УДК 621.3

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ НАДСТРОЙКИ IP MULTIMEDIA SUBSYTEM AS A SERVICE С ПОМОЩЬЮ STAGED-EVENT DRIVEN ARCHITECTURE

Слепец Ю.Ю., Гаттуров В.К.
Институт телекоммуникационных систем НТУУ «КПИ», Украина
E-mail: yuriy.slipets@gmail.com

Software implementation of IP Multimedia Subsystem as a Service setup by Staged-event driven architecture

This article shows the one of many ways for IP Multimedia Subsystem (IMS) cloud implementation using the Staged-event driven architecture (SEDA) using it's advantages for creating big data services.

На данном этапе развития сетей связи концепция IMS не пользуется большой популярностью. Причиной тому, на момент создания спецификаций 3GPP были следующие факторы:

- Необходимость поддержки протокола IPv6;
- Отсутствие терминалов IMS;
- Необходимость модернизации систем Operation Support System (OSS) и Business Support System (BSS);
- Отсутствие поддержки не SIP-приложений.

Устранение каждой из этих проблем влекло за собой огромные затраты на реорганизацию уже построенных GSM и UMTS сетей. Но в процессе эволюции мира телекоммуникаций и широкое внедрение интеллектуальных технологий, дало возможность преобразовать автономное физическое IMS решение в программную надстройку на уже существующей сети, используя концепцию облачных вычислений. Так совсем недавно появилась еще одна разновидность облачных бизнес-моделей построения дополнительных сервисов - IP Multimedia Subsystem as a Service (IMSaaS).

В случае облачной реализации IMS будет происходить слияние основного функционала (P-CSCF, S- CSCF, I-CSCF, и HSS) (рис 1.) в один облачный ресурс. Интеллект каждой из функций реализован на отдельной виртуальной машине IMS-Virtual Machine (IMS-VM). [1, 3]

Основная проблема модели IMSaaS заключается в создании взаимодействий между функциональными объектами сети, которые будут соответствовать, стандартизированным в 3GPP, интерфейсам взаимодействия.

Предложенный способ реализации облачных функциональных объектов и взаимодействий между ними основан на использовании Staged-event driven architecture (SEDA) – архитектуры для программной реализации систем массового обслуживания.

В этой модели, каждый этап воплощает в себе надежный, многократно используемый программный компонент, который выполняет подмножество

обработок запросов. Выполняя контроль допуска для каждой очереди событий (вызовов/заявок), услуга может быть подготовлена для сильной загрузки, предотвращая перегрузку ресурсов, когда превышается выделенный на службу ресурс. [2]

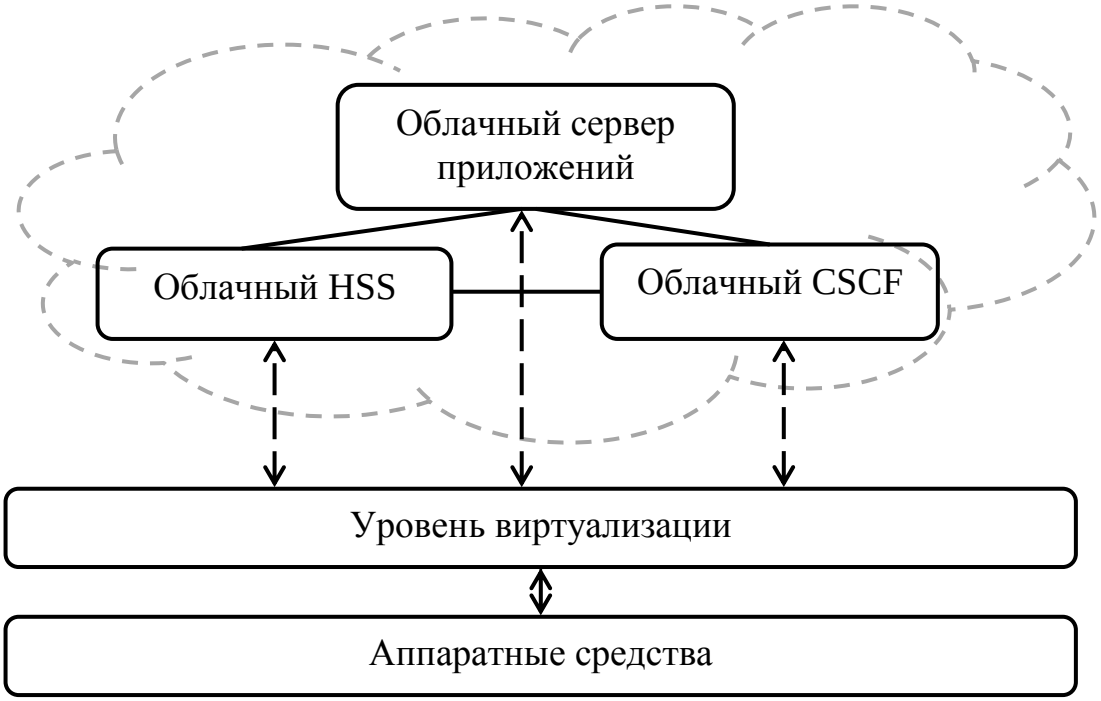

Рис. 1 Иллюстрация упрощенной IMSaaS [3].

