فهرست مطالب

۱ تست نرم‌افزار

۲ تست امنیت نرم‌افزار

۳ استاندارد تست نرم‌افزار: OWASP

۴ روش تست نرم‌افزار: ASVS

۵ تست برنامه‌های کاربردی وب بر اساس OWASP

فصل اول

۱

تست نرم افزار

تست نرم‌افزار

1-1 خلاصه فصل

«کیفیت، هدف اصلی تمامی افراد، شرکت ها و بصورت کلی سازمان هایی است که در جهت تولید و توسعه نرم افزار گام بر می دارند. در واقع کیفیت نرم افزار یک اصل مهم رقابتی در زمینه تولید یک محصول است، یکی از شاخه های اصلی برای رسیدن به این هدف، تست یا آزمون نرم افزار است. در واقع تست نرم افزار به دنبال خطایابی و عیب یابی محصول، قبل از تحویل به مشتری است. اینکه هم توسعه دهندگان و هم کاربران نهایی بر روی یک نرم افزار کارآمد و قابل بکارگیری که پاسخگوی نیازمندی های تعریف شده باشد هم نظر باشند بسیار اهمیت دارد.

آمارهایی که از سازمان های رسمی[1] در زمینه مشکلات نرم افزارها و پیامدهای ناشی از آن منتشر می شود، مشکلاتی که در برخی موارد غیر قابل جبران می باشند، نیاز و توجه ویژه به مقوله تست نرم افزار را نشان می دهد، فعالیت هایی که باید در فرآیندهای چرخه حیات نرم افزار در نظر گرفته شود و با دیدی جامع و توجه به تمام جوانب، آزمون های متفاوتی را انجام داد تا بتوان ریسک های موجود در محصول را به حداقل برسانیم.

اهداف کلی این بخش:

- آشنایی با مفهوم تست نرم افزار و بررسی چرخه حیات آن
- بررسی روش ها و سطوح تست نرم افزار
- آشنایی با متریک های تست نرم افزار
- آشنایی با دوره های تست و استانداردهای مرتبط

1-2 تست نرم افزار چیست

تست نرم افزار، فرآیند اجرای یک برنامه یا سیستم با هدف پیدا کردن خطاهاست، یا هر فعالیتی با هدف ارزیابی ویژگی یا قابلیت یک برنامه یا سیستم مشخص و برآورده کردن نتایج مورد نیاز را، دربر می گیرد. تست نرم افزار، برخلاف دیگر فرآیندهای فیزیکی که ورودی ها دریافت و خروجی ها تولید می شوند نیست. تفاوت های نرم افزار در شیوه ای است که در آن با شکست مواجه شده است. اغلب سیستم های فیزیکی در یک مجموعه از راه های ثابت شکست می خورند. در مقابل، نرم‌افزار می تواند از راه های بسیار متفاوتی و ناشناخته ای دچار شکست شود. به طور کلی کشف تمام حالت‌های مختلف شکست برای نرم افزار، کاری غیر ممکن است. بر خلاف اکثر سیستم های فیزیکی، اغلب کاستی‌ها در نرم افزار، خطاهای طراحی هستند نه عیب های ساخت. نرم افزار از فرسایش، فرسودگی و کهنگی رنج نمی برد، به طور کلی تا زمان به روز رسانی ها یا تا زمان منسوخ شدن، تغییر نمی‌کند. بنابراین یک بار که نرم افزار تولید شود، عیوب طراحی (یا اشکال ها[2]) فراموش می شوند و تا زمان فعال سازی، پنهان باقی می مانند.

اشکالات نرم افزاری تقریبا همیشه در هر ماژول نرم افزاری با اندازه متوسط وجود دارند؛ نه به خاطر اینکه برنامه نویسان بی دقت یا بی مسئولیت هستند، بلکه به خاطر اینکه به طور کلی پیچیدگی نرم افزار سخت و

۱ (Computer Emergency Response Team (CERT
۲ Bugs

دیر مهار است و بشر تنها توانایی محدودی برای مدیریت پیچیدگی دارد. همچنین این مطلب برای هر سیستم پیچیده ای صحیح است که کاستی‌های طراحی، هرگز نمی توانند به طور کامل نفی شوند.

