوسط زمان بین شکستها یا تعداد شکستها در یک بازهی زمانی) تمرکز دارند. حال آنکه هدف از به کارگیری این مدلها باید پر کردن شکاف بین معماری و نیازمندیها باشد. یعنی باید بتوان از طریق تحلیلی که آنها به ما ارائه میدهند رابطهی بین معماری مناسب یا نا مناسب را با نیازمندی مربوطه پیدا و ردیابی کرد.

سایر روشها

منظور از سایر روشها، مدلهایی است که از تکنیکهای تککاره[49] استفاده میکنند که این تکنیکها بر مبنای تئوری‌های خاص هستند (مانند بلاک دیاگرامهای قابلیت اطمینان، تحلیل ریسک، تحلیل مد شکست، تئوری قطعی هرج و مرج[50] و تئوری مدل ابری). یا روشهایی که برای موردهای خاصی شکل گرفتهاند (مثل مدل تک کاره برای یک سیستم خاص مبتنی بر برنامه نویسی چند نسخهای) یا روشهای مبتنی بر متریکهای مهندسی و محک گرفتن[51]یا تخمین قابلیت اطمینان در فازهای اولیه که بر اساس نیازمندیهای رفتاری سیستم است.

پروفایل عملیاتی

قابلیت اطمینان یک سیستم به نحوهی استفادهی کاربران آن سیستم بستگی دارد؛ در واقع این نحوهی استفادهی عملیاتی از یک سیستم است که فضای ورودی یک سیستم را مشخص میکند و استفاده از ورودیهای واقعی (ورودیهایی که در عمل کاربران از آنها بیشتر استفاده میکنند) در موردهای تست، لازمهی تخمین دقیق قابلیت اطمینان سیستم است. در اینجاست که مفهوم پروفایل عملیاتی مطرح میشود.

پروفایل عملیاتی از طریق تقسیم کردن فضای ورودی به یک تعداد کلاس یا مجموعهی متمایز ایجاد میشود. سپس بر اساس اینکه چقدر احتمال دارد که کلاسی انتخاب شود، به آن یک احتمال نسبت داده میشود. برای مثال اگر فرض کنیم که میتوانیم نمونههای تست را از بین دو کلاس (الف) و (ب) انتخاب کنیم، آنگاه نمونههای تست باید بر اساس احتمال وقوع هریک از این کلاسها، انتخاب شوندمثلاً. اگر احتمال کلاس (الف) دو برابر احتمال کلاس (ب) باشد، آنگاه احتمال اینکه موردهای تست (که به طور تصادفی انتخاب میشوند) متعلق به کلاس (الف) باشند، دو برابر بیشتر خواهد بود.

در واقع پروفایل عملیاتی، نحوهی استفاده از سیستم را در قالب ویژگیها و خصیصههای کمّی تعریف میکند. بنابراین جایگاه تعیین پروفایل عملیاتی در مهندسی قابلیت اطمینان ضروری است. پروفایل عملیاتی به ما کمک میکند که میزان بهرهوری و قابلیت اطمینان را افزایش دهیم. به این صورت که با هدایت فرآیند تست و نیز انتخاب موردهای تست از بین بخشی از فضای ورودی که احتمال انتخابشان بیشتر است، و نیز انتخاب موردهای تستی که باعث میشود پر استفادهترین عملیات سیستم بیشترین میزان تست را داشتهباشند، سطح اطمینان از قابلیت اطمینان اندازهگیری شده را به حداکثر میرساند [1].

