

SIEMENS



SQA
DAYS 2008

Компонентная среда разработки инструментария нагрузочного тестирования

Евгений Рачинский. СПбГУ в сотрудничестве с Siemens Corporate Technology

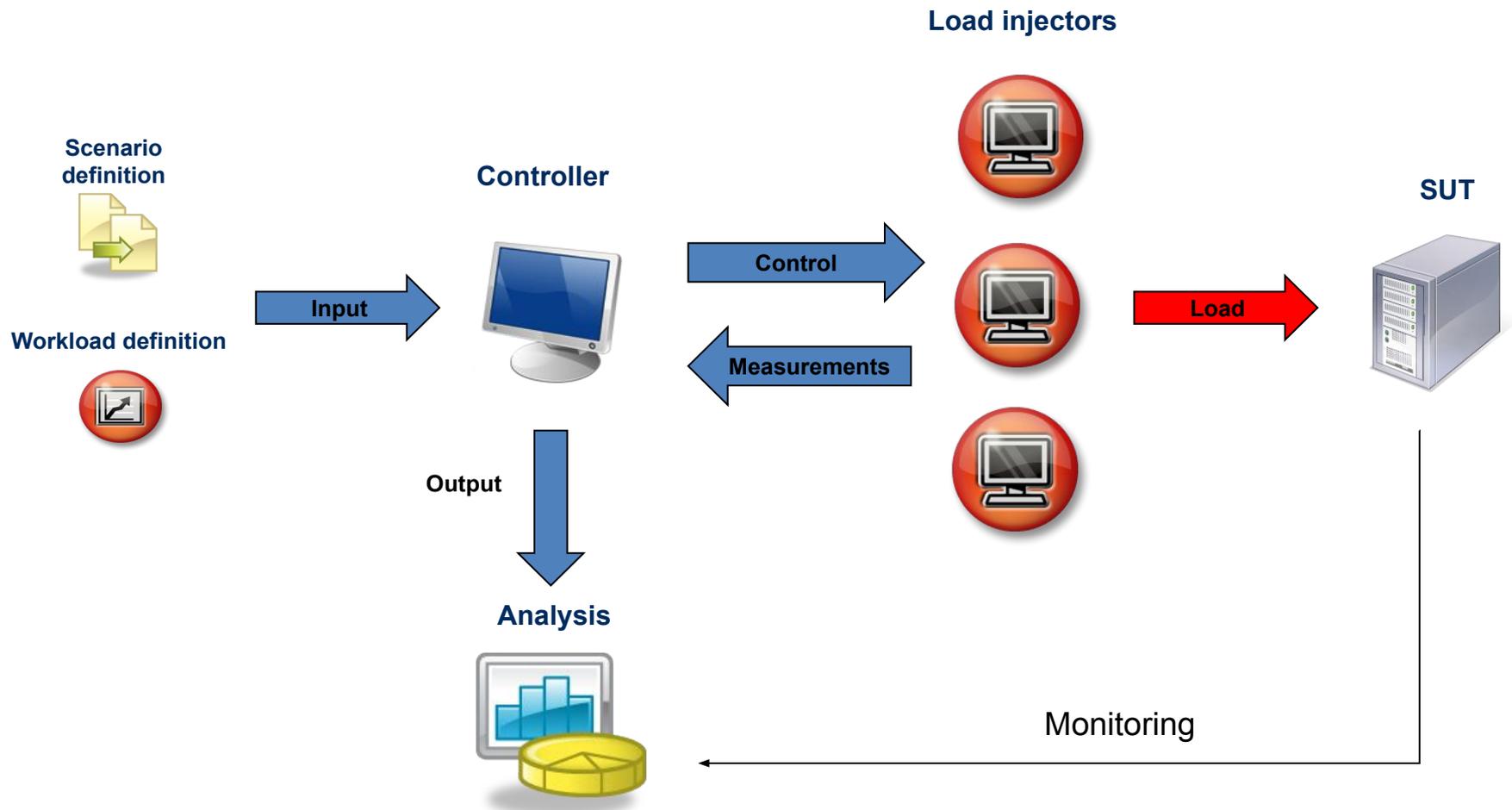
Нагрузочное тестирование

- Изучение поведения многопользовательской системы под нагрузкой
- Цели:
 - Оценка характеристик производительности системы под нагрузкой
 - Поиск узких мест в системе
 - Планирование производительности
- Средства
 - Моделирование нагрузки
 - Инструментарий генерации нагрузки и измерения показателей
 - Методы анализа результатов

