





دانشگاه شهید بهشتی
دانشکده مهندسی برق و کامپیوتر

ارائه روش تست خودکار نرم افزار بر پایه هست‌شناسی

پایان نامه کارشناسی ارشد مهندسی کامپیوتر
گرایش نرم‌افزار

هادی سروری

کتابخانه مرکزی
تهران

استاد راهنما:

دکتر اسلام ناظمی

۱۳۸۸/۱۲/۲

۱۳۸۸

سه

۱۳۱۶۴۴


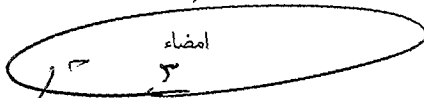




دانشگاه شهید بهشتی
دانشکده مهندسی برق و کامپیوتر

پایان نامه کارشناسی ارشد مهندسی کامپیوتر گرایش نرم افزار
تحت عنوان:

ارائه روش تست خودکار نرم افزار بر پایه هست‌شناسی

در تاریخ ۱۳۸۸/۵/۲۴ پایان نامه دانشجو، (هادی سروری)، توسط کمیته تخصصی داوران مورد بررسی و تصویب نهائی قرار گرفت.

	دکتر اسلام ناظمی	۱- استاد راهنما:
	دکتر فریدون شمس	۲- استاد داور (داخلی)
	دکتر سید حسن میریان	۳- استاد داور (خارجی)
	دکتر فرح ترکمنی آذر	۴- نماینده تحصیلات تکمیلی

با تشکر و قدر دانی فراوان از استاد عزیز و گرانقدر جناب آقای دکتر ناظمی که در تمامی مراحل پروژه و در دوره تحصیل کارشناسی ارشد یار و یاور بنده بودند و با تشکر از استاد عزیز جناب آقای دکتر شمس که زحمات فراوانی در این سالها برای رشته کامپیوتر و بنده حقیر متقبل شدند.

کلیه حقوق مادی مترتب بر نتایج مطالعات،
ابتکارات و نوآوریهای ناشی از تحقیق موضوع
این پایان نامه متعلق به دانشگاه شهید بهشتی
می باشد.

به نام خدا


نام و نام خانوادگی: هادی سروری

عنوان پایان نامه: ارائه روش تست خودکار نرم افزار بر پایه هست‌شناسی

استاد راهنما: آقای دکتر اسلام ناظمی

اینجانب هادی سروری تهیه کننده پایان نامه کارشناسی ارشد حاضر خود را ملزم به حفظ امانت داری و
قدردانی از زحمات سایر محققین و نویسندگان بنا بر قانون Copyright می دانم. بدین وسیله اعلام
می نمایم که مسئولیت کلیه مطالب درج شده با اینجانب می باشد و در صورت استفاده از اشکال؛
جداول، و مطالب سایر منابع، بلافاصله مرجع آن ذکر شده و سایر مطالب از کار تحقیقاتی اینجانب
استخراج گشته است و امانتداری را به صورت کامل رعایت نموده ام. در صورتی که خلاف این مطلب
ثابت شود، مسئولیت کلیه عواقب قانونی با شخص اینجانب می باشد.

