

ВИЗУАЛИЗАЦИЯ МЕХАНИЗМА УПРАВЛЕНИЯ ТРАНЗАКЦИЯМИ КАК ИНСТРУМЕНТ ОБУЧЕНИЯ

Калашников Б.Д., Моисеенко С.И.

ФГАОУ ВО «Южный федеральный университет»,

Институт высоких технологий и пьезотехники

г. Ростов-на-Дону

E-mail: bogdan.dm1995@yandex.ru, smois77@gmail.com

Способы взаимодействия между БД в гетерогенных системах

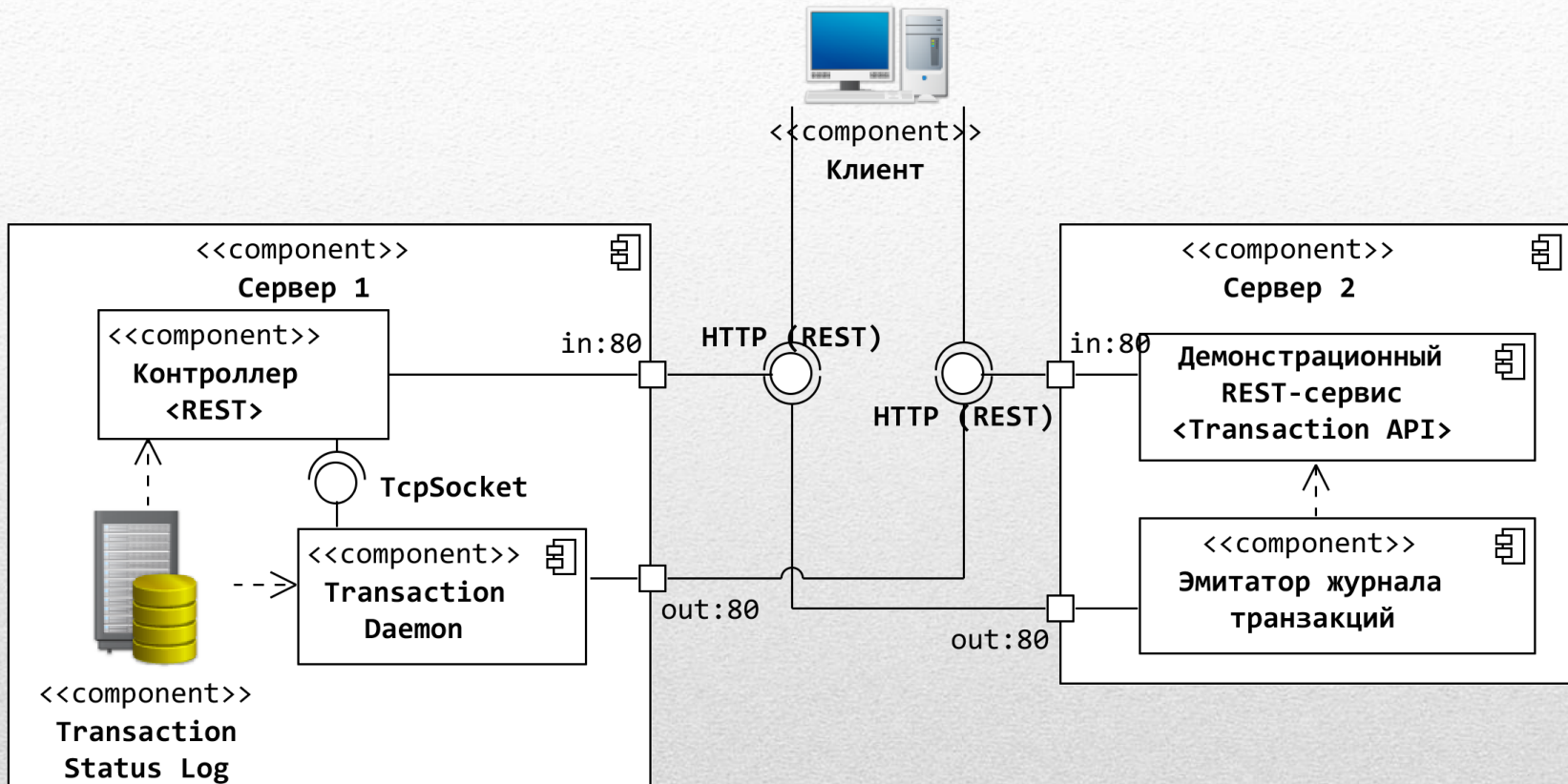


REST

REST (сокр. от англ. *Representational State Transfer* — «передача состояния представления») — архитектурный стиль взаимодействия компонентов распределённого приложения в сети. REST представляет собой согласованный набор ограничений, учитываемых при проектировании распределённой гипермедиа-системы. Передача данных обычно осуществляется в одном из определенных форматов (чаще всего JSON, реже XML или YAML)

Ресурс	POST create	GET read	PUT update	DELETE delete
/books	Создание новой книги	Список книг	-	-
/books/id	-	Получение книги по id	Обновление полей книги	Удаление книги