به دلیلی مشابه پیچیدگی، کشف عیوب طراحی در نرم افزار، به همان اندازه مشکل است. به این علت که نرم افزار و هر سیستم دیجیتالی، پیوسته نیستند، تست مقادیر مرزی برای تضمین صحت و درستی، کافی نمی باشند. تمام مقادیر ممکن نیاز دارند که تست و تأیید شوند اما تست کامل غیر عملی است. تست جامع یک برنامه ساده که تنها دو ورودی عدد صحیح ۳۲ بیتی را با هم جمع می کند، موجب $۲^{۶۴}$ نمونه تست مجزا می شود که انجام آن صدها سال زمان خواهد برد، حتی اگر تست ها با نرخ هزاران تست در ثانیه انجام شود. بدیهی است برای یک ماژول نرم افزاری واقعی، نتیجه فراتر از مثال نام برده می باشد. اگر ورودی ها، از دنیای واقعی باشند، مشکل بدتر خواهد شد. زیرا زمان بندی و تأثیرات محیطی غیرقابل پیش بینی و اثرات متقابل انسان، همگی پارامترهای ورودی ممکن تحت بررسی، خواهند بود.

مشکلات بیشتر در رابطه با طبیعت پویای برنامه ها می باشد. اگر در طی تست اولیه شکستی رخ دهد و کد تغییر یابد، نرم افزار ممکن است حالا برای یک نمونه تستی که قبلا کار نمی کرده، کار کند و جواب دهد. اما رفتار آن بر روی نمونه تست های پیش از رخ دادن خطا، که قبلا آن ها را سپری کرده، دیگر نمی تواند تضمین شده باشد. برای به حساب آوردن این امکان، عملیات تست باید از ابتدا شروع شود که هزینه انجام این کار اغلب گران و بازدارنده است.

عملیات تست معمولا برای اهدافی انجام می گیرد از جمله:

- **بهبود کیفیت**

از آنجایی که کامپیوترها و نرم افزار در برنامه های کاربردی حیاتی مورد استفاده قرار می گیرند، پیامد یک اشکال می‌تواند بسیار شدید و جدی باشد. اشکالات می توانند باعث خسارت های بزرگی شوند. اشکال ها در سیستم های بحرانی باعث سقوط هواپیما، انحراف مأموریت شاتل فضایی، توقف معاملات در بازار بورس و موارد بدتری شده اند. اشکالات می‌توانند باعث کشتن انسان ها و باعث فجایع شوند. در دنیای احاطه شده با کامپیوتر، کیفیت و قابلیت اطمینان نرم‌افزار، موضوع مرگ و زندگی است.

کیفیت به معنای منطبق بودن با الزامات طراحی مشخص و تعیین شده است. وجود صحت، حداقل نیازمندی کیفیت، به معنای انجام آنچه که نیاز است می باشد والبته تحت شرایط مشخص شده. اشکال زدایی، دیدگاه محدودی از تست نرم‌افزار است که به شدت به وسیله برنامه نویس برای پیدا کردن نواقص طراحی انجام می شود. نقص ناشی از طبیعت انسان تقریبا این امر را غیر ممکن می سازد که یک برنامه نسبتا پیچیده در اولین بار درست باشد. پیدا کردن مشکلات و رفع آن ها، هدف اشکال زدایی در فاز برنامه نویسی است.

همچنین می توانیم کیفیت در میان محصولات مختلف را تحت خصوصیات مشابه بر اساس نتایج یک تست یکسان مقایسه کنیم. ما نمی توانیم کیفیت را به صورت مستقیم تست کنیم اما این کار را می توانیم به وسیله فاکتورهای مربوطه برای قابل مشاهده کردن کیفیت، انجام دهیم. این فاکتورها می توانند به عوامل و جزئیات بیشتری در سطوح پایین تر شکسته شوند. جدول ۱۱ متداول ترین ملاحظات و فاکتورهای کیفیت را نمایش می دهد.

جدول ۱-۱ نمونه فاکتورهای کیفیت نرم افزار

کیفیت های بیرونی[۱]	کیفیت های درونی[۲]	کیفیت های بعدی[۳]
صحت	کارآیی	انعطاف پذیری
قابلیت اطمینان	تست پذیری	قابلیت استفاده مجدد
قابلیت استفاده	مستندات	قابلیت نگهداری
تمامیت و یکپارچگی	ساختار	

یک تست خوب، سنجش ها برای همه فاکتورهای مرتبط را فراهم می کند. اهمیت هر فاکتور خاص، از یک برنامه کاربردی به برنامه دیگر متفاوت است. هر سیستمی که با زندگی انسان رابطه دارد، در خطر است و باید تأکید شدید بر قابلیت اطمینان و یکپارچگی این سیستم ها داشت. در سیستم های تجاری قابلیت های استفاده و نگهداری فاکتورهای کلیدی هستند در حالی که برای یک برنامه علمی ممکن است هیچ یک قابل توجه نباشند.