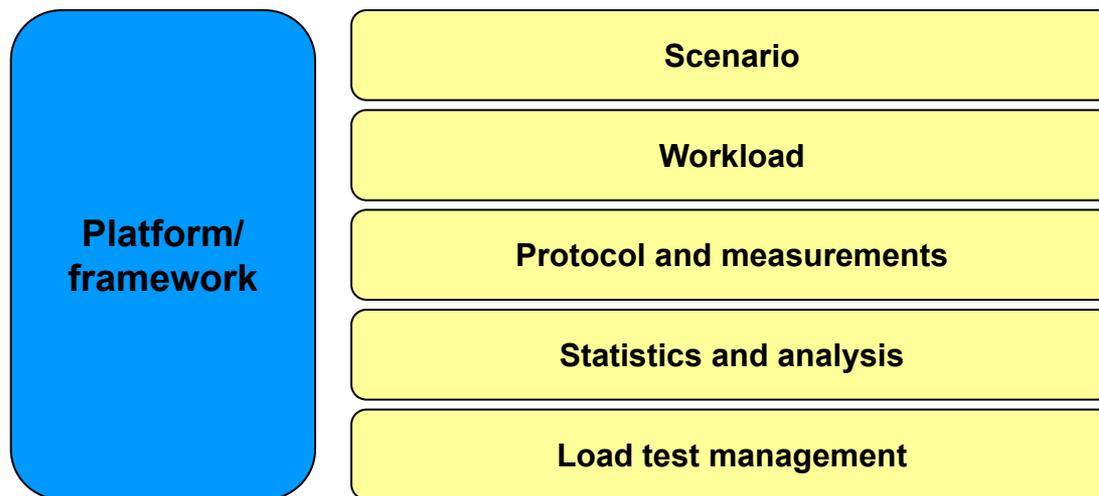
Инструментарий

- Существующие средства нагрузочного тестирования не обладают достаточной гибкостью и адаптивностью
- **Решение: создание платформы для разработки инструментария нагрузочного тестирования**
- Области применения
 - Нагрузочное и стресс тестирование с использованием нестандартных протоколов
 - Поддержка статистических методов в нагрузочном тестировании
 - Автоматизация нагрузочного тестирования
 - Поддержка continues integration process

Типовая архитектура



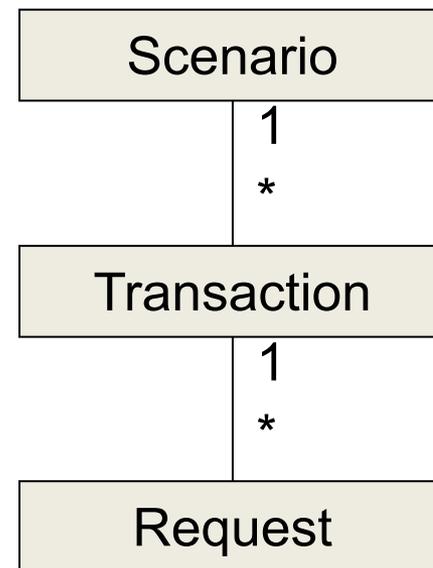
Области расширения



- Компонентная среда (OSGi/Java)
- Архитектура основанная на plug-ins
- API, точки расширения

Сценарий

- Симулирует поведение пользователей
- Java классы
 - Сценарий
 - Транзакция (шаг)
 - Запрос
- Иерархичность, модульность
- Идентификация частей сценария
- Генерация кода сценариев
- Отладка (Eclipse)



Сценарий (пример)

```
public class Service1Scenario extends
WebScenario {

Transaction1 transaction1 = new Transaction1();
Transaction2 transaction2 = new Transaction2();

public void run() {

    runTransaction(transaction1);
    sleep(1000);
    runTransaction(transaction2);

}

}
```

```
public class Transaction1 extends WebTransaction {
Request1 request1 = new Request1();

public void run() {
    runRequest(request1);
}

}
```

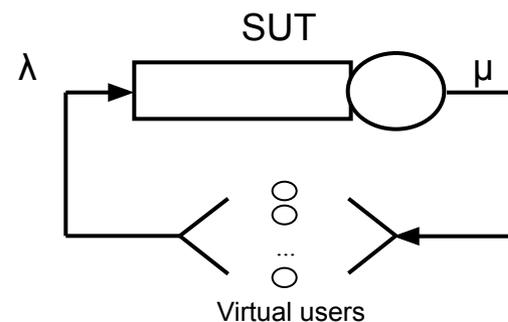
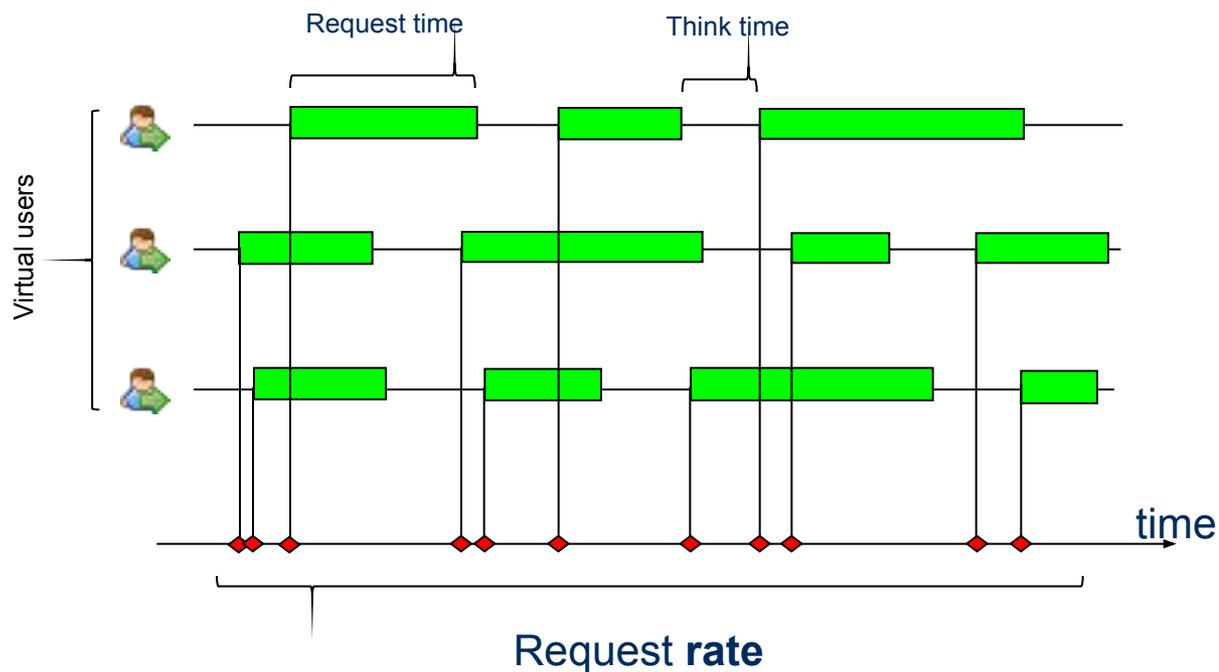
```
public class Request1 extends WebRequest {

public void run() {
    HttpResponse response = null;

    response =
        getContext().
            getClient().execute(url );
}

}
```

