# Тестирование производительности

1. **Нагрузочное тестирование** Load Testing *Достаточно ли быстро работает система?*
2. **Тестирование стабильности** Stability Testing *Достаточно ли надежно работает система на большом интервале времени?*
3. **Тестирование отказоустойчивости** Failover Testing *Сможет ли система переместиться сама на другой сервер в случае сбоя основного сервера?*
4. **Тестирование восстановления** Recovery Testing *Как быстро восстановится система?*
5. **Стрессовое тестирование** Stress Testing *Что произойдет при незапланированной нагрузке?*
6. **Тестирование объемов** Volume Testing *Как будет работать система, если объем базы данных увеличится в 100 (1000…) раз?*
7. **Тестирование масштабируемости** Scalability Testing *Как будет увеличиться нагрузка на компоненты системы при увеличении числа пользователей?*
8. **Тестирование потенциальных возможностей** Capacity Testing *Какое количество пользователей может работать?*
9. **Конфигурационное тестирование** Configuration Testing *Как заставить систему работать быстрее?*
10. **Тестирование сравнения** Compare Testing *Какое оборудование и ПО выбрать?*

# Нагрузочное тестирование

это автоматизированное тестирование, имитирующее работу определенного количества бизнес пользователей на каком-либо общем (разделяемом ими) ресурсе.

Часто в русскоязычных источниках по НТ понимают все виды испытаний

# Терминология

1. **Виртуальный пользователь** (Virtual User) - программный процесс, циклически выполняющий моделируемые операции
2. **Итерация** (Iteration) – это один повтор выполняемой в цикле операции
3. **Интенсивность выполнения операции** (Operation Intensity) - частота выполнения операции в единицу времени, в тестовом скрипте задается интервалом времени между итерациями
4. **Нагрузка** (Loading) - совокупное выполнение операций на общем ресурсе (тр./сек, хитов/сек)
5. **Производительность** (Performance) - количество выполняемых операций за период времени (N операций за M часов)
6. **Масштабируемость приложения** (Application Scalability) - пропорциональный рост производительности при увеличении нагрузки
7. **Профиль нагрузки** (Performance Profile) - это набор операций с заданными интенсивностями, полученный на основе сбора статистических данных либо определенный путем анализа требований к тестируемой системе
8. **Нагрузочной точкой** называется рассчитанное (либо заданное Заказчиком) количество виртуальных пользователей в группах, выполняющих операции с определенными интенсивностями

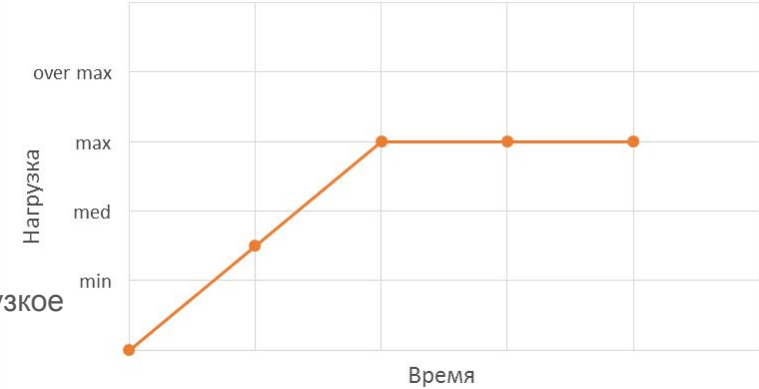
# Производительность и нагрузка

Производительность  определяется следующими факторами:

- скоростью работы программного обеспечения;
- скоростью работы аппаратного обеспечения;
- скоростью работы сети.

Во время тестирования могут осуществляться следующие операции, позволяющие более точно измерить производительность и определить “узкое место” системы:

- измерение времени выполнения выбранных операций при определенных интенсивностях выполнения этих операций;
- определение количества пользователей, одновременно работающих с приложением;
- определение границ приемлемой производительности при увеличении нагрузки (при увеличении интенсивности выполнения этих операций).



# Цели нагрузочного тестирования

1. Оценка производительности и работоспособности приложения на этапе разработки и передачи в эксплуатацию
2. Оценка производительности и работоспособности приложения на этапе выпуска новых релизов, патч-сетов
3. Оптимизация производительности приложения, включая настройки серверов и оптимизацию кода
4. Подбор соответствующей для данного приложения аппаратной (программной платформы) и конфигурации сервера

