УДК 681.007.05

Р.В. Гребинник, А.В. Липанов

Харьковский национальный университет радиоэлектроники, Харьков

ИССЛЕДОВАНИЕ МЕТОДОВ ПОСТРОЕНИЯ ОПТИМАЛЬНОЙ АРХИТЕКТУРЫ ОБЛАЧНЫХ СИСТЕМ

В статье проводится анализ подходов к проектированию оптимальных сервис-ориентированных архитектур. Исследование и создание математического аппарата для оптимизации архитектуры и рабочего процесса сложной сервис-ориентированной системы является очень важной задачей, поскольку ее решение позволит разрабатывать оптимальные системы как с точки зрения программных, так и аппаратных средств, а это в свою очередь позволит повысить эффективность обслуживания пользователей. Оптимизация рабочего процесса независимых сервисов выполняется за счет математического аппарата статистики и теории массового обслуживания.

Ключевые слова: *объект, класс, SOA, метод, архитектура, модель, статистика, система массового обслуживания.*

Введение

Сервис-ориентированная архитектура (SOA, service-oriented architecture) – модульный подход к разработке программного обеспечения, основанный на использовании сервисов (служб) со стандартизированными интерфейсами.

В основе SOA лежат принципы многократного использования функциональных элементов информационных технологий, ликвидации дублирования функциональности в ПО, унификации типовых операционных процессов, обеспечения перевода операционной модели компании на централизованные процессы и функциональную организацию на основе промышленной платформы интеграции.

Компоненты программы могут быть распределены по разным узлам сети, и предлагаются как независимые, слабо связанные, заменяемые сервисы-приложения. Программные комплексы, разработанные в соответствии с SOA, часто реализуются как набор веб-сервисов, интегрированных при помощи известных стандартных протоколов (SOAP и т. п.) [1].

Интерфейс компонентов SOA-программы предоставляет инкапсуляцию деталей реализации конкретного компонента (операционной системы, платформы, языка программирования, и т. п.) от остальных компонентов. Таким образом, SOA предоставляет гибкий и элегантный способ комбинирования и многократного использования компонентов для построения сложных распределённых программных комплексов.

SOA хорошо зарекомендовала себя для построения крупных корпоративных программных приложений. Целый ряд разработчиков и интеграторов предлагают инструменты и решения на основе SOA (например, платформы Intel SOA Expressway, JBoss SOA Platform, IBM WebSphere, Software AG webMethods, Oracle/BEA Aqualogic, Microsoft Windows Communication Foundation, SAP NetWeaver, TIBCO).

Основной раздел

1. Сервис-ориентированное программирование

Архитектура, как таковая, не привязана к какой-то определенной технологии. Организация системы независима от используемой вычислительной платформы (платформ) и независима от применяемых языков программирования. Проектируются сервисы, что не зависят от конкретных приложений, с одинаковыми интерфейсами доступа к ним. Организация сервисов стоится, как система слабосвязанных компонентов [1].

Главное, что отличает SOA – это использование независимых сервисов с четко определенными интерфейсами, которые для выполнения своих задач могут быть вызваны некоторым стандартным способом, при условии, что сервисы заранее ничего не знают о дополнении, который их вызывает, а приложение не знает, каким образом сервисы выполняют свою задачу.

SOA также может рассматриваться как стиль архитектуры информационных систем, который позволяет создавать приложения, построенные путем комбинации слабосвязанных и взаимодействующих сервисов. Эти сервисы взаимодействуют на основе какого-либо строго определенного независимого интерфейса (например, WSDL). Определение интерфейса скрывает языково-зависимую реализацию сервиса. Появление сервисно-ориентированного подхода сделало очередную реформу в теории разработки программного обеспечения, оставив в прошлом концепцию объектно-ориентированного программирования. Ряд архитектурных особенностей SOA позволяет уменьшить степень связанности различных элементов системы. Для взаимодействия компонентов используется сравнительно небольшой

набор простых интерфейсов, которые имеют только общую семантику и доступны всем провайдерам и потребителям. Через эти интерфейсы передаются сообщения, ограничены некоторым словарем. А поскольку данные только общая структура корпоративной системы и словарь, то вся семантика и бизнес-логика, специфичная для приложений, описывается непосредственно в этих сообщениях.

2. Практические аспекты применения SOA

Практические аспекты сервисно-ориентированной технологии позволяют решить проблемы масштабируемости, интегрировать сети передачи данных и голоса, упростить процедуры проектирования и управления сетями, а также создать другие распределенные приложения, прозрачно взаимодействующие с ресурсами систем с помощью прикладных программных интерфейсов и открытых стандартов.