برای ایجاد پروفایل عملیاتی باید 5 کار اساسی انجام شود؛ تعیین پروفایل مشتری، تعیین پروفایل کاربر، تعیین مدهای سیستم و مشخصات آنها، تعیین پروفایل تابعی (نیازمندیها) و نیز تعیین پروفایل عملیاتی (پیادهسازی). پروفایل عملیاتی از

طریق ایجاد لیستهای دقیقی از مشتریان، کاربران، مدهای سیستم، توابع و نیز عملیاتی که سیستم نیاز دارد که آنها را تحت هر مجموعه از موقعیتها (کلاسهایی که در مورد آنها صحبت کردیم) فراهم کند، به صورت سلسله مراتبی ایجاد می–شود. برای هریک از این بخشها، احتمال رخدادشان تخمین زده شده و در نتیجه یک توصیف کمّی از مشخصات (پروفایل) بدست میآید. برای اطلاعات بیشتر از نحوهی ایجاد پروفایلهای گفته شده و نیز مشاهدهی مثالهایی در این زمینه شما را به منبع [1]ارجاع میدهیم.

انتخاب مدل

انتخاب مدل برای استفاده در یک کاربرد و مورد مشخص، تا کنون توجهات زیادی را در مقولهی قابلیت اطمینان نرمافزار به خود جلب کردهاست. نویسنده[12]در ابتدا مدلهای قابلیت اطمینان نرمافزار را بر اساس فاز مورد استفاده دستهبندی میکند (در بخش (2-6) این دستهبندی آورده شدهاست). سپس به معرفی معیارهایی که از آنها برای انتخاب مدل استفاده میکند، پرداختهو پس از آن الگوریتمی جهت انتخاب مدل مناسب معرفی میکند.

معیارهای مورد استفاده برای انتخاب مدل

معیارهایی که برای انتخاب مدل، مورد استفاده قرار گرفتهاند عبارتند از:

فاز چرخهٔ توسعهٔ نرمافزار

بر این اساس مدل بر اساس این که در چه فازی قرار است مورد استفاده قرار بگیرد، انتخاب میشود؛ مثلاً اگر در فازهای اولیه توسعهی نرمافزار میخواهیم قابلیت اطمینان را پیشبینی کنیم دیگر نمیتوانیم از مدلهای رشد قابلیت اطمینان نرمافزار که در فاز تست قابل به کارگیری هستند، استفاده کنیم.

خروجی مطلوب کاربر

از معیارهای مهم جهت انتخاب مدل، خروجیِ مورد انتظار است. بر این اساس باید مدلی که میتواند خروجی مورد نظر را تولید کند، انتخاب شود. به عنوان مثال اگر خروجی مورد نیاز متوسط زمان تا شکست است[52]، دیگر نمیتوانیم از مدل فرآیند نامتجانس پواسن جیول و اکیموتو استفاده کنیم زیرا خروجی حاصل از این مدل متوسط تعداد شکستها است.

ورودی مورد نیاز مدل

در صورتیکه اطلاعات و دادههای مورد نیاز مدل فراهم نباشد، بدیهی است که نمیتوان از آن مدل استفاده کرد. به عنوان مثال مدلهای رشد قابلیت اطمینان نرمافزار به اطلاعاتی نظیر زمان شکستها نیاز دارند و در صورتیکه این اطلاعات فراهم نباشد نمیتوان از آنها استفاده کرد.

روند نمایش داده شده به وسیلهی اطلاعات

بر اساس این معیار، منحنی دادههای شکستِ جمعآوری شده، رسم شدهو با منحنی مدل مورد نظر مقایسه میشود. اگر هر دو منحنی مشابه بودند، شانس بیشتری وجود دارد تا مدل مورد نظر نتایج دقیقی تولید کند. به عنوان مثال اگر یک روند کلی نزولی در دادهها مشاهده میشود، انتخاب ما میتواند از بین مدلهای مقعری شکل باشد. اما اگر ابتدا یک روند صعودی و سپس نزولی مشاهده شود، باید یک مدل Sشکل انتخاب شود.

برآورده شدن فرضیات مدل

همانطور که قبلاً نیز اشاره شد، هر مدل فرضیات مربوط به خود را دارد. این فرضیات باید برآورده شوند تا بتوان از مدل مورد نظر استفاده کرد. البته احتمال اینکه تمام فرضیات مورد نظر برآورده شود کم است، اما در اغلب موارد باید اکثر فرضیات و یا مهمترین فرضیات مدل برآورده شود.