نام و نام خانوادگی دانشجو: هادی سروری

امضاء و تاریخ:  ۱۳۸۸ / ۵ / ۱۳

با تقدیم به همسر عزیزتر از جانم

فهرست مطالب

عنوان

صفحه

۱.....	فصل اول
۱.....	مقدمه
۴.....	فصل دوم
۴.....	مروری بر روش‌های مختلف خودکارسازی آزمون نرم‌افزار
۴.....	۱-۲ مقدمه
۴.....	۲-۲ آزمون خودکار نرم‌افزار
۴.....	۱-۲-۲ آزمون نرم‌افزار چیست؟
۷.....	۲-۲-۲ فعالیت بازبینی و اعتبار سنجی
۸.....	۳-۲-۲ سطوح مختلف آزمون نرم‌افزار
۹.....	۱-۳-۲-۲ آزمون واحد
۱۰.....	۲-۳-۲-۲ آزمون یکپارچه‌سازی
۱۱.....	۳-۳-۲-۲ آزمون سیستم
۱۲.....	۴-۳-۲-۲ آزمون پذیرش
۱۲.....	۴-۲-۲ مستندات آزمون
۱۲.....	۱-۴-۲-۲ توصیفات نرم‌افزار و آزمون
۱۴.....	۲-۴-۲-۲ مستندات طراحی آزمون
۱۵.....	۳-۴-۲-۲ استراتژی آزمون
۱۵.....	۴-۴-۲-۲ طرح‌های آزمون
۱۸.....	۵-۴-۲-۲ مستندات نتایج آزمون
۱۸.....	۳-۲ روش‌های خودکار سازی

- ۱۹..... ۱-۳-۲ کارهای انجام شده در زمینه خودکار سازی
- ۲۱..... ۲-۳-۲ مقایسه روشهای گذشته با این پژوهش
- ۲۲..... ۴-۲ روشهای مختلف تولید موردهای آزمون نرم افزار
- ۲۴..... ۱-۴-۲ روشهای عمومی
- ۲۴..... ۲-۱-۴-۲ آزمون مثبت و منفی
- ۲۵..... ۲-۱-۴-۲ آزمون جعبه سیاه و جعبه سفید (جعبه شیشه‌ای)
- ۲۶..... ۳-۱-۴-۲ پیش بینی خطاها
- ۲۶..... ۲-۴-۲ روشهای آزمون وظیفه‌مندی
- ۲۷..... ۱-۲-۴-۲ بخش بندی برابر
- ۲۷..... ۲-۲-۴-۲ تحلیل مقادیر مرزی
- ۲۸..... ۳-۲-۴-۲ آزمون مداخله جویانه
- ۲۸..... ۴-۲-۴-۲ آزمون تصادفی
- ۲۸..... ۵-۲-۴-۲ تحلیل انتقال حالتها
- ۲۸..... ۶-۲-۴-۲ آزمون ایستا
- ۲۹..... ۳-۴-۲ آزمونهای غیر وظیفه‌مندی
- ۲۹..... ۱-۳-۴-۲ آزمون پیکره بندی و نصب
- ۲۹..... ۲-۳-۴-۲ آزمون سازگاری و مشارکت
- ۲۹..... ۳-۳-۴-۲ آزمون مستندات و راهنمایهای کاربر
- ۲۹..... ۴-۳-۴-۲ آزمون تصحیح در صورت بروز خطا
- ۳۰..... ۵-۳-۴-۲ آزمون کارایی
- ۳۰..... ۶-۳-۴-۲ آزمون فشار
- ۳۰..... ۵-۲ آزمون نرم افزار مبتنی بر مدل
- ۳۲..... فصل سوم

۳۲	مروری بر هست‌شناسی
۳۲	۱-۳ هست‌شناسی
۳۳	۱-۱-۳- تعریف هست‌شناسی
۳۵	۱-۲-۱-۳- انواع هست‌شناسی‌ها
۳۸	۱-۳-۳- زبانهای هست‌شناسی
۴۴	۲-۳ توصیف ابزار protege
۴۵	۱-۲-۳- ویرایشگر قابها
۴۵	۲-۲-۳- ویرایشگر OWL
۴۷	فصل چهارم
۴۷	رویکرد پیشنهادی
۴۷	۱-۴ تولید موردهای آزمون بر مبنای نمودار تغییر حالت ۲.۰ UML با بیان هست‌شناسی
۴۸	۱-۱-۴- نمودار تغییر حالت
۴۹	۲-۱-۴- تولید موردهای آزمون از روی نمودار تغییر حالت
۵۲	۲-۴ متد پیشنهادی برای تولید مورد آزمون از نمودار تغییر حالت
۵۲	۱-۲-۴- انتخاب گزاره‌های
۵۳	۲-۲-۴- تشکیل توابع ارزیابی برای انتخاب موردهای آزمون
۵۴	۳-۲-۴- مثال داده انتخابی برای نمودار تغییر حالت ماشین بستنی فروشی
۵۴	۳-۴ الگوریتم هیوریستیک انتخاب کلنی
۵۴	۱-۳-۴- الگوریتم
۵۶	۲-۳-۴- استفاده از الگوریتم CS برای ایجاد داده‌های آزمون
۵۶	۱-۲-۳-۴- کد کردن نمونه‌های آزمون به آنتی بادی‌ها
۵۷	۲-۲-۳-۴- تابع مقبولیت
۵۷	۳-۲-۳-۴- حالت خاتمه