Программные комплексы компаний Microsoft, SAP, ORACLE и других производителей являются наборами модулей для отдельных задач. У них также существуют проблемы интеграции, как внутри этих условно "монолитных" систем, так и с внешними системами других производителей.

Все представленные решения не "живут" в динамике изменения требований к объединенным информационным системам, не поддерживающих жизненный цикл функциональных систем, требуют колоссальных затрат ресурсов, не учитывают фактор единого времени в поддержании целостности и сопоставимости этих различных интегрированных систем, не эффективны при большом количестве систем [2].

Появление концепции SOA ознаменовало собой новый виток интеграционных технологий и связанных с ними надежд. Вобрав в себя технологические достижения компонентного программирования и Web-сервисов, SOA предлагает возможность гибкой работы с элементами бизнес-процессов и ИТ-инфраструктурой, лежащей в их основе, как с компонентами, которые можно использовать многократно и гибко комбинировать при изменении требований бизнеса. Благодаря этому SOA способна фундаментальным образом изменить подходы, применяемые в разработке и внедрении программных систем для бизнеса, обеспечивая преимущество в конкурентной борьбе.

Для организации взаимодействия сервисов нужна среда, которая обеспечит динамическую маршрутизацию запросов от прикладного компонента – потребителя сервиса и получения результатов от приложения – провайдера сервиса. Для этого может потребоваться поддержка синхронных и асинхронных коммуникаций низкого уровня между приложениями, трансформация и высокоскоростной распределение данных, трансляция протоколов, кэширование функций Web-сервисов, виртуализация ввода / вывода и т. д.

Конечная цель SOA – обеспечить представление бизнес-процессов, как взаимодействующих сервисов. Средства управления бизнес-процессами обеспечивают интеграцию в нужной последовательности сервисов, которые могут быть как локально-реализованными в ИТ-инфраструктуре компании, так и удаленными, если процесс на определенных этапах обращается к ресурсам партнерских компаний [3].

Проекты, созданные на базе SOA, – сложные методично, масштабные, затрагивающие основные бизнес-процессы. Кроме этого, задачи планирования оптимальной SOA архитектуры не решена в общем виде. Уровень взаимодействия сервисов и компонентов системы сугубо индивидуален и зависит от требований и конкретной реализации системы.

Задача оптимизации взаимодействия компонентов сложной сервис-ориентированной системы не может быть решена в общем виде, потому стоит задача планирования и распределения ресурсов масштабируемой системы. Фактически, вся сложность реализации архитектуры ложится на плечи разработчиков и невозможно дать полноценный анализ или подход к оптимизации. Предлагается создание набора правил и методов для анализа и распределения ресурсов вычислительной среды SOA.

3. Анализ оптимальной SOA архитектуры

В сервисно-ориентированной архитектуре все достаточно просто организовано: есть поставщики сервисов (приложения), создан реестр, где публикуется информация о поставщиках и куда заходит потребитель, для того, чтобы выбрать необходимый сервис. Потребитель получает нужную ссылку и связывается с поставщиком, чтобы передать запрос на обработку данных и получить результаты. Веб-архитектура предполагает такое взаимодействие через Интернет: например, сервисы через реестр сервисов публикуются в Интернет (курсы валют, погода, котировки акций). По мере реализации проекта построения интеграционного решения на базе SOA происходит публикация сервисов в реестре, и когда приходит время "связывать" приложения в рамках какого-то бизнес-процесса, это реализуется достаточно просто, потому что все осуществляется централизованно, через реестр сервисов. Таким образом, SOA дает возможность убрать лишние продукты и интеграционные решения, наглядно представить взаимодействия систем, оперативно конфигурировать решение при изменении бизнес-процессов и др. [4]

Задачи планирования оптимальной SOA архитектуры не решены в общем виде. Уровень взаимодействия сервисов и компонентов системы сугубо индивидуален и зависит от требований и конкретной реализации системы. Задача оптимизации взаимодей-

ствия компонентов сложной сервис-ориентированной системы не может быть решена в общем виде, предлагается создать уровень абстракций для анализа распределения нагрузки и ресурсов сервера или облачной системы. Основная идея оптимизации работы распределенной системы представлена на рис. 1.

