

Автоматизиро ванное тестирование



Основные преимущества Автоматизированного тестирования

- проводить чаще регрессионное тестирование
- быстро предоставлять разработчикам отчет о состоянии продукта
- получить потенциально бесконечное число прогонов тестов
- обеспечить поддержку Agile и экстремальным методам разработки
- сохранять строгую документацию тестов
- обнаружить ошибки, которые были пропущены на стадии ручного тестирования

Инструменты Автоматизированного тестирования

- JUnit — тестирование приложений для Java
- TestNG — тестирование приложений для Java
- NUnit — порт JUnit под .NET
- Selenium — тестирование приложений HTML; поддерживает браузеры Internet Explorer, Mozilla Firefox, Opera, Google Chrome, Safari
- TOSCA Testsuite — тестирование приложений HTML, .NET, Java, SAP
- UniTESK — тестирование приложений на Java, Си

Инструменты Автоматизированного тестирования

Automated Web Testing Tools	Automated GUI Testing Tools	Unit Testing Frameworks	Automated Testing Cloud Services
<u>Selenium</u>	<u>Squish</u>	<u>NUnit</u>	<u>Sauce Labs</u>
<u>Watir</u>	<u>Ranorex</u>	<u>xUnit.net</u>	<u>TestingBot</u>
<u>Windmill</u>	<u>TestComplete</u>	<u>PyUnit / unittest</u>	<u>Gridlastic</u>
<u>Ranorex</u>	<u>Test Studio</u>	<u>Junit</u>	<u>CircleCI</u>
<u>SoapUI</u>	<u>eggplant</u>	<u>TestNG</u>	<u>Tddium</u>
<u>Sahi</u>		<u>PHPUnit</u>	<u>CloudBees</u>
<u>Tellurium</u>		<u>Symfony Lime</u>	<u>Mailosaur</u>
		<u>Test::Unit</u>	
		<u>RSpec</u>	

Автоматизация тестирования

Преимущества автоматизации тестирования:

- **Повторяемость** – все написанные тесты всегда будут выполняться однообразно, то есть исключен «человеческий фактор». Тестировщик не пропустит тест по неосторожности и ничего не напутает в результатах.
- **Быстрое выполнение** – автоматизированному скрипту не нужно сверяться с инструкциями и документациями, это сильно экономит время выполнения.
- **Меньшие затраты на поддержку** – когда автоматические скрипты уже написаны, на их поддержку и анализ результатов требуется, как правило, меньшее время чем на проведение того же объема тестирования вручную.
- **Отчеты** – автоматически рассылаемые и сохраняемые отчеты о результатах тестирования.
- **Выполнение без вмешательства** – во время выполнения тестов инженер-тестировщик может заниматься другими полезными делами, или тесты могут выполняться в нерабочее время (этот метод предпочтительнее, так как нагрузка на локальные сети ночью снижена).

Автоматизация тестирования

Преимущества автоматизации тестирования:

- Автотесты работают быстрее, чем человек
- Автотесты выполняются с большей точностью
- Автоматизация тестирования позволяет повысить качество продукта
- Автоматизация может использоваться практически во всех процессах тестирования
- Автотесты могут выполняться ночью

Автоматизация тестирования

Недостатки автоматизации тестирования:

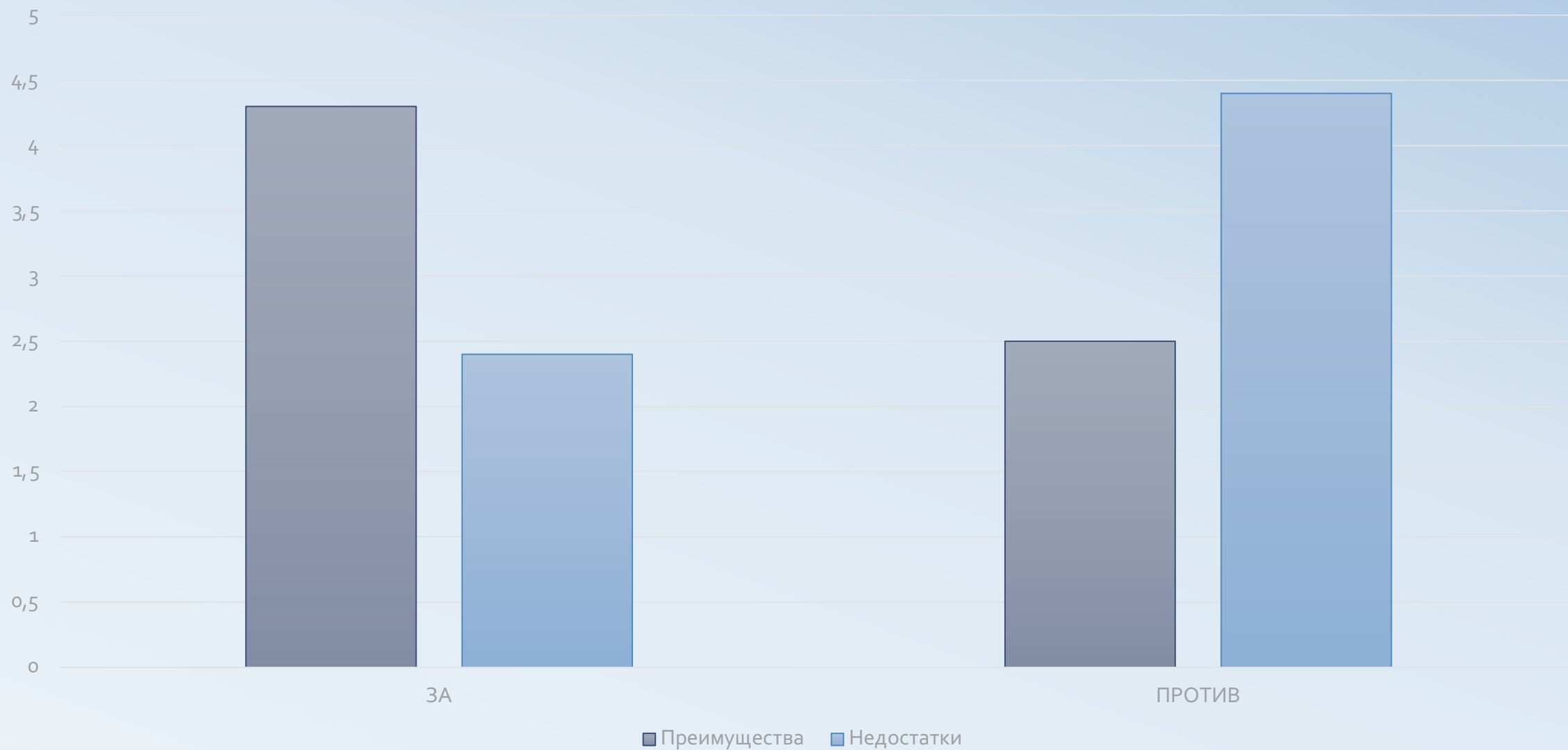
- **Повторяемость** – все написанные тесты всегда будут выполняться однообразно. Это одновременно является и недостатком, так как тестировщик, выполняя тест вручную, может обратить внимание на некоторые детали и, проведя несколько дополнительных операций, найти дефект. Скрипт этого сделать не может.
- **Затраты на поддержку** – несмотря на то, что в случае автоматизированных тестов они меньше, чем затраты на ручное тестирование того же функционала – они все же есть. Чем чаще изменяется приложение, тем они выше.
- **Большие затраты на разработку** – разработка автоматизированных тестов это сложный процесс, так как фактически идет разработка приложения, которое тестирует другое приложение. В сложных автоматизированных тестах также есть фреймворки, утилиты, библиотеки и прочее. Естественно, все это нужно тестировать и отлаживать, а это требует времени.
- **Стоимость инструмента для автоматизации** – в случае если используется лицензионное ПО, его стоимость может быть достаточно высока. Свободно распространяемые инструменты как правило отличаются более скромным функционалом и меньшим удобством работы.
- **Пропуск мелких ошибок** - автоматический скрипт может пропускать мелкие ошибки на проверку которых он не запрограммирован. Это могут быть неточности в позиционировании окон, ошибки в надписях, которые не проверяются, ошибки контролов и форм с которыми не осуществляется взаимодействие во время выполнения скрипта.