۵۷	۴-۴ پیاده‌سازی
۵۷	۴-۴-۱- استفاده از ابزار پشتیبانی UML برای ترسیم نمودار تغییر حالت و استخراج XML
۵۸	۴-۴-۲- تعریف هست‌شناسی برای نمودار تغییر حالت
۶۰	۴-۴-۳- استفاده از رابطه‌های عمودی در بین نمودارهای UML
۶۲	۴-۴-۱-۳- استفاده از UML(VR) برای آزمون نرم‌افزار
۶۲	۴-۴-۴- تبدیل کدهای XML به OWL
۶۳	۴-۴-۵- استفاده از ابزار DOM برای پارس کردن کدهای XML
۶۴	۴-۴-۶- پیاده‌سازی برنامه تولید موردآزمون
۶۵	۴-۴-۷- اجرا و ارزیابی داده‌های آزمون
۶۶	فصل پنجم
۶۶	مطالعه موردی
۶۶	۵-۱ مقدمه
۶۷	۵-۲ مثال ماشین فروش خودکار (ماشین بستنی فروشی)
۷۰	۵-۳ نمودار تغییر حالت برای برگزاری یک کلاس
۷۳	۵-۴ نمودار تغییر حالت سیستم تنظیم کننده درجه حرارت
۷۶	۵-۵ نمودار تغییر حالات حساب بانکی
۷۹	۵-۶ نمودار تغییر حالت نحوه محاسبه تخفیف برای کلاس مشتری در دفتر فروش
۸۱	۵-۷ نمودار تغییر حالت خودپرداز بانک
۸۵	فصل ششم
۸۵	مقایسه و نتیجه گیری
۸۶	۶-۱ پیشنهادات
۸۸	فهرست مراجع

فهرست شکلها

- شکل ۱-۲ : سطوح سه گانه توصیفات نرم افزار ۱۳
- شکل ۲-۲ : بخش بندی های برابر ۲۷
- شکل ۱-۳ : یک هست شناسی کوچک نمونه ۳۴
- شکل ۲-۳ : دسته بندی محتوایی Lassia , McGuining ۳۶
- شکل ۳-۳ : هست شناسی سطح بالای sowa ۳۷
- شکل ۴-۳ : دسته بندی موضوعی هست شناسی ۳۷
- شکل ۵-۳ : زبان های هست شناسی و ارتباط آنها ۳۸
- شکل ۶-۳ : شمایی از دو محیط ویرایشگر نرم افزار protege ۴۵
- شکل ۱-۴ : نمودار تغییر حالت با دو حالت ترکیبی ۴۸
- شکل ۲-۴ : نمودار تغییر حالت ماشین بستنی فروشی ۵۲
- شکل ۳-۴ : الگوریتم انتخاب کلنی ۵۵
- شکل ۴-۴ : الحاق کد شده تمام ورودیها ۵۷
- شکل ۵-۴ : ترسیم نمودار تغییر حالت ۵۸
- شکل ۶-۴ : تعریف هست شناسی در نرم افزار protege ۵۹
- شکل ۷-۴ : تعریف رابطه عمودی ۶۱
- شکل ۸-۴ : استفاده از اصول بیان شده تحت عنوان XSLT برای تبدیل کدهای XML به OWL ۶۳
- شکل ۹-۴ : شمایی کلی مراحل تولید موردهای آزمون در روش ارائه شده ۶۴
- شکل ۱-۵ : نمودار تغییر حالت ماشین بستنی فروشی ۶۸
- شکل ۲-۵ : قسمتی از فایل XML استخراج شده از نرم افزار magicdraw ۶۹
- شکل ۳-۵ : داده های آزمون تولید شده برای نمودار تغییر حالت ماشین بستنی فروشی ۷۰
- شکل ۴-۵ : نمودار تغییر حالت برگزاری یک درس ۷۱
- شکل ۵-۵ : قسمتی از فایل XML ساخته شده برای نمودار تغییر حالت برگزاری یک کلاس ۷۲