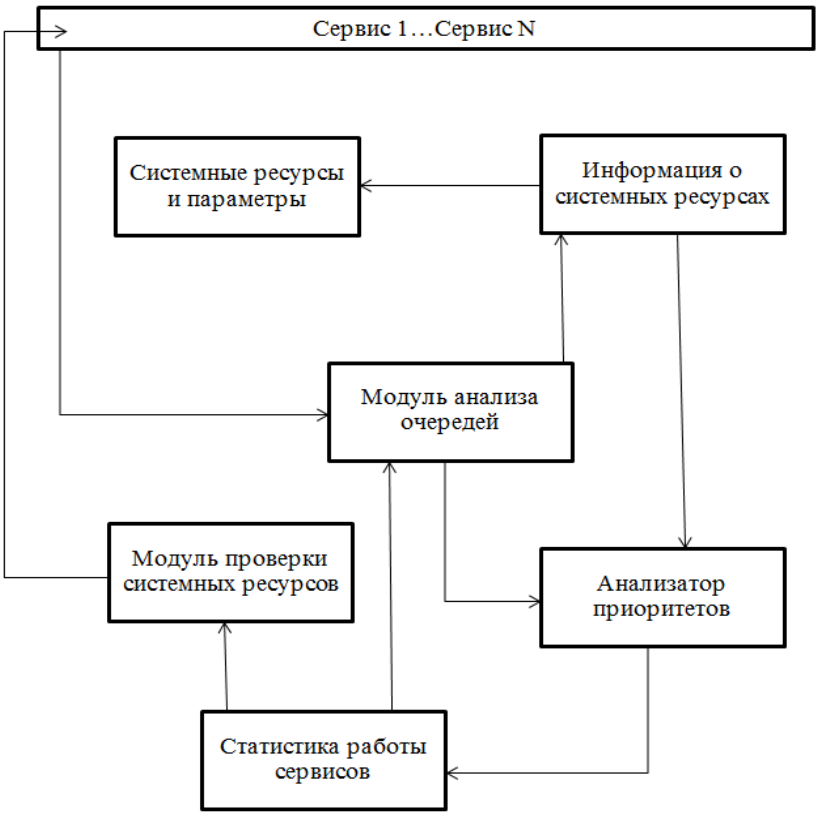

Рис. 1. Подход к оптимизации SOA архитектуры

Как видно из рис. 1, основным подходом к планированию распределения ресурсов будет анализ на основании теории массового обслуживания. На ее основе будет анализироваться очередь обработки каждого сервиса. Входные данные подвергаются анализу с использованием аппарата математической статистики и используя полученные результаты можно получить статистические данные по работе каждого сервиса, в которые будут входить:

– частота поступления заявки,
– время обслуживания заявки,
– объем входных данных,
– затраченные ресурсы вычислительного центра.

На основе этих данных можно проводить анализ и затем вырабатывать рекомендации по изменению архитектуры системы, а также можно реализовать специальные средства моделирования, которые позволят проводить анализ архитектуры еще до того как она реализована.

4. Сервис-ориентированная система —система массового обслуживания

Сервис-ориентированная система представляет собой систему массового обслуживания (СМО) и включает четыре основных элемента: поток сообщений, поступающих в систему обслуживания, характеристики качества и дисциплину обслуживания. Понятие потока сообщений включает информацию о модели потока вызовов (требований на соединения), закон распределения, длительность обслуживания (передачи) сообщений, множество адресов источников и приемников сообщений, а также типе занимаемого для передачи сообщений канала и способе передачи.

Математический аппарат теории массового обслуживания информации базируется на теории вероятностей, комбинаторике и математической статистике. Методы последней применяются в основном для обработки данных, получаемых при измерении параметров потоков сообщений и показателей качества обслуживания в реальных системах.

Отметим, что при интенсивности поступающей нагрузки ρ, равной или больше числа узлов обслуживания системы N, с вероятностью равной 1 постоянно будут заняты все узлы обслуживания и длина очереди будет бесконечной – явление «взрыва». Поэтому, чтобы система могла функционировать нормально и очередь не росла безгранично, необходимо выполнить условие:

$$\rho < N . \qquad (1)$$

Вероятность того, что система в установившемся режиме находится в состоянии k (P_k) определяем по формуле (второе распределение Эрланга):

$$P_k = \begin{cases} \frac{\rho^k}{k!} P_0, \text{ при } k = \overline{0, N}, \\ \frac{\rho^k}{N^{k-N} N!} P_0, \text{ при } k = \overline{N, \infty} \end{cases}, \qquad (2)$$

где $P_0 = \frac{1}{\sum_{k=0}^{N} \frac{\rho^k}{k!} + \frac{\rho^{N+1}}{N!(N-\rho)}}$.

Статистические данные, получаемые во время работы SOA сервера используются для вычисления оптимальной длины очереди для каждого сервиса и общей нагрузки на систему.