- **وارسي و اعتبار سنجي**

یکی دیگر از اهداف مهم تست، بحث وارسی و اعتبارسنجی می باشد. تست می تواند به عنوان معیار به کار رود. تست به شدت به عنوان یک ابزار در فرایند V&V مورد استفاده قرار می گیرد. تست کنندگان می توانند بر اساس تفسیری از نتایج تست، ادعا کنند که محصول تحت شرایط خاص کار می کند، یا کار نمی کند. تست با هدف اعتبارسنجی آثار و نتایج محصول، تست تمیز یا تست مثبت نامیده می شود. تعداد محدوی تست نمی تواند کار کردن نرم افزار در تمام موقعیت‌ها را تأیید کند. در مقابل، تنها یک تست شکست خورده برای نشان دادن این که نرم افزار کار نمی کند کافی است. تست کثیف یا تست منفی به تست هایی گفته می شود که هدفشان شکست نرم افزار می باشد و نمایش اینکه نرم‌افزار کار نمی کند.

- **تخمین قابلیت اطمینان**

قابلیت اطمینان نرم افزار ارتباطات مهمی با بسیاری از جنبه های نرم افزار از جمله ساختار و میزان تست آن دارد . بر اساس پروفایل عملیاتی (تخمینی از دفعات استفاده از ورودی های گوناگون به برنامه)، عملیات تست می تواند به عنوان یک متد نمونه آماری به منظور به دست آوردن داده های ناموفق برای تخمین قابلیت اطمینان به کار گرفته شود.

1-3 وابستگي تست و كيفيت

در بسیاری از شرکت ها حدود ۳۰ تا ۵۰ درصد هزینه تولید نرم افزار صرف تست آن می شود، هنوز هم خیلی از افراد اعتقاد دارند که نرم افزارها قبل از انتشار، به درستی تست نمی شوند. این شرایط به دو دلیل به وجود می‌آید. اول این که تست نرم افزار امری بسیار مشکل است. دوم این که تست معمولا" بدون متدولوژی مشخص و ابزار لازم انجام می گیرد. آنچه که باعث می شود تا شرکت‌ها هزینه زیادی صرف تست نرم افزارها کنند، چیزی جز دستیابی به کیفیت مطلوب نیست.

کیفیت چیزی است که ما در تولیدات، فرایندها و خدمات به دنبال آن هستیم. کیفیت یک ویژگی منحصر به فرد نیست، بلکه یک مشخصه چند بعدی است و می تواند در یک فرایند یا محصول وجود داشته باشد. کیفیت به مشخصه ای اطلاق می شود که :

- تعدادی از نیازمندی‌های توافق شده را برآورده کند.
- به وسیله معیارهای سنجش و اندازه گیری توافق شده ارزیابی شود.
- به وسیله فرآیند مورد قبولی تولید شود.

کیفیت نرم افزار را می‌توان به دو دسته تقسیم کرد:

کیفیت محصول: کیفیت محصول در مورد کیفیت محصولی که به وسیله فرآیندها تولید می شود، بحث و نتیجه گیری می کند.

کیفیت فرآیند: کیفیت فرآیند به میزان مقبولیت و کیفیت یک فرآیند که برای تولید یک محصول اجرا می شود، اشاره می کند. اگر ما یک فرایند با کیفیت داشته باشیم، ریسک تولید محصول با کیفیت پایین بسیار کم می شود، در حالی که خلاف این مورد معمولا" درست نیست. یعنی داشتن یک محصول نهایی با کیفیت بالا، دلیل بر وجود یک فرآیند با کیفیت بالا نیست.