- شکل ۵-۶: مورد‌های آزمون ساخته شده برای نمودار تغییر حالت برگزاری یک کلاس..... ۷۳
- شکل ۵-۷: نمودار تغییر حالت برای سیستم تنظیم دما..... ۷۴
- شکل ۵-۸: قسمتی از فایل XML ساخته شده برای نمودار تغییر حالت سیستم تنظیم دما..... ۷۵
- شکل ۵-۹: داده‌های آزمون تولید شده برای نمودار تغییر حالت سیستم تنظیم دما..... ۷۵
- شکل ۵-۱۰: نمودار تغییر حالت برای حساب بانکی..... ۷۷
- شکل ۵-۱۱: قسمتی از فایل XML ساخته شده برای نمودار تغییر حالت حساب بانکی..... ۷۸
- شکل ۵-۱۲: داده‌های آزمون تولید شده برای نمودار تغییر حالت حساب بانکی..... ۷۸
- شکل ۵-۱۳: نمودار تغییر حالت کلاس مشتری در سیستم فروش..... ۸۰
- شکل ۵-۱۴: قسمتی از کد XML ساخته شده برای نمودار تغییر حالت کلاس مشتری در سیستم فروش..... ۸۰
- شکل ۵-۱۵: داده‌های آزمون برای نمودار تغییر حالت کلاس مشتری در سیستم فروش..... ۸۱
- شکل ۵-۱۶: نمودار تغییر حالت دستگاه خودپرداز..... ۸۲
- شکل ۵-۱۷: قسمتی از فایل XML ساخته شده بوسیله نرم‌افزار magicdraw..... ۸۳
- شکل ۵-۱۸: داده‌های آزمون ساخته شده برای نمودار تغییر حالت خودپرداز بانک..... ۸۴

فهرست جدولها

- جدول ۱-۳: سازنده کلاس در OIL+DAML ۴۰
- جدول ۲-۳: اصول بدیهی در OIL+DAML ۴۱
- جدول ۳-۳: ویژگیهای زبانهای هست‌شناسی مبتنی بر وب را نشان می‌دهد ۴۳
- جدول ۱-۴: نام هر یک از توابع در پیاده‌سازی و کاربرد آنها ۶۵