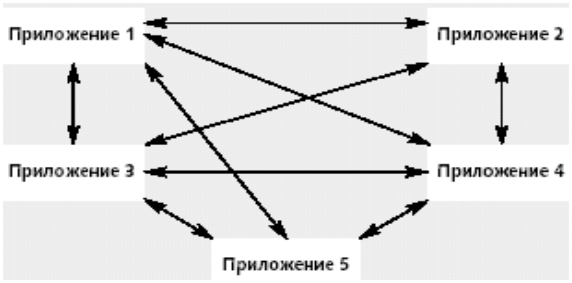

Рис. 2. Взаимодействие связных приложений

Все функции приложений определены как сервисы. В качестве сервиса может выступать как целое приложение, так и отдельные его функциональные модули. Сервисами могут быть прикладные функции, реализующие определенную бизнес-логику, бизнес-транзакции, состоящие из нескольких функций низкого уровня, и системные функции, отражающие специфику различных операционных платформ [5].

Все сервисы независимы друг от друга. Они выполняют определенные действия по запросам, полученным от других сервисов, и возвращают результаты. Все детали этого полностью скрыты: в концепции SOA сервисы – это "черные ящики".

В интерфейсе сервиса определены параметры и описан результат. Другими словами, интерфейс определяет суть сервиса, а не технологию его реализации. На архитектурном уровне для обращения к сервису не имеет значения, является он локальным (реализованный в этой системе) или удаленным (внешним по отношению к ней), какой протокол используется для передачи вызова, какие компоненты инфраструктуры при этом применены. SOA предполагает наличие единой схемы обращения к сервису независимо от того, находится ли они в том же применении, в другом адресном пространстве многопроцессорной системы, на другой аппаратной платформе в корпоративной intranet-сети или приложения в системе партнера.

4 Система моделирования нагрузки и реализация метода оптимизации архитектуры

В начальном состоянии системы, все сервисы имеют заданные по умолчанию приоритеты выполнения.

Разработчики могут предполагать возможную нагрузку на сервер и примерное число запросов на единицу времени для каждого сервиса. Но реальные результаты могут быть другими. Неверный расчет этих параметров может привести к снижению производительности сервера.

Накопление статистики по работе системы и использование ограничения очередей обеспечит гибкость в распределении ресурсов. Для этих целей предлагается использовать математическую статистику и теорию массового обслуживания.

Для анализа очереди заявок следует рассматривать четыре основных типа ресурсов:

- CPU,
- место на HDD,
- RAM,
- время выполнения.

В процессе моделирования в программном продукте SOA Modeling на вход подаются распределенные случайным образом на отрезке параметры нагрузки по каждому типу ресурсов. Формирование статистической выборки, расчет выборочного среднего, математического ожидания и дисперсии входящих сообщений, производится на основании методов математической статистики. Изменения в распределении ресурсов системы отображаются в реальном времени и гибко адаптируются на основании накопленной статистики, что представлено на рис. 3, 4.

Для анализа стрессовых ситуаций, в случайные моменты времени на вход подаются непредвиденные данные, близкие к предельным значениям. Обработка таких стрессовых ситуаций довольно предсказуема и поддается контролю.

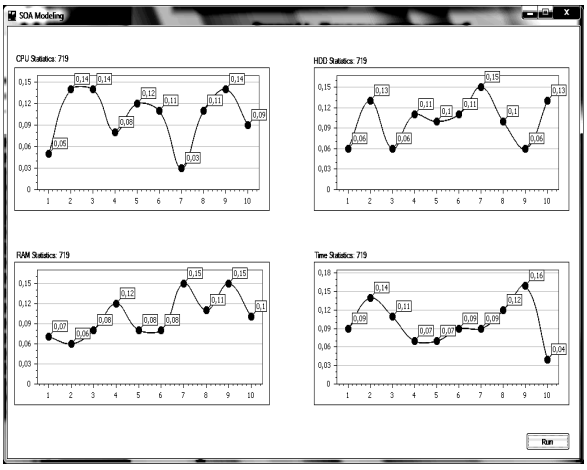

Рис. 3. Моделирование поступления заявок на сервер SOA

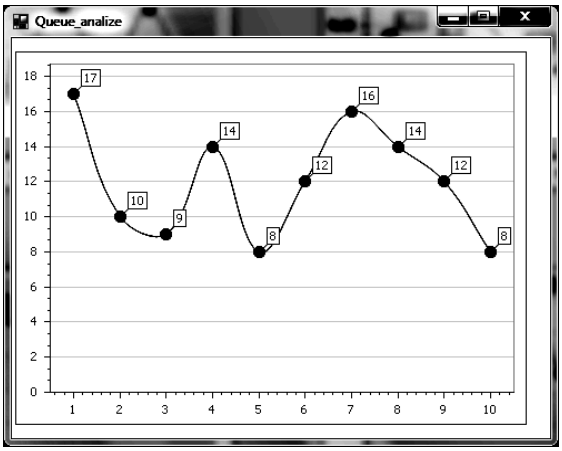

Рис. 4. Моделирование размера очереди

Когда статистическая выборка превышает 200 – 300 элементов, непредсказуемые нагрузки на сервер в случайные моменты времени не влияют на работоспособность системы в целом.