1-4 دلایل اهمیت تست نرم‌افزار و پیامدهاي عدم وجود آن

صرف نظر از محدودیت‌ها، تست بخش جدایی‌ناپذیر در توسعه نرم‌افزار است و باید به طور گسترده در هر فاز در چرخه توسعه نرم‌افزار مستقر شود. به طور معمول، بیش از ۵۰ درصد از زمان توسعه در تست به سر می‌برد. نرم‌افزاری که به درستی کار نکند، می‌تواند تأثیر مخرب زیادی بر روی یک سازمان داشته باشد. بدون انجام ارزیابی بر روی نرم‌افزار، نمی‌توان از مشکلات عملیاتی، کیفتی و در نتیجه امنیتی آن مطلع شد که این مسئله منجر به بروز مشکلات زیادی ازجمله این موارد می‌شود:

- ❖ از دست دادن سرمایه: این مورد می‌تواند شامل از بین رفتن حقوق مشتری‌ها باشد که به دلیل عدم رعایت الزامات قانونی منجر به مجازات‌های مالی می‌گردد.
- ❖ از دست دادن زمان: علت این مورد می‌تواند معاملاتی باشد که زمان زیادی برای پردازش می‌برد اما می‌تواند شامل کارکنی شود که قادر نیست به خاطر یک خطا یا کاستی کار کند.
- ❖ خسارت به شهرت کسب و کار: اگر یک سازمان با توجه به مشکلات نرم‌افزاری، قادر به ارائه خدمات به مشتریان خود نباشد، مشتریان اعتماد و ایمان خود به این سازمان را از دست می‌دهند. (و احتمالا کسب و کار خود را در جایی دیگر انجام می‌دهند)
- ❖ آسیب یا مرگ: برخی از سیستم‌های بحرانی در بحث ایمنی، اگر به درستی کار نکنند می‌توانند باعث آسیب و یا مرگ و میر شوند. (به عنوان مثال نرم‌افزار کنترل ترافیک پرواز)

1-5 چه موقع باید عمل تست انجام شود

برای اینکه تست هزینه موثر داشته باشد باید بتوان خطاها را تا آنجا که امکان دارد به محض بروزشان

از منبع حذف کند. بنابراین تست باید یک فرایند مستمر و پیوسته در طول چرخه حیات نرم افزار باشد. تست متدیک می تـواند هرگاه که یک محصول به یک وضعیت جدید از پالایش می رسد, انجام شود. اما مهندسی نرم افزار صحیح به بیش از این نیاز دارد. مهندس نرم افزار باید تست غیر رسمی را همزمان با توسعه محصول انجام دهد. فرضیات باید حذف شوند و با قوانین بررسی شده جایگزین شوند. وقتی یک خطا آشکار می شود باید پیگیری شود تا منبع آن یافت شود, سپس روالها باید تغییر داده شده و اصلاح شوند. تا از وقوع مجدد خطاها جلوگیری شود. با ترکیب تست غیر رسمی و متدیک هدفمان ایجاد یک روش کار است که فقط منحصر به کشف زود هنگام خطاها نباشد بلکه از وقوعشان جلوگیری شود. گذشته از همه آنچه که ذکر شد موثر ترین روش تولید محصول از لحاظ هزینه, دریافت محصول سالم از همان ابتد است که هزینه ناشی از درمان اشتباهات را از بین می برد.

1-6 تست نرم افزار در تئوري و عمل

یکی از مشکلات مهم در کامپیوتر توسعه نرم افزار قابل اعتماد است. متد اصلی برای رسیدن به یک محصول نرم افزاری مطمئن تست می باشد. استراتژی های تست محصول, آن را روی زیر مجموعه کوچکی از ورودی هایش اجرا می کنند و مشکل اساسی از همین جا آغاز می شود که موارد تست را چگونه انتخاب کنیم تا تنها با تکیه بر نمونه کوچکی از ورودی‌ها اطمینان کافی در مورد محصول کسب کنیم. استراتژی های فراوانی در زمینه تست بیان شده است. اغلب استراتژی ها نوعی دانش و آگاهی در مورد برنامه را به کار می گیرند که ممکن است ساختار برنامه, مشخصات تابعی برنامه اطلاعات در مورد انواع خطاهایی که غالباً روی می دهند یا ترکیبات گوناگونی از آنها باشد. اتوماسیون فرایند تست عامل مهمی در بکارگیری یک متد است که شامل ابزارهایی برای تولید موارد تست است تا یک معیار تست را برآورده سازد. ابزارهایی که موارد تست را اجرا کرده و نظارت می کنند و ابزارهایی که صحت خروجی ها را بررسی می کنند. مساله مهم دیگر نوع نتایجی است که از حاصل تست می توان به آنها (درباره قابلیت اعتماد برنامه) رسید.