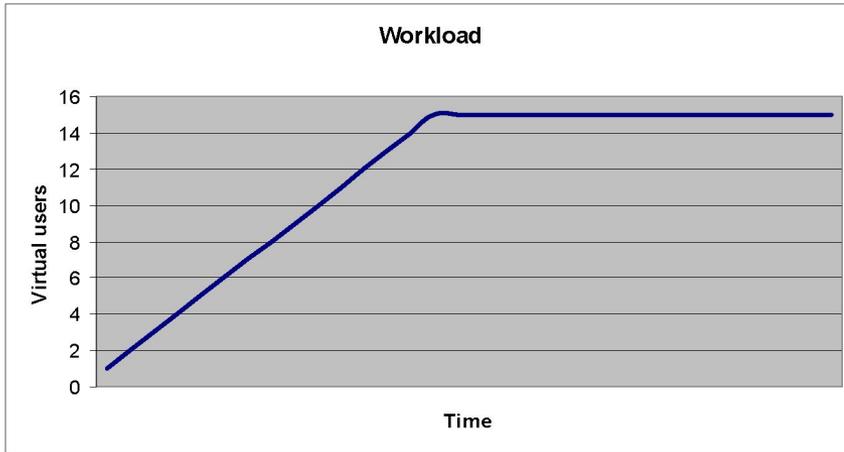
Нагрузка



- Нагрузка это частота обращений к системе
- Определяется через
 - количество виртуальных пользователей **N**
 - среднее значение и распределение времени ответа

Определение нагрузки (пример)

$$V(t) = \min\{V_{\max}, ((t / Interval) + 1)V_{increment}\}$$

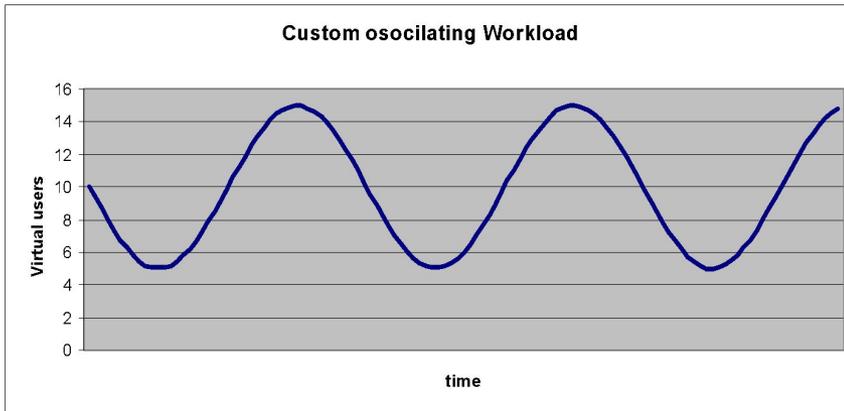


```
int maximumVirtualUsers = 15;
int incrementInterval = 60;
int incrementVirtualUsersBy = 1;

public IThinkTime getThinkTimeOnTime(long time) {
    return new ConstantThinkTime(1000);
}

public int getVirtualUserNumberOnTime(long time) {
    int vu = Math.min(
        maximumVirtualUsers,
        (int)((time/incrementInterval)+1)*incrementVirtualUsersBy
    );
    return vu;
}
```

$$V(t) = V_c + A \sin(freq * t)$$



```
int Amplitude = 5; //virtual users
double Frequency = 1/60; //virtual users per second
int VerticalShift = 10; //oscillate around

public IThinkTime getThinkTimeOnTime(long time) {
    return new ExponentialThinkTime(1000);
}

public int getVirtualUserNumberOnTime(long time) {
    return (int)(VerticalShift + Amplitude*Math.sin(Frequency*time));
}
```