Полученные значения используются в математическом аппарате теории массового обслуживания для оценки текущего состояния системы, времени обслуживания заявок и анализа оптимального размера очереди для каждого независимого сервиса.

Этот подход позволяет гибко распределять ресурсы сервера или облачной системы для обеспечения наиболее оптимальной работы всей системы в целом, исключение простоев или необоснованного роста очереди запросов.

Выводы

Сервис-ориентированная архитектура, основанная на использовании сервисов (служб) со стандартными интерфейсами является широко распространенным подходом, используемым при реализации распределенных и облачных систем.

Одна из важных задач, стоящих перед разработчиками облачных систем – это построение оптимальной архитектуры сервис-ориентированной системы. В ходе исследований решалась задача построения математической и компьютерной модели для оптимального планирования и распределения ресурсов масштабируемой облачной системы. В основе математической модели лежит аппарат математической статистики и теории массового обслуживания. Статистические данные, собираемые во время работы системы, по всем независимым сервисам, используются для определения оптимального количества выделяемых ресурсов (CPU, RAM, HDD, Время выполнения). Также на основании полученной статистики формируются ограничения количества обрабатываемых заявок и длины очереди. Полученные результаты используются для оптимизации ресурсной составляющей SOA-системы, оптимизации работы и взаимодействия сервисов. Моделирование при помощи разработанной компьютерной модели также дает возможность проводить анализ параметров не только существующей, но и разрабатываемой системы, которая строится на основе сервис-ориентированной архитектуры. Таким образом решается задача оптимизации архитектуры как существующих систем так и тех, которые разрабатываются.

Список литературы

1. Stephen Bennett, Thomas Erl, Clive Gee SOA Governance : Governing Shared Services On - Premise & in the Cloud. – Hardcover 2012. – 675 p.

2. David Chou, John deVadoss, Thomas Erl SOA with .NET & Windows Azure : Realizing Service - Orientation with the Microsoft Platform. – Hardcover 2012. – 893 p.

3. Thomas Erl SOA Design Patterns. – NewYork.: Hardcover, 2011. – 865 p.

4. Thomas Erl, Anish Karmarkar, Priscilla Walmsley. Web Service Contract Design & Versioning for SOA. – Hardcover 2011. – 826 p.

5. Thomas Erl SOA Principles of Service Design. – NewYork.: Hardcover, 2010. – 573 p.

Поступила в редколлегию 11.05.2012

Рецензент: д-р техн. наук, проф. Е.П. Путятин, Харьковский национальный университет радиоэлектроники, Харьков.

ДОСЛІДЖЕННЯ МЕТОДІВ ПОБУДОВИ ОПТИМАЛЬНОЇ АРХІТЕКТУРИ ХМАРНИХ СИСТЕМ

Р.В. Гребінник, А.В. Ліпанов

У статті проводиться аналіз підходів до проектування оптимальної сервіс-орієнтованих архітектури. Дослідження і створення математичного апарату для оптимізації архітектури і робочого процесу складної сервіс-орієнтованої системи є дуже важливим завданням, оскільки її рішення дозволить розробляти оптимальні системи як з погляду програмних, так і апаратних засобів, а це у свою чергу дозволить підвищити ефективність обслуговування користувачів. Оптимізація робочого процесу незалежних сервісів виконується за рахунок математичного апарату статистики і теорії масового обслуговування.

Ключові слова: *об'єкт, клас, SOA, метод, архітектура, модель, статистика, система масового обслуговування.*

ANALYSIS OF METHODS FOR OBJECT RECOGNITION IN SYSTEMS FOR VISUAL INFORMATION ANALISYS

R.V. Grebinnyk, O.V. Lipanov

This paper contains the analysis of the optimal design of service-oriented architectures. Research and create their own mathematical tools for optimizing the workflow of complex service-oriented system. Optimizing workflow of independent services performed by the mathematical apparatus of statistics and queuing theory.

Keywords: *domain objects, object, class, SOA, architecture, methods, models, statistics, queuing system.*