1-6-1 تئوري تست

در طول سال ها استراتژی های تست زیادی پیشنهاد شده اند که هر کدام بر یک قضیه منطقی استوار است. اگر شما عمل تست را چنان و چنین انجام ندهید چگونه می توانید مطمئن باشید که برنامه کار می کند؟ یا "اگر ما عمل X را انجام دهیم ممکن است Y را از دست بدهیم لذا بهتر است Z را انجام دهیم" حتی طرحهای تستی وجود دارند که معادل اثبات های برنامه هستند. متاسفانه این طرح ها مجبورند به طور جدی انواع خطاهایی را که می توانند کنترل شوند یا آنهایی را که در عمل نمی توانند پیاده سازی شوند را محدود کنند.

ارزیابی های زیادی از استراتژیهای گوناگون صورت گرفته و مقایسه های زیادی بین آنها انجام شده است. اما چندین عامل، ارزش آنها را در عمل محدود می کند. و واضح است که هیچ برنده آشکاری از نظر استراتژی وجود ندارد. به عنوان مثال معنی اینکه تست مسیر از تست شاخه بهتر است چیست؟ درست است که تست مسیر برنامه را کاملتر تست می‌کند ولی هزینه اش نیز بالاتر است و هنوز این امکان وجود دارد که مجموعه ای از n مورد تست برای تست شاخه در نهایت به خطایی دست یابد که n*m مورد تست مسیر نتواند. آیا یک مقایسه عادلانه، تست شاخه را m برابر تکرار می‌کند؟ از هر روش به چه نتایجی درباره قابلیت اعتماد برنامه می توان رسید؟ تست تصادفی دانش حداقلی را از برنامه برای طراحی موارد تست به کار می برد. به نظر می رسد که هیچ شانسی برای رقابت با استراتژی هایی که دانش برنامه را برای طراحی موارد تست به کار می برند وجود نداشته باشد. اما برخی از مطالعات انجام شده نشان می دهند که تست تصادفی نزدیک به تست پارتیشن عمل می کند. از طرف دیگر موقعیت هایی وجود دارند که تست تصادفی در آنها ناموثر است.

1-6-2 تست در عمل

تعداد کمی از استراتژی های تست فراوانی که پیشنهاد شده اند در عمل به کار رفته اند. عوامل زیادی نظیر هزینه, کمبود ابزار اتوماسیون برای پشتیبانی آنها, مشکلات عملی و تئوری در پیاده سازی آنها, کمبود ارزیابی های واقعی کارایی و ... به این امر کمک می کنند. یک متدولوژی تست نرم افزار به صورت زیر است

- یک دور بازرسی دستی کد برنامه
- یک مقدار ۹۵ درصدی از پوشش سراسری شاخه
- یک مقدار ۹۰ درصدی از پوشش سراسری call – pair
- ۹۸ درصد تست بازگشتی اتوماتیک
- ۵۰ درصد یا بیشتر پوشش مسیر برای ماژول های بحرانی.

چنین متدولوژی ای یک تعداد از استراتژی های تست را بدون تأمین نیازمندی های هر یک به طور کامل ترکیب می‌کند. مقایسه ها و ارزیابی های تئوری استراتژی های گوناگون، چگونه می توانند در داخل چنین ترکیبی از استراتژی ها به کار گرفته شوند؟ در حوزه قابلیت اعتماد نرم افزار سوالات از پاسخ ها فراترند. یک چیز که واضح به نظر می رسد این است که از تقابل بیشتر بین تئوری و عمل نتیجه بهتری حاصل می شود.

1-7 تفاوت تست و اشکال زدایی

تست اشکال زدایی نیست. اشکال زدایی در رابطه با خطاهای ویژه مستقر در اجزای محصول است اما تست با اجرای یک برنامه به قصد یافتن خطاها توسط ابزار منطقی یا فیزیکی صورت می گیرد. یک برنامه تست شده برنامه ای است که هنوز شرایطی که آن را وادار به شکست می کند در آن یافت نشده اند یعنی اگر نتیجه تست با نتیجه مورد نظر سازگار است، فرض می شود که تست قطعه با موفقیت روبرو شده و صحیح است. در تست یک نرم افزار دو هدف عمده مورد نظر است:

- رسیدن به کیفیت مناسب (Debug Testing) : تست به عنوان وسیله ای برای رسیدن به قابلیت اطمینان به کار می رود که در اینجا هدف بررسی نرم افزار برای کشف باگهای قابل اصلاح است که نرم افزار را از نظر قابلیت اطمینان بهبود می بخشد.
- ارزیابی کیفیت موجود (Cope rational Testing) : تست به عنوان وسیله ای برای کسب اطمینان از اینکه نرم افزار قابلیت اطمینان کافی برای به انجام رساندن مقاصدی را که به منظور آنها طراحی شده است "دارا است" به کار می رود. در اینجا هدف ارزیابی قابلیت اطمینان است.