طبیعت پروژه

طبیعت پروژه شامل اینکه نرمافزار پایانپذیر هست یا خیر، اندازهی نرمافزار و مواردی از این قبیل میتواند باشد. همچنین سبکهای معماری مختلف (مانند سیستمهای تحملپذیر اشکال، فراخوانی و بازگشت، روش ترتیبی/خط لوله دستهای و روش فیلتر موازی/ خط لولهای) نیز میتواند به عنوان معیاری برای انتخاب مدل در نظر گرفته شود.

ساختار پروژه

اطلاعات ساختاری پروژه، میتواند به عنوان معیاری دیگر در جهت انتخاب مدل مناسب استفاده شود. در واقع اطلاعات ساختاری به ما میگویند که روابط بین ماژولها چگونه است. این معیار در نرمافزارهای مبتنی بر جزء قابل استفاده است. در واقع در صورتیکه بخواهیم از مدلی استفاده کنیم که به اطلاعات ساختاری اینچنینی نیاز داشتهباشد، باید نرمافزار و سیستم ما مبتنی بر جزء باشد.

فرآیند تست

نوع فرآیند تستی که از آن استفاده میکنیم نیز میتواند به عنوان معیاری در جهت انتخاب مدل مناسب باشدمثلاً. اگر سیستم ما مبتنی بر جزء است، دیگر تست کلی سیستم و نگاه کردن به سیستم به صورت یک واحد مناسب نیست و در نتیجه مدلی که برای تخمین قابلیت اطمینان سیستم نیز در نظر گرفته میشود نباید به سیستم به حالت کلی نگاه کند و باید مبتنی بر معماری باشد.

فرآیند توسعه

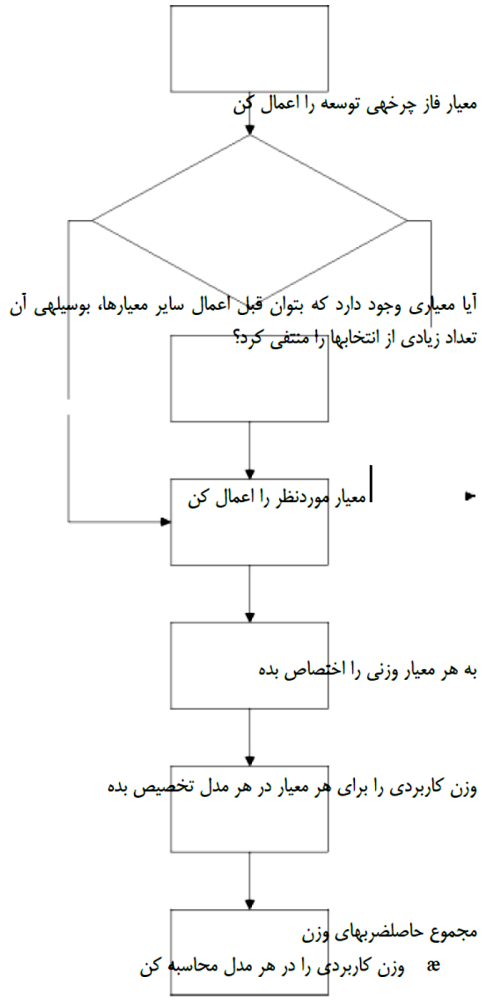
فرآیند توسعه نیز میتواند به عنوان فاکتوری برای انتخاب مدل محسوب شود؛ که خود شامل گزارش خطاها، روش مورد استفادهمثلاً(روش افزایشی) و مسائلی از این دست میباشد. به عنوان مثال اگر با یک تاخیر گزارش خرابی به دلیل تعیین مشکل مواجه هستیم و بین نقصها و اشکالات وابستگی دو جانبه وجود دارد، بهتر است که مدلهای S شکل برای اندازه-گیری قابلیت اطمینان نرمافزار استفاده شوند.