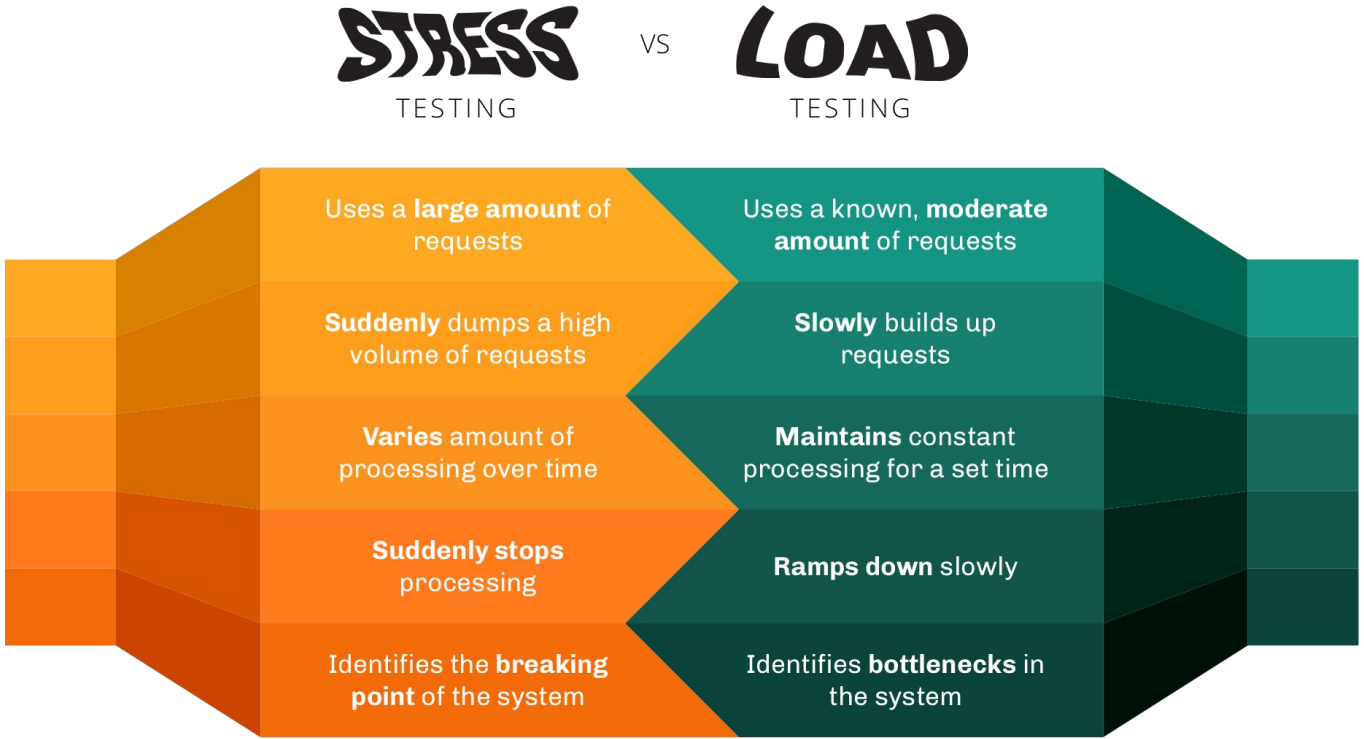
# Уточняем цели

Если интересует **исследование производительности приложения** - тестирование производительности (Performance Testing)

Если целью является понимание **насколько приложение устойчиво в режиме длительного использования** - тестирование стабильности (Stability Testing)

Стресс тестирование (Stress Testing) имеет своей целью проверить возвращается ли система после запредельной нагрузки (и как скоро) к нормальному режиму, также целями стрессового тестирования могут быть проверки поведения системы в случаях когда, один из серверов приложения в пуле перестаёт работать, аварийно изменилась аппаратная конфигурации сервера базы данных и т.д.

# Load VS Stress



# Этапы нагрузочного тестирования

Анализ требований и сбор информации о тестируемой системе

Конфигурация тестового стенда для нагрузочного тестирования

Разработка модели нагрузки

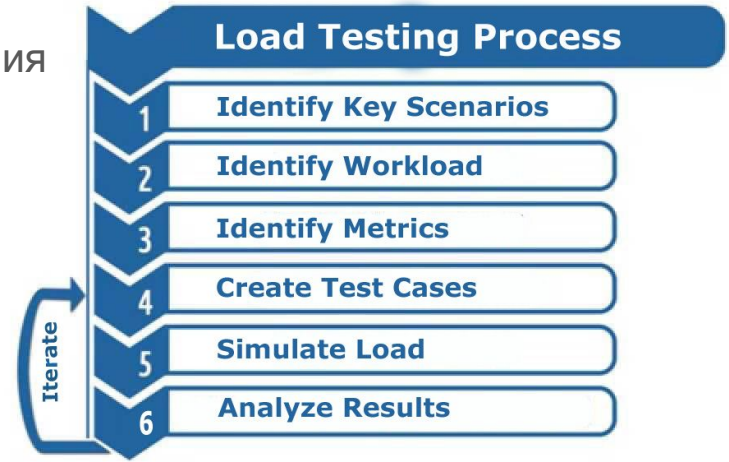
Выбор инструмента для нагрузочного тестирования

Создание и отладка тестовых скриптов

Проведение тестирования

Анализ результатов

Подготовка, отправка и публикация отчета



# Модель нагрузки

- список тестируемых операций
- интенсивность выполнения операций
- зависимость изменения интенсивности выполнения операций от времени

В список тестируемых задач должны войти операции, критичные с точки зрения бизнеса, а также с технической точки зрения:

- Критичными с точки зрения бизнеса являются операции, скорость выполнения которых, реально влияет на производительность бизнес процесса.
- Критичными с технической точки зрения являются ресурсоемкие операции, требующие большое количество памяти и серьезно задействующие процессор, создающие значительный сетевой трафик.