متدهای Debug انتخاب تصادفی داده تست از یک پروفایل عملیاتی را در نظر نمی گیرند در حالی که این امر برای متدهای عملیاتی بسیار حائز اهمیت است. متدهای Debug بدون هیچ مدرک واقعی ادعای موثر بودن در آشکار سازی نواقص و معایب و اصلاح و رفع آنها را دارند، اما یک ارزیابی قابل دفاع از قابلیت اطمینان نتایج ارائه نمی دهند.

از طرف دیگر متدهای عملیاتی که ارزیابی صحیح و دقیق را عرضه می کنند ممکن است برای رسیدن به قابلیت اطمینان مفید نباشند. بررسی ارتباط این دو هدف با به کارگیری تحلیل و آنالیز احتمالی امکان پذیر است. یک متد تست سیستماتیک معیاری برای انتخاب موارد تست دارد و یک معیار برای تصمیم گیری اینکه

چه موقع تست را پایان دهد. اغلب تست های سیستماتیک یا بر پایه یافتن اشکالات با نمونه برداری از همه موقعیت هایی که احتمال دارد منجر به شکست شود تکیه دارند، یا با تمرکز روی موقعیت هایی که بیشترین احتمال را در این امر دارند.

1-8 مفاهیم بنیادي در تست نرم‌افزار

در فرآیند توسعه، هدف آن است که یک سیستم با مشخصات خواسته شده تولید شود. هدف این فرآیند از یک سو برآورده ساختن نیازهای کاربران و از سوی دیگر تضمین کیفیت مناسب عملکرد سیستم است. بنابراین بایستی حاوی مکانیزم هایی برای اعتبارسنجی و وارسی باشد. پیدا کردن نقص در اواخر چرخه حیات توسعه می تواند به طور قابل توجهی در زمان و سرمایه پر هزینه باشد. فرآیندهای خوب V&V می توانند قبل از اینکه سیستم وارد تست شود، به کاهش نقص در توانمندی های عملیاتی کمک کنند. ارزیابی نرم افزار، یک عنصر از عنوان گسترده‌تری است که اغلب با وارسی و اعتبارسنجی شناخته می شود.

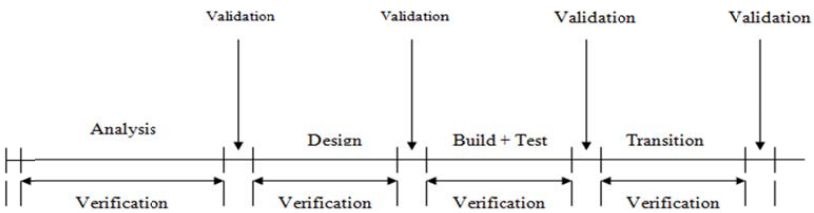
1-8-1 وارسي

مطابق استاندارد ISO 9000 وارسی عبارت است از فراهم آوردن شواهد عینی در مورد اینکه الزامات و یا خواسته‌های مشخص شده برآورده شده‌اند. اصطلاح "وارسی شده" به‌منظور مشخص‌کردن وضعیت مربوطه (پس از وارسی) بکار می‌رود. وارسی می‌تواند شامل فعالیت‌هایی باشد از قبیل:

- ✓ انجام محاسبات به روش‌های دیگر
- ✓ مقایسه مشخصات یک طراحی جدید با مشخصات طراحی‌های مشابه که درستی‌آنها به اثبات رسیده است
- ✓ انجام آزمون‌ها و اثبات از طریق نشان دادن
- ✓ بازنگری مدارک پیش از صدور

1-8-2 اعتبارسنجي

مطابق استاندارد ISO 9000، اعتبارسنجی عبارت است از تأیید از طریق فراهم آوردن شواهد عینی در مورد اینکه الزامات و یا خواسته‌ها برای استفاده مورد نظر یا کاربرد خاص برآورده شده‌اند. اصطلاح "اعتبارسنجی شده" به منظور مشخص کردن وضعیت مربوطه (پس از اعتبارسنجی) به‌کار می‌رود. شرایط استفاده می‌تواند واقعی یا شبیه‌سازی شده باشد. شکل I چگونگی ارتباط V&V در چرخه حیات نرم افزار را نمایش می دهد.



شکل I جایگاه V&V در SDLC