الگوریتم انتخاب مدل

پس از معرفی معیارهای انتخاب مدل، الگوریتمی جهت انتخاب مدل مناسب بر اساس معیارهایی که در قسمت قبل گفتهشد، ارائه میشود. این الگوریتم از 6 گام اصلی تشکیل شده-است که در ادامه هریک از آنها را توضیح میدهیم. در واقع روال کار به این صورت است که مدلها بر اساس معیارهای مورد نظر امتیاز میگیرند و در پایان هر مدلی که بالاترین امتیاز را داشته باشد، به عنوان مدل مناسب انتخاب میشود. در شکل (1) نمودار گردشی مربوط به این الگوریتم نشان داده شدهاست.

گام اول: در این گام اولین معیار انتخاب مدل یعنی «فاز چرخهی توسعهی نرمافزار» باید بررسی شود. در واقع خروجی این گام مدلهایی خواهد بود که در فاز مورد نظر ما (یعنی فازی که میخواهیم قابلیت اطمینان را در آن اندازهگیری کنیم) قابل استفاده باشند. در بخش (2-6)که دستهبندی مربوط به منبع [12] معرفی شد، مشخص کردیم که هر دسته از مدلها در چه فازی از چرخهی حیات نرمافزار قابل استفاده هستند.

گام دوم: در این مرحله به دنبال معیاری میگردیم که بتواند در فازی که در آن قرار داریم، قبل از بررسی سایر معیارها، تعداد زیادی از انتخابهای ممکن را حذف کندمثلاً. اگر ما در فاز طراحی قرار داریم، مدلهای قابل استفاده در این فاز دو دستهی مدلهای پیشبینی سریع (مدلهای جعبه سیاه).



شکل ۱: نمودار گردشی الگوریتم انتخاب مدل

نتیجه گیری

اکثر مدلهای اندازهگیری قابلیت اطمینان نرمافزار از فرمولها و محاسبات ریاضی و آماری پیچیدهای استفاده میکنند و این باعث پیچیدهشدن اندازهگیری قابلیت اطمینان شدهاست. اکثر کارهای صورت گرفته در زمینهی اندازهگیری قابلیت اطمینان، یا به ارائهی یک مدل جدید اختصاص داشتهاست و یا در راستای ارتقاء مدلهای قبلی بودهاست، که این خود نه تنها از پیچیدگی این مقوله کم نکردهاست بلکه در برخی موارد باعث هرچه پیچیدهتر شدن این مقوله و اضافه شدن فرمولها و محاسبت پیچیدهی جدیدی نیز شدهاست [4]. از این رو ضرورت نیاز به مطالعات قاعده مند و نیز دستهبندی و طبقهبندی کردن این مدلها احساس میشود تا از پیچیدگی آنها کاسته شود و بتوان به سمت کاربردی کردن آنها و به کارگیری آنها در عمل پیش رفت.

از دیگر چالشهایی که در این زمینه وجود دارد این است که مدل واحدی وجود ندارد که از آن بتوان در همهی موقعیتها استفادهکرد، زیرا مفروضات هر مدل فقط برای یک موقعیت خاص در نظر گرفتهشده و در واقع هر مدل بر اساس نیاز در یک مساله و موقعیت خاص شکل گرفتهاست. تا کنون پژوهشها و مطالعات خوبی در زمینهی انتخاب مدل و اینکه چه مدلی با توجه به فاکتورهای مورد نظر ما میتواند برای کار ما مناسب باشد ارائه شدهاست که یکی از پژوهشهای پر استناد در این زمینه منبع [12] میباشد. اما همچنان ضرورت یکپارچهسازی مدلها و رسیدن به یک مدل واحد که مستقل از کاربرد باشد، احساس میشود و از ملزومات رسیدن به این هدف بررسی و بازبینی کامل مدلهای ارائهشده و نیز انجام تحقیقات قاعده مند مثل منبع [4] میباشد.