چکیده

تعریف موردهای آزمون یکی از قسمت‌های مهم و سخت آزمون نرم‌افزار می‌باشد که قسمت اعظم هزینه آزمون را در بر خواهد داشت. این پژوهش روش جدیدی را برای تولید خودکار موردهای آزمون پیشنهاد می‌کند. داده‌های آزمون از روی نمودار حالت (UML (Unified Modeling language استخراج می‌شود. استفاده از هست‌شناسی قابلیت بازنمایی دانش توسط ماشین را فراهم کرده و با استفاده از کدهای XML(Extensible Markup language منتج از نمودار حالت و تبدیل آن به دانش در قالب کدهای زبان owl(Ontology Web Language) هست‌شناسی، امکان بدست آوردن خودکار داده‌های آزمون با استفاده از یک روش مکاشفه‌ای فراهم می‌شود. روش مکاشفه‌ای انتخاب کلنی داده‌های آزمون تصادفی بدست آمده اولیه را بهبود می‌دهد تا نهایتاً داده‌هایی در مقاطع مرزی بدست آید. با توجه به اینکه روشهای پیش از این برای تولید موردهای آزمون از روشهایی غیر از روشهای مکاشفه‌ای استفاده می‌کنند در این روش بر خلاف روشهای دیگر داده‌های غیر صحیح و رشته‌ای نیز قابل تولید خواهند بود. در تحقیقات گذشته کارهای فروانی برای خودکارسازی مراحل مختلف آزمون انجام شده و هر کدام قسمتی از آنرا خودکار نموده‌اند ولی پژوهش انجام شده مراحل تولید داده‌های آزمون را کاملاً به صورت خودکار انجام داده است. در این پژوهش با بیان نمودار حالت بوسیله هست‌شناسی، قابلیت گسترش بیان دیگر نمودارهای UML در این هست‌شناسی به وجود آمده که بواسطه آن می‌توان موارد آزمون را با تلفیق دیگر نمودارها نیز بدست آورد.

کلمات کلیدی

آزمون نرم‌افزار، آزمون مبتنی بر مدل، هست‌شناسی، مورد آزمون، نمودار حالت، انتخاب کلنی، protégé

فصل اول

مقدمه

آزمون نرم‌افزار یکی از فعالیتهای حیاتی توسعه آن می‌باشد. گرچه در سالهای اولیه ساخت نرم‌افزار به آزمون اهمیت زیادی داده نمی‌شد و آزمون نرم‌افزارها به اصلاح کد برنامه‌ها خلاصه می‌شد ولی امروزه تمام توسعه دهندگان نرم‌افزار به آن به عنوان یکی از مهمترین بخشهای تولید نرم‌افزار توجه دارند و روشها و تکنولوژیهای فراوانی برای آن وجود دارد و آزمون نرم‌افزار عمده هزینه تولید نرم‌افزار یعنی بین ۴۰ تا ۷۰ درصد هزینه کلی آنرا را به خود اختصاص می‌دهد.

در آزمون نرم‌افزار سعی داریم خطاها و معایب موجود در آن را هر چه بیشتر مشخص کنیم، لذا برای این کار سعی می‌کنیم حالتی نظیر حالتی واقعی نرم‌افزار را بوسیله داده‌های آزمایشی به وجود آوریم تا در صورت عدم مشاهده نتایج مورد نظر پی به اشتباهات آن ببریم.

به علت پیچیدگیهای نرم‌افزارها قادر به آزمون کردن آنها در کلیه حالات و شرایط آنها نیستیم و لذا انجام یک آزمون جامع و کامل که نرم‌افزار را در کلیه حالات آن مورد آزمایش قرار دهد تا کلیه خطاها در نرم‌افزار مشخص و اصلاح شود وجود ندارد.

انجام یک آزمون از سه مرحله اصلی استراتژی و طراحی آزمون، ساخت موارد آزمون و نهایتاً اجرا و ارزیابی آزمونها تشکیل شده‌است. این موارد در طی اجرای آزمون بارها تکرار می‌شوند و حجم انبوه آنها هزینه آزمون را افزایش می‌دهد. خودکارسازی آزمون کمک شایانی در کاهش هزینه‌های آزمون خواهد داشت. پژوهشهای فراوانی برای خودکارسازی مراحل مختلف آزمون انجام شده‌است و بیشتر پژوهشهای انجام شده و ابزارهای ساخته شده بر روی مرحله نهایی یعنی اجرا و ارزیابی خودکار آزمون تمرکز کرده‌اند، و این به خاطر ماهیت تکرار پذیری این مرحله می‌باشد و اینکه هزینه‌های

بیشتری از آزمون را شامل می‌شود. ولی تعریف و تولید موردهای آزمون در یک استراتژی خاص به علت مشکل بودن انجام خودکار آن و ساختارمندی کمتر، زیاد مورد توجه قرار نگرفته است. در این پژوهش نه تنها امکان تولید موردهای آزمون با استفاده از نمودار حالت را به صورت خودکار به وجود آورده‌ایم بلکه با تعریف یک هست‌شناسی جدید برای نمودار حالت زمینه‌ای را برای تولید خودکار موردهای آزمون بوسیله دیگر نمودارهای طراحی نرم‌افزار و یا تلفیقی از آنها را بدست آورده‌ایم.