Автоматизация тестирования

Недостатки автоматизации тестирования:

- **Повторяемость** – все написанные тесты всегда будут выполняться однообразно. Это одновременно является и недостатком, так как тестировщик, выполняя тест вручную, может обратить внимание на некоторые детали и, проведя несколько дополнительных операций, найти дефект. Скрипт этого сделать не может.
- **Затраты на поддержку** – несмотря на то, что в случае автоматизированных тестов они меньше, чем затраты на ручное тестирование того же функционала – они все же есть. Чем чаще изменяется приложение, тем они выше.
- **Большие затраты на разработку** – разработка автоматизированных тестов это сложный процесс, так как фактически идет разработка приложения, которое тестирует другое приложение. В сложных автоматизированных тестах также есть фреймворки, утилиты, библиотеки и прочее. Естественно, все это нужно тестировать и отлаживать, а это требует времени.
- **Стоимость инструмента для автоматизации** – в случае если используется лицензионное ПО, его стоимость может быть достаточно высока. Свободно распространяемые инструменты как правило отличаются более скромным функционалом и меньшим удобством работы.
- **Пропуск мелких ошибок** - автоматический скрипт может пропускать мелкие ошибки на проверку которых он не запрограммирован. Это могут быть неточности в позиционировании окон, ошибки в надписях, которые не проверяются, ошибки контролов и форм с которыми не осуществляется взаимодействие во время выполнения скрипта.

Авто или ручное тестирование?



Основное применения автоматизации

- Труднодоступные места в системе (бэкенд процессы, логирование файлов, запись в БД)
- Часто используемая функциональность, риски от ошибок в которой достаточно высоки. Автоматизировав проверку критической функциональности, можно гарантировать быстрое нахождение ошибок, а значит и быстрое их решение.
- Рутинные операции, такие как переборы данных (формы с большим количеством вводимых полей. Автоматизировать заполнение полей различными данными и их проверку после сохранения)
- Валидационные сообщения (Автоматизировать заполнение полей не корректными данными и проверку на появление той или иной валидации)
- Длинные end-to-end сценарии
- Проверка данных, требующих точных математических расчетов
- Проверка правильности поиска данных

Основное применения автоматизации

- Участки кода, исполнение которых трудно визуализировать и получить осязаемую информацию о протекающих процессах (back-end процессы, занесение в базу данных, занесение логов в файл).
- Функциональность продукта, которая будет использоваться наиболее часто и возникновение ошибок которой связано с достаточно высоким риском.
- Типовые часто выполняемые операции, которые обычно связаны с обработкой данных.
- Сообщения об ошибках. Требуется автоматизация разнесения некорректных данных по соответствующим полям и тестирование корректности проверки правильности данных и сообщений об ошибках.
- Комплексная проверка поведения всей системы, как целостного объекта (end-to-end testing).
- Проверка числовых массивов, нужных для достоверных математических операций.
- Тестирование корректности отображаемых результатов поиска в ответ на запрос по нужным данным.

Всё зависит от предъявляемых к проверяемой системе требований, возможностей, которые позволяет реализовать выбранный для автоматического тестирования инструмент.

План тестирования определяется международным стандартом **IEEE 829-1983**. В нем должны быть предусмотрены как минимум три раздела содержащие, следующие описания:

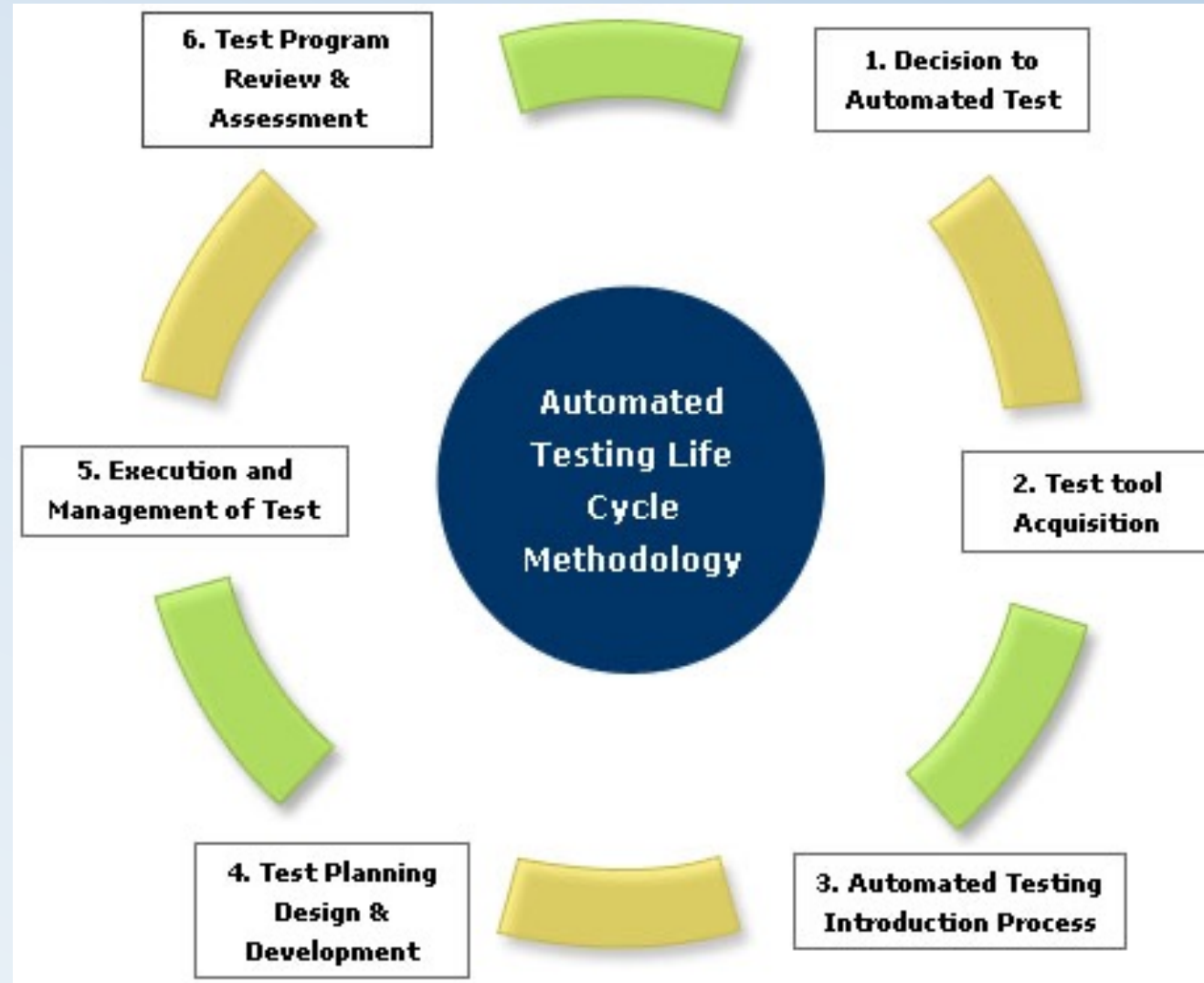
1. что будет тестироваться (тестовые требования, тестовые варианты);
2. какими методами и насколько подробно будет тестироваться система;
3. план-график работ и требуемые ресурсы (персонал, техника).

Этапы тестирования

Вид тестирования	Стадия, этап	Объект	Критерий
Структурное, надежности	Разработка	Компоненты	Покрытие ветвлений, функции
Сборочное	Разработка	Подсистемы	Функциональность, степень проверки компонентов
Функциональное	Разработка	Система в целом	Соответствие функциональным требованиям ТЗ
Регрессионное	Разработка, сопровождение	Система в целом	Проверка качества внесения изменений
Нагрузочное	Разработка, сопровождение	Система в целом	Оценка статистических характеристик системы, соответствие ТЗ, ТТХ, подбор конфигурации оборудования
Стрессовое	Разработка, сопровождение	Система в целом	Корректность работы системы при предельных нагрузках

Методологии

Automated Testing Lifecycle Methodology



Методологии

Behavior-Driven Development