Поддержка сетевых протоколов и измерений

- Любой протокол, имеющий клиентские Java библиотеки
- Регистрация значений:
 - Время исполнения
 - Ошибки протокола/приложения
 - Проткол-специфические измерения (размер пакета, время установления соединения, DNS время и т.п.)
- Инструментирование библиотек протоколов
- Поддержка протоколов интегрируется в базовые классы сценариев

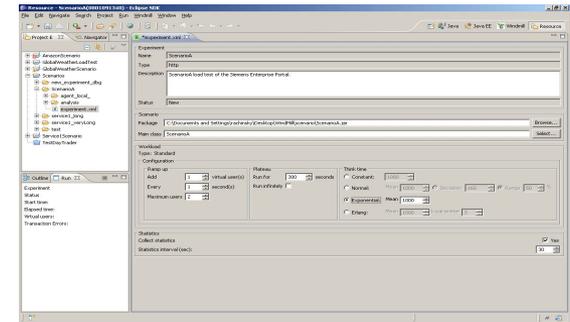
Run-time статистика и журналирование

- Измерения собираются в статистики:
 - Среднее значение
 - Частота событий
 - Счетчики
 - Определенная пользователем
- Масштабирование значений
 - Среднее значение на разных масштабах времени
- Расширяемый «движок» статистики
 - Filter/Pipe design pattern
- Журналирование
 - Набор CSV файлов, Apache Derby
 - Формат определен протоколом

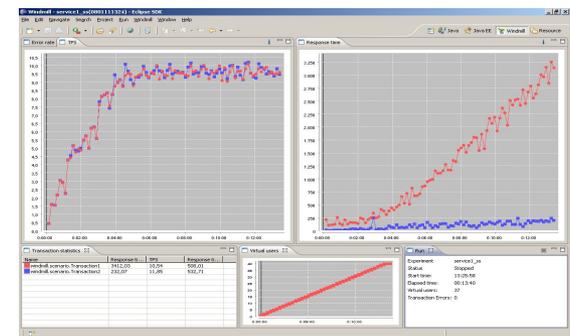
Управление тестом

- Пользовательский интерфейс
 - Eclipse RCP UI
 - Command line
- Подготовка теста
 - Eclipse Java IDE, PDE
 - Рабочее окружение Eclipse
- Исполнение теста и мониторинг
 - Графики статистики
- Расширяемость
 - Eclipse plugin-ins
 - Chart API (JFree chart)
 - ANT tasks

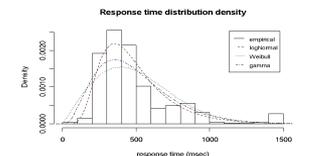
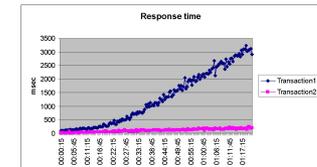
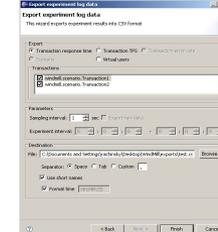
Load test definition