UML-диаграмма компонентов

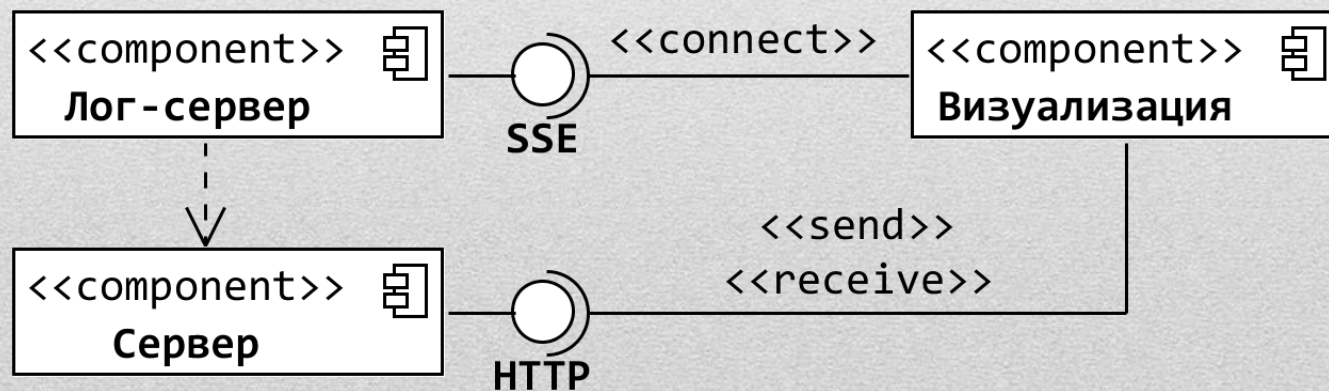


Управлением распределенными транзакциями занимается контроллер, исполняющий роль центрального узла системы. С помощью контроллера реализуется протокол двухфазной фиксации, гарантирующий согласованность распределенных данных.

Принцип работы визуализации


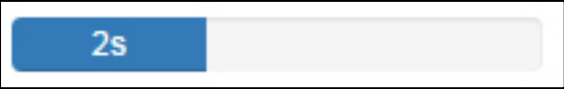

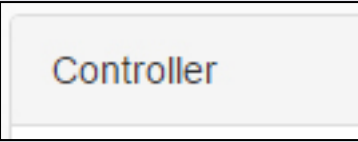
- Для исследуемого компонента системы в отдельном потоке запускается локальный *HTTP-сервер*, предоставляющий единственный *ресурс* – очередь сообщений, основанную на *Server-sent events*.
- Внутри компонента доступен логгер, привязанный к этому HTTP-серверу.
- При логгировании сообщения оно отправляется через *SSE-соединение* веб-странице.

Итого веб-страница работает как удаленный логгер, на котором средствами JS происходит визуализация.



GUI

[illegible]

	<p>«Лампочки», отражают состояние узла.</p>
	<p>Прогресс-бары, отражают состояние медленно текущих процессов.</p>
	<p>Конечные узлы системы. Каждый является отдельным веб-сервером и поддерживает своё SSE-соединение.</p>
	<p>Контроллер, координирует выполнение транзакции и поддерживает информацию о состоянии агентов транзакции.</p>
<p>Event log</p>	<p>Журнал событий, позволяет просмотреть данные в том виде, в котором они пришли с сервера.</p>
<p>Transaction generator</p>	<p>Позволяет генерировать JSON-данные транзакции</p>

Генератор транзакций

[2 Phase Commit](#) [Visualisation](#) [Event log](#) [Transaction generator](#) [About](#)

Parameters

Global timeout

100000

Number of Services

3

Service local timeout

60000

to

90000

JSON

```
{
  "timeout": 100000,
  "actions": [
    {
      "_id": "Service #0 GET catholicos",
      "service": {
        "url": "http://localhost:5010/api",
        "timeout": 78502
      },
      "url": "/catholicos",
      "method": "GET",
      "data": {},
      "headers": {}
    },
    ...
  ]
}
```

Go!

<http://localhost:5000/api/alpha/transactions>

Так как контроллер также является REST-сервисом, то создание транзакции – это HTTP-запрос, в который в формате JSON упакован набор подзапросов, составляющих транзакцию. В левой панели указываются параметры транзакции. Саму транзакцию, представленную в формате JSON, можно сформировать на правой панели, с неё же можно запустить выполнение транзакции.

Использованные технологии

- Python (Flask, gevent) – backend
- SSE – server-push технология
- JS (ES6, JQuery, json-formatter-js), Bootstrap – frontend

Литература

1. Web-сервисы RESTful: основы - <http://www.ibm.com/developerworks/ru/library/ws-restfu/>
2. Коннолли Т., Бегг К., Страчан А. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 3-е изд.:пер. с англ.: Уч.пос. – М.: Изд.дом «Вильямс», 2003. – 1436 с.
3. Введение в JSON - <http://www.json.org/json-ru.html>