اکثر مدلهای اندازهگیری قابلیت اطمینان نرمافزار از دادهها و اطلاعات شکست نرمافزار (در زمان تست) استفاده میکنند و تنها مدلهای پیشبینی قابلیت اطمینان نرمافزار هستند که از معیارهای دیگری مثل پارامترهای نرمافزار استفاده میکنند. در واقع مدلهای پیشبینی قابلیت اطمینان، از آنجایی که در فازهای اولیه به کار گرفتهمیشوند، باعث صرفهجویی در هزینه–های بعدی (مثلا هزینههای مربوط به تست و بازیابی اشکال) خواهند شد اما تعداد این مدلها نسبت به مدلهای تخمینی کم است [5]. لذا نیاز است به این گونه مدلها (در راستای صرفه جویی در هزینهها) بیشتر پرداخته شود.

همچنین در منابعی که در این مقاله از آنها استفاده شده–است هیچکدام به طور صریح، مزایا و معایب هر مدل را بررسی نکردهاند؛ شاید دلیل آن این باشد که این مدلها بیشتر در محیطهای آکادمیک و آزمایشگاهی مورد استفاده قرار گرفتهاند تا در دنیای واقعی؛ از این رو نیاز است تا این مدلها به سمت استفاده در عمل و صنعت پیش بروند تا مزایا و معایب هر کدام از آنها در عمل معلوم شود. بر اساس مطالعهی صورت گرفته در منبع [4] از بین کارهای انجام شده در این حوزه، فقط 8% از کارها مربوط به استفادهی مدلها در صنعت بودهاست و 92% از کارهای انجام شده در محیطهای آکادمیک صورت گرفتهاست.

مراجع

[1] M. R.Lyu, Handbook of software reliability engineering. IEEE Computer Society Press, McGraw Hill, 1996.
[2] A.K. Pandey,N.K.Goyal, Early Software Reliability Prediction a Fuzzy Logic Approach, Springer, 2013.
[3] S. Yamada, Software reliability modeling Fundamental and Applications, Japan, Springer, 2014.
[4] F. Febrero, C. Calero, M. A. Moraga, “A systematic mapping study of software reliability modeling,” Information and software technology, Elsevier, vol. 56, no. 8, pp. 839-849, August 2014
[5] Z. KRAJČUŠKOVÁ, “Software reliability models,” Proceeding of 17th International Conference on Radioelektronika, IEEE, pp.1-4, April 2007.
[6] H. A. Stieber, “A family of software reliability growth models,” Proceeding of 31th Annual International Computer Software and Applications Conference, IEEE, vol. 2, pp. 217-224, July2007.
[7] L. shanmugam, L. Florence, “An Overview of Software Reliability Models,” International Journal of Advanced Research in Computer Science and Software Engineering, vol. 2, no. 10, pp. 36-42, October 2012.
[8] engineering series, London, Springer, 2006.
[9] N. Fenton, N. Neil, W. Marsh, P. Hearty, L. Radlinski&P. Krause, “On theeffectiveness of early life cycle defect prediction with Bayesian Nets,” Empirical of Software Engineering,ACM, vol. 13, no. 5, pp. 499–537, October 2008.
[10] J. An, J. Zhu, “ Software reliability modeling with integrated test coverage,” Proceeding of Fourth International Conference on Secure Software Integration and Reliability Improvement, IEEE, pp. 106-112, June 2010.
[11] A. Immonen, E. Niemela, “Survey of reliability and availability prediction methods from the viewpoint of software architecture,” Software & System Modeling, Springer, vol. 7, no. 1, pp. 49-65, Jan 2007.
[12] Ch. A. Asad, M. I. Ullah, M. J. Rehman, “An approach for software reliability model selection,” Proceedings of the 28th Annual InternationalComputer Software and Applications Conference, IEEE, vol. 1, pp. 534-539, Sept 2004.