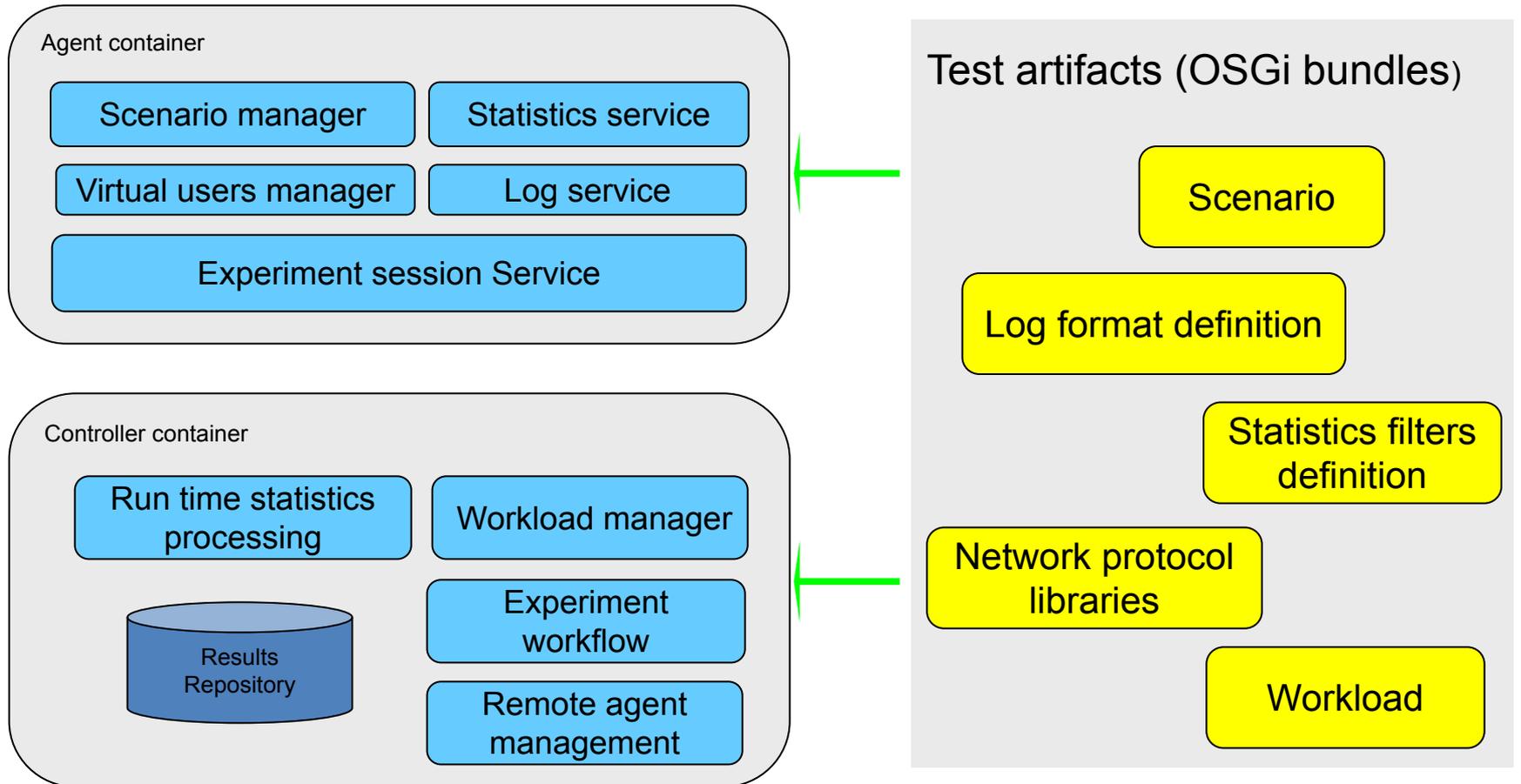
Load test controller



Results export



Компоненты платформы

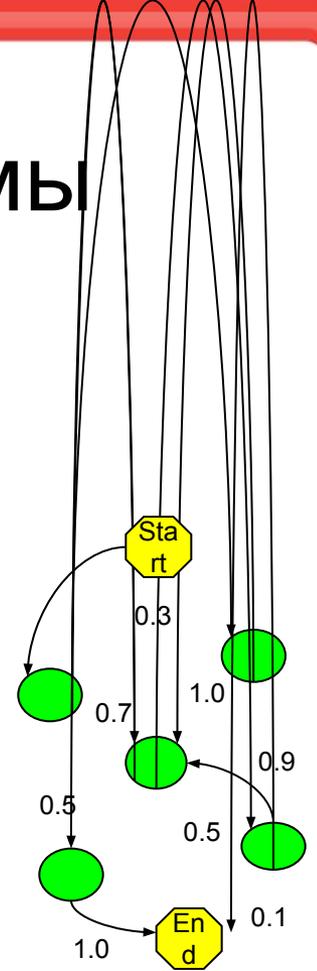


Замечания по реализации

- Высокая производительность агентов
- Сложность точного измерения времени в Java (msec, nanosec)
- Синхронизация потоков виртуальных пользователей
- Синхронизация времени в распределенной среде
- Интенсивный поток данных (измерений)
- Сложность использование Java аннотаций и рефлексии

Приложения платформы

- Стандартное нагрузочное тестирование
 - HTTP, SOAP, RMI
- Сложные сценарии
 - Различные протоколы в одном сценарии
 - Симуляция вероятностного поведения пользователя (CBMG)
 - Генерация кода сценариев (из трасс или моделей)
- Нагрузочное тестирование и регулярная сборка
 - ANT task для определения и запуска теста

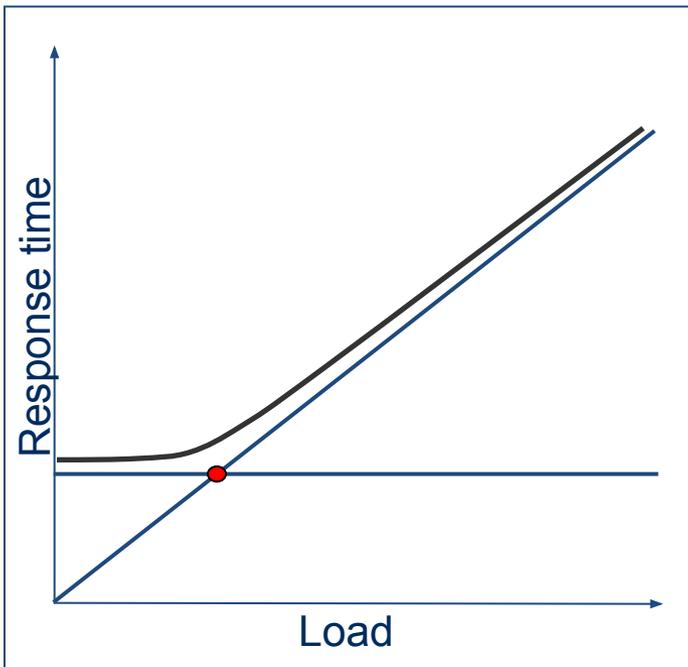


Приложения платформы (2)