هر یک از روشهایی که از نمودارهای طراحی مانند نمودار کلاس، نمودار ترتیب و غیره را برای تولید موردهای آزمون استفاده کرده‌اند داده‌های موجود در این نمودارها را جهت تولید داده‌های آزمون به کار می‌برند. در این پژوهش یک نمادسازی ریاضی برای نشان دادن رابطه‌های بین نمودارهای مختلف طراحی بیان شده‌است. که از آن به رابطه‌های افقی بین نمودارهای طراحی یاد کرده‌ایم. این رابطه‌ها امکان ساختن داده‌های آزمون با ترکیب نمودارهای طراحی را به وجود آورده است.

UML امروزه به شکل رایجی برای طراحی نرم‌افزار استفاده می‌شود و ابزارهای متنوعی آنرا پشتیبانی می‌کند، اکثر این ابزارها امکان ذخیره نمودارهای طراحی را به شکل XML (Extensible Markup language) دارند، در این پژوهش با استفاده از نرم‌افزار Magic Draw نمودارهای حالت مربوط به طراحی نرم‌افزار را رسم می‌کنیم. در این نرم‌افزار نیز امکان استخراج کدهای XML تعبیه شده است.

هست‌شناسی تعریف شده اجزاء مختلف یک نمودار حالت را در لیست موجودیتهای خود و روابط میان این موجودیتها را در بر می‌گیرد. ابزار protégé که نرم‌افزاری قوی برای پشتیبانی زبان هست‌شناسی OWL می‌باشد مورد استفاده قرار گرفته است و از واسط برنامه نویسی این ابزار برای برقراری ارتباط و ساختن نمونه‌های هست‌شناسی آن استفاده شده است. در قسمت اول کار با پارس کردن کدهای XML با ابزار DOM (Document Object Model) و تفسیر آنها امکان ساختن نمونه‌های هست‌شناسی ساخته شده برای نمودار حالت به وجود می‌آید، در مرحله بعدی هست‌شناسی مورد نظر برای پیمایش نمودار حالت و ساختن داده‌های آزمون مورد استفاده قرار می‌گیرد.

برای ساختن داده‌های آزمون ابتدا داده‌های مورد نظر مبتنی بر داده‌های اولیه شروط یالهای نمودار حالت بدست می‌آیند و سپس با توجه به توابع ارزیابی تعریف شده با استفاده از روش مکاشفه‌ای انتخاب کلنی بهبود می‌یابند. توابع ارزیابی به گونه‌ای طراحی می‌شوند که داده‌های نهایی تولید شده بوسیله الگوریتم بهینه‌سازی در نهایت در مقاطع مرزی مقادیر خود

باشند. تابعی برای پیاده‌سازی الگوریتم انتخاب کلنی توسعه داده شده است و نهایتاً داده‌های آزمون تولید شده در یک فرمت خاص در فایل متنی ذخیره خواهند شد.

فصل دوم

مروری بر روش‌های مختلف خودکارسازی آزمون نرم‌افزار

۲-۱- مقدمه

در این فصل نگاه کلی راجع به روش‌های آزمون نرم‌افزار خواهیم داشت و به اینکه آزمون نرم‌افزار چیست و شامل چه فعالیت‌هایی می‌باشد می‌پردازیم. سطوح مختلف آزمون نرم‌افزار را بیان خواهیم کرد و سپس راجع به روش‌های مختلف آزمون سخن خواهیم گفت و در مورد خودکارسازی آزمون بحث را ادامه خواهیم داد. و در نهایت در مورد آزمون نرم‌افزار مبتنی بر مدل شرح کاملی بیان خواهیم کرد.