SEDA использует динамическое управление временными параметрами (например, параметрами планирования каждого этапа) для управления нагрузкой, например, путем адаптивного сброса нагрузки.

Основной единицей обработки в SEDA является ступень. Ступень представляет собой автономный компонент приложения, который состоит из обработчика событий (event handler), входящей очереди событий (event queue), и пула потоков (thread pool), как показано на рис 2. Каждый каскад управляется одним или несколькими контроллерами, которые влияют на занятие ресурсов, планирование, распределение потоков, и управление доступом. Работа потоков в ступени заключается в извлечении набора событий из входящей очереди и вызове соответствующего обработчика событий. Обработчик событий обрабатывает каждый пакет событий, и отправляет избыточные события/заявки в очереди других ступеней. [2]

Событие обрабатываемое на ступени, как правило, представляет собой запрос клиента интернет-службы, например, запрос HTTP для веб-страницы. Однако события могут представлять другую информацию, имеющую отношение к работе службы, такие как установление сетевого соединения, то есть передачу управляющей сигнальной информации. Программная модель SEDA использует пакеты событий, так как это позволяет повысить производительность.

Обработчик событий – функция, которая принимает пакет событий в качестве входных данных, обрабатывает эти события, и при перегрузке переводит в очередь других ступеней.

Потоки являются тем самым ключевым процессом реализации «параллелизма» в SEDA. Они выполняют функцию «подхвата» избыточных ступеней других ступеней либо вывода избыточных заявок из своей очереди. [2].

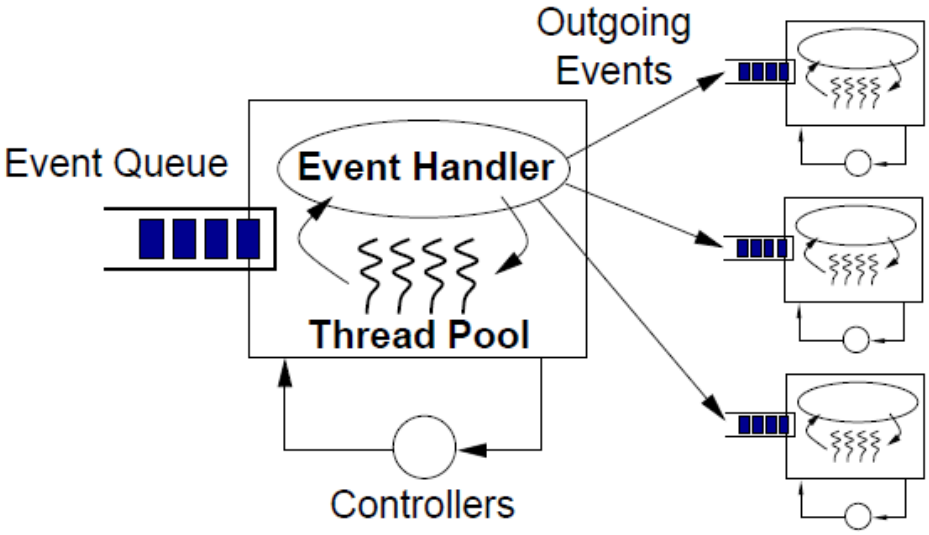

Рис. 2 Ступень SEDA [2].

Преимущество такого подхода заключается в том, что управление продолжением может выполняться приложением, когда это наиболее удобно (т.е. на этапе пересечение границ ступеней), а не в любое время во время операции обработчика событий.

Выводы: SEDA дает возможность создавать программные продукты, которые могут работать с огромным количеством входящих заявок, что является одним из основных признаков телекоммуникационной сети. Соответственно данная архитектура полностью удовлетворяет требованиям для реализации IMSaaS, что будет подтверждено и реализовано в последующей исследовательской работе.

Литература

1. Гольдштейн А. Б., Гольдштейн Б. С., SoftSwitch СПб.: БХВ – Санкт-Петербург, 2006. – 368 с.: ил.
2. An Architecture for Highly Concurrent,Well-Conditioned Internet Services: Matthew David Welsh. 211 p.
3. Cloudifying the 3GPP IP Multimedia Subsystem for 4G and Beyond: A Survey Mohammad Abu-Lebdeh, Jagruti Sahoo, Roch Glitho, Constant Wette Tchouati, Concordia University, Montreal, QC, Canada. 8 p.