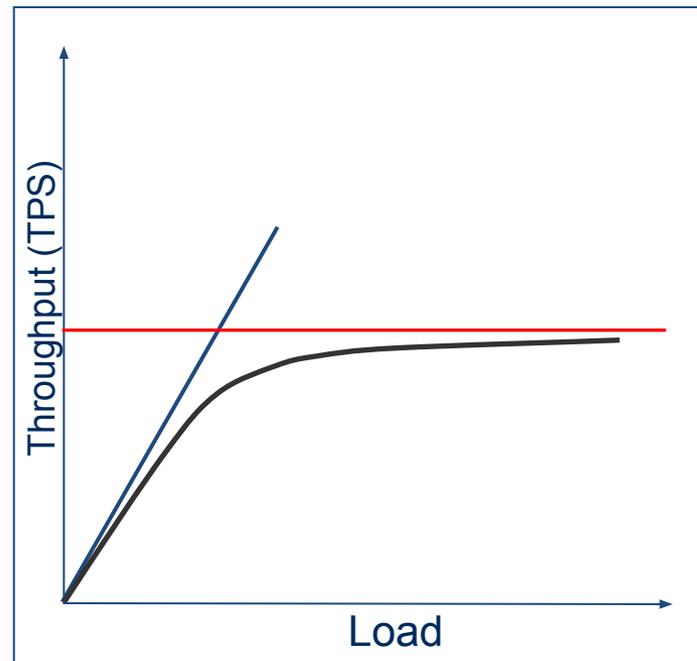
- Генерация заданной нагрузки
 - Пуассоновский поток запросов
- Автоуправление нагрузкой в зависимости от текущих показателей производительности
 - Измерение среднего значения времени ответа с заданным доверительным интервалом
 - Автоматический поиск максимальной пропускной способности системы (max TPS)
- Симуляция пульсирующей нагрузки

Максимальная пропускная способность системы

Average response time



Throughput (TPS)



Анализ результатов

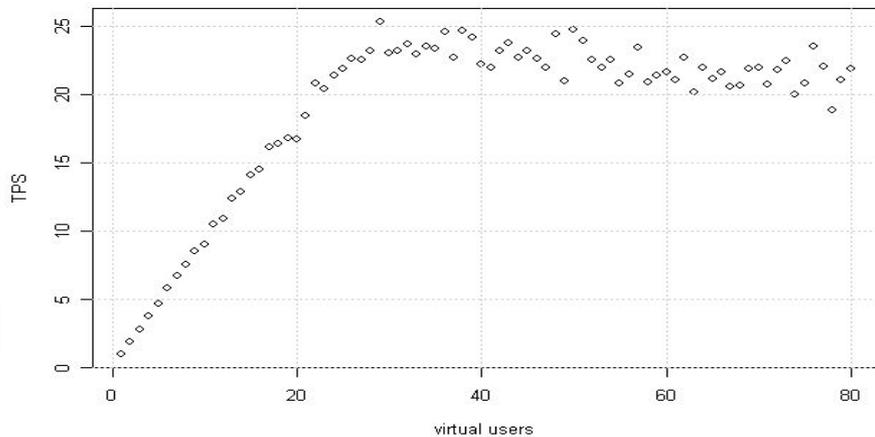
- Пакеты статистической обработки
 - S-Plus (R statistics)
- Дисперсионный анализ
 - сравнение производительности альтернативных конфигураций системы
- Корреляционный анализ
- Вывод параметров аналитических моделей (очереди)
- Построение моделей «черного ящика»

Спасибо за внимание!