۲-۲- آزمون خودکار نرم‌افزار

۲-۲-۱- آزمون نرم‌افزار چیست؟

آزمون نرم‌افزار واژه جدید و ناشناخته‌ای به نظر نمی‌رسد. فرهنگ لغت واژه آزمون را اینگونه معنی کرده است، " فرایند ارزیابی کیفیت مواد، معاینه کردن، آزمایش کردن ". و واژه آزمون نرم‌افزار در [۱] چنین معنی شده است " روشی برای مشخص کردن خطاهای نرم‌افزار با اجرای آن در شرایط آزمایشی ". برنامه‌های کامپیوتری از همان اوایل پیدایش، تحت

آزمون قرار می‌گرفته‌اند. در روزهای اولیه توسعه نرم‌افزار روشهای ساختیافته آزمون کمتری وجود داشت و خطایابی^۱ کدها یکی از مراحل اصلی فرایند تولید نرم‌افزار به شمار می‌رفت. همچنان که فرایند توسعه نرم‌افزارها رشد پیدا کرد و روشهای ساخت‌یافته تولید مانند روشهایی که در [۲] عنوان شده است ارائه شد، راهکارهای آزمون نرم‌افزار نیز رشد و توسعه پیدا کرد و به روشها و روشهای ساخت‌یافته و پیچیده‌تری تغییر شکل داد. بیشتر افرادی که امروزه در ساخت نرم‌افزار به روشهای مدرن کار می‌کنند به آزمون نرم‌افزار و هدف آن به عنوان یک فعالیت اصلی و ذاتی ساخت نرم‌افزار می‌نگرند.

برخی از تعاریف رسمی که از آزمون نرم‌افزار بیان شده است به شرح زیر می‌باشد:

"آزمون نرم‌افزار فعالیتی است که به ارزیابی یک مشخصه یا قابلیت نرم‌افزار یا سیستم کمک می‌کند و معین می‌کند که نتایج مورد نظر بدست خواهد آمد یا خیر؟"^[۳].

این تعریف به صورت عادی نیازمندیهای مشخص شده را پوشش می‌دهد، یعنی مبتنی بر درک مستقیم از آزمون نرم‌افزار است و وضعیت‌هایی که سیستم باید در آنها کار کند و نیازمندی‌هایی که باید برآورده شوند را بیان می‌کند. این تعریف به آزمون مثبت^۲ معروف است.

"آزمون نرم‌افزار فرایند اجرای یک برنامه یا سیستم با قصد پیدا کردن خطاهاست"^[۴].

این تعریف به گونه‌ای غیر مستقیم کمتر به نیازمندیهای سیستم پرداخته است در عوض اندیشه جستجوی خطاهای خارج از حیطه نیازمندیهای سیستم که شامل هر خطایی یا مشکلی می‌باشد را می‌پروراند. این تعریف بیشتر به آزمون منفی شناخته می‌شود.

می‌توان گفت آزمون نرم‌افزار ترکیبی از معیارهای هر دو آزمون مثبت و منفی می‌باشد. یعنی در آزمون نرم‌افزار کنترل می‌کنیم که نرم‌افزارها نیازمندیهای مورد نظر را برآورده می‌کنند و سعی می‌کنیم تا خطاهایی که ممکن است در ساخت سیستم، اجرای عملیات و یا بهروری پایین وجود داشته باشد را پیدا کنیم.

در تعریف آزمون با استفاده از واژه ریسک، ریسک به معنی قابلیت مواجهه نرم‌افزار تحت آزمون^۳ با خطا است یا قدرت

اینکه مشکلات و خطاهای ممکن برای کاربر را کاهش دهد.

^۱ Debugging

^۲ positive testing

^۳ Application Under Test(AUT)