# Метрики

1. Время реакции
2. Пропускная способность
3. Метрики, специфичные для аппаратного обеспечения
4. Метрики, связанные с базами данных

# Виды нагрузочных испытаний

1. Нагрузочное тестирование вручную

   Ручное нагрузочное тестирование — это когда система оценивается без автоматизированных инструментов нагрузочного тестирования, то есть симулированные пользователи создаются вручную.

2. Инструменты для внутреннего тестирования

   Поскольку нагрузочное тестирование является непрерывным процессом, особенно в период роста, многие организации предпочитают создавать собственные средства автоматизации нагрузочного тестирования.

3. Инструменты тестирования с открытым исходным кодом

   Существует множество инструментов тестирования с открытым исходным кодом. Будучи программами с открытым исходным кодом, они бесплатны для использования, предлагают широкие возможности для модификации и опираются на мощную поддержку сообщества.

4. Инструменты автоматизации нагрузочного тестирования корпоративного класса

   Корпоративные инструменты тестирования предоставляют различные возможности для масштабирования в соответствии с потребностями сайтов электронной коммерции, сервисных платформ и профессиональных организаций всех типов.

# Популярные инструменты нагрузочного тестирования

| № | Название | Описание | Язык программирования |
|---|---|---|---|
| 1 | LoadRunner (Enterprise, Professional, Cloud) | Линейка платных решений от компании Micro Focus (один из старейших инструментов НТ, принадлежавший в разные годы фирмам Mercury и Hewlett Packard). | Основной язык для написания скриптов – C, но также можно программировать на Java, C#, JavaScript. |
| 2 | JMeter и BlazeMeter | Инструмент для НТ с открытым исходным кодом и коммерческий продукт на его основе с бесплатными библиотеками. | Непосредственно скрипты создаются через оконный интерфейс визуальным программированием, но есть возможность расширять логику, например, с помощью Groovy, или писать свои плагины на Java. |
| 3 | Яндекс.Танк | Собственная разработка компании Яндекс с открытым исходным кодом. | В зависимости от применяемого генератора нагрузки способ написания скриптов может быть разный, в том числе с использованием скриптов JMeter. |
| 4 | Gatling | Бесплатный инструмент с открытым исходным кодом, набирающий популярность. | Скрипты пишутся на Scala. |

# Преимущества

1. Предотвращает простои и сбои в работе приложений
2. Мониторинг стандартов производительности
3. Сокращение расходов
4. Повышение эффективность
5. Соблюдение соглашения об уровне обслуживания (SLA)
6. Планирование мощностей

# Проблемы и ограничения

**“Неосязаемость”**

Нагрузочное тестирование не обязательно является самым заметным инструментом, поскольку одним из его основных преимуществ является выявление потенциальных проблем до того, как они возникнут в реальной ситуации. Многие негативные финансовые и иные последствия, связанные с простоем сайта и сбоями в работе приложений, просто не реализуются.

**Сложность**

Как инструменты нагрузочного тестирования с открытым исходным кодом, так и собственные инструменты могут иметь высокий барьер для входа на техническом уровне. В зависимости от размера и сложности организации, у них может не быть сотрудников или ресурсов для проведения нагрузочного тестирования.

# Эффективное нагрузочное испытание

1. Использует реалистичные сценарии

   Сценарии тестирования должны как можно ближе напоминать реальное поведение пользователей. Внимательно изучите поведение пользователей. Почему они используют ваше приложение? Какие типы устройств они используют для доступа к нему?

2. Не начинается с нуля

   Многие тестировщики начинают тестирование с нулевой нагрузки и постепенно добавляют симулированных пользователей. Хотя в этом методе есть определенная польза, не забывайте также проводить тестирование, когда система уже находится под нормальной нагрузкой. Это поможет избежать ложных срабатываний и приведет к более точным результатам, поскольку в реальном мире ваша система редко, если вообще когда-либо, будет иметь нулевую нагрузку.

3. Использует реальные данные

   Как показывает предыдущая практика, чем качественнее данные, полученные перед тестированием, тем полезнее результаты тестирования. Для разработки реалистичных сценариев обратитесь к данным, полученным ранее с помощью инструментов мониторинга.

4. Анализ и повторение

   После нагрузочного тестирования ваша команда захочет определить узкие места и соответствующий им код.