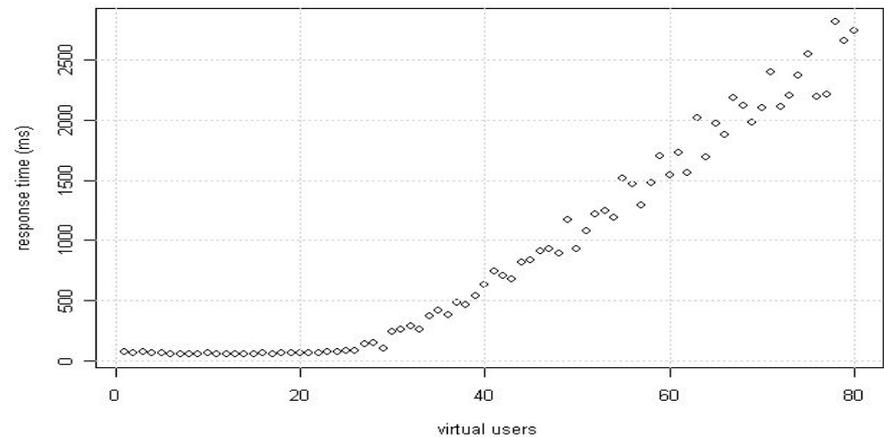
Backup slides

Descriptive statistics example – KPI VS. load

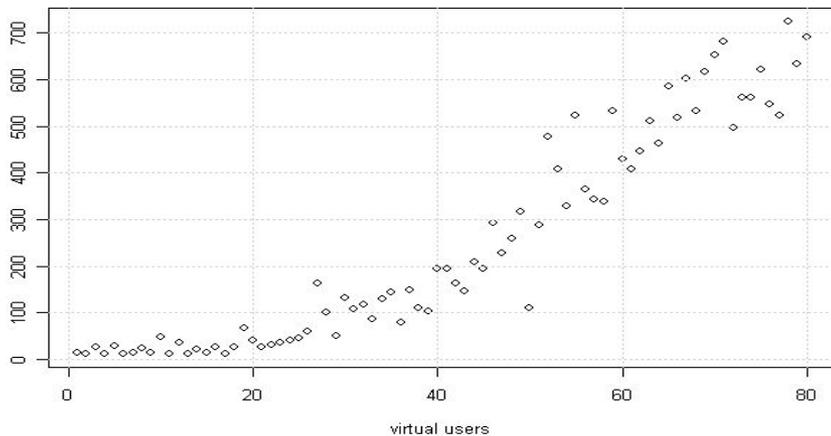
Transactions per second



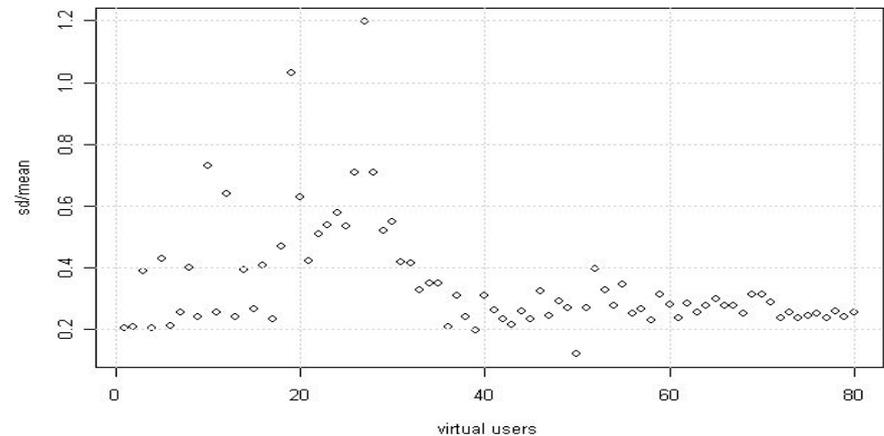
Average response time



Response time standard deviation



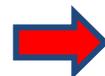
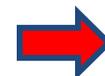
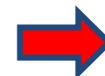
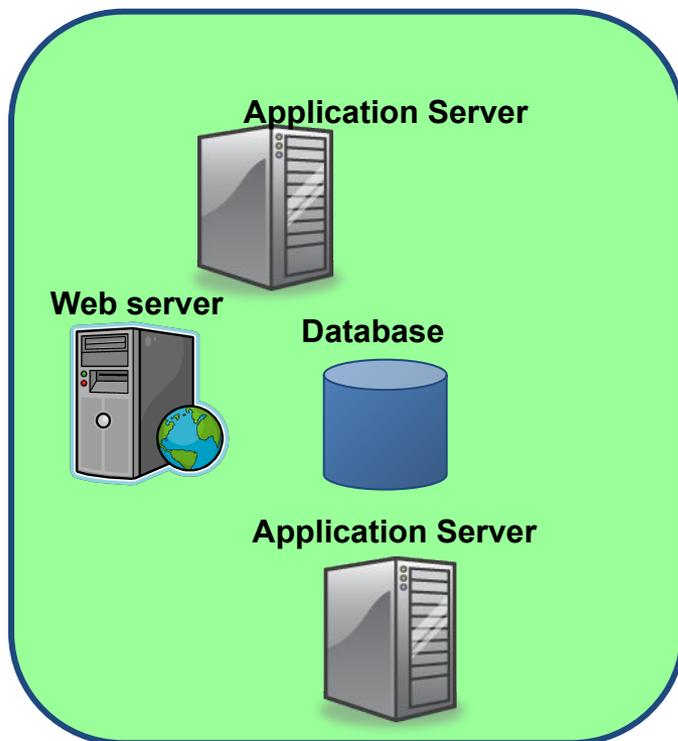
Response time coefficient of variation



Тестирование производительности распределенных систем

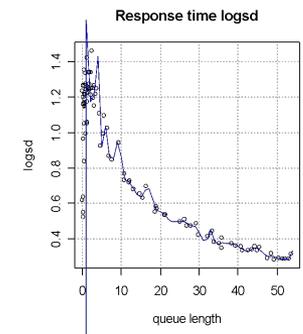
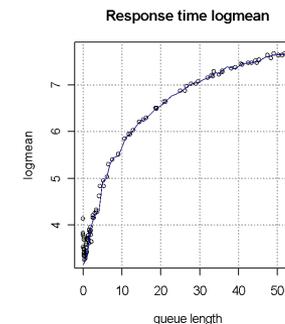
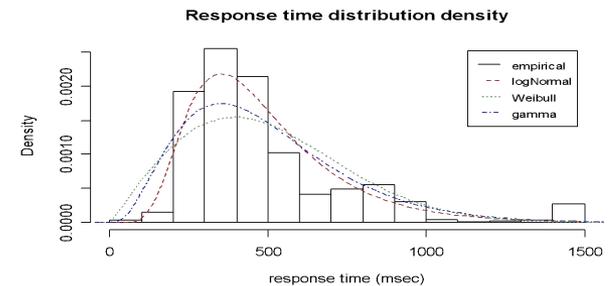
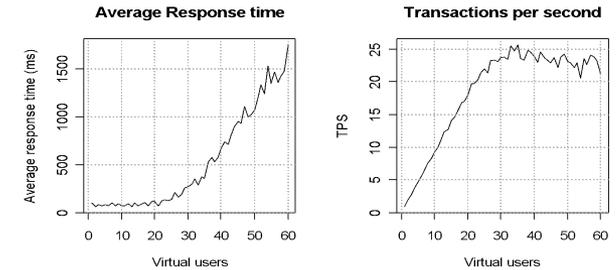
Доступно для тестирования под
нагрузкой

Недоступно для тестирования под
нагрузкой



Approach overview

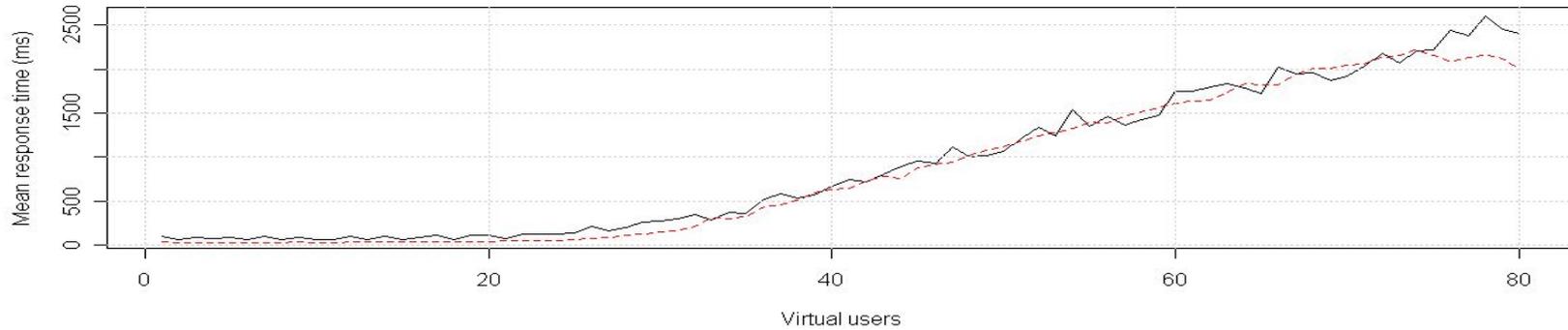
- Perform load test and collect measurements
 - Windmill + dynamic workload
- Determine response time distribution
 - Statistical tests
 - Lognormal, Gamma, Weibull distributions
- Fit response time distribution parameters
 - Non-parametric models (cubic splines)
- Setup runtime simulator



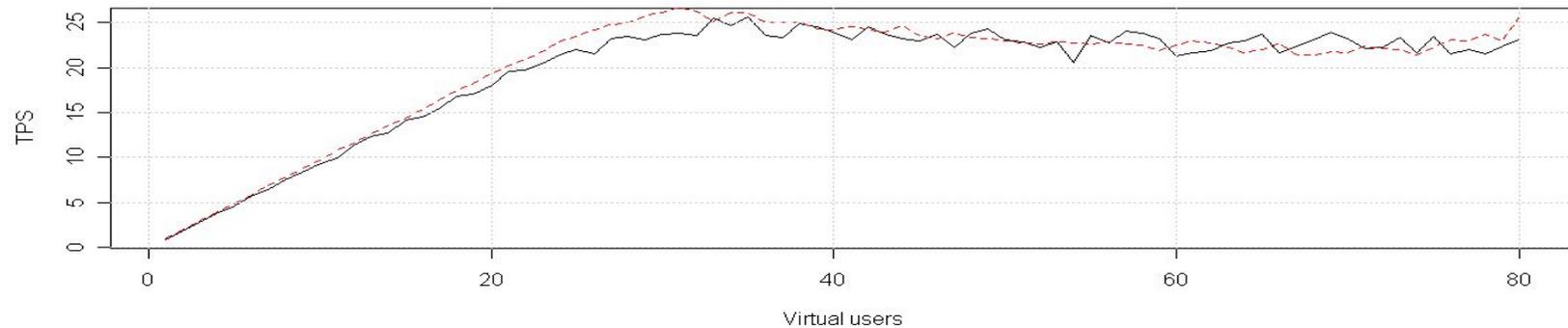
Results comparison

Legend: — SUT with real service — SUT with emulated service

Mean Response time



Transactions per second



time

