ПОДХОДЫ К КОМПОНОВКЕ ВЕБ-СЕРВИСОВ НА ОБЛАКЕ

Г. В. Бондарчук

ВВЕДЕНИЕ

В связи с ростом объема данных, обрабатываемых предприятиями, появляется потребность в новых технологиях, удовлетворяющих современным требованиям. Требуются новые архитектурные подходы к созданию информационных систем. Одним из современных подходов является использование сервис-ориентированной архитектуры (СОА) [1]. Новые парадигмы, такие как сервис-ориентированные архитектуры и облачные вычисления, привели к превращению информационных технологий в стратегический фактор [2]. СОА решает такие проблемы, как зависимость от платформ разработки, компонентная зависимость (сильное связывание). Основными преимуществами подхода являются открытость за счет использования стандартов, быстрая интеграция приложений, ориентированность на межсетевое взаимодействие.

ПОДХОДЫ К КОМПОНОВКЕ ВЕБ-СЕРВИСОВ

Согласно [3], веб-сервис (веб-служба) – это услуга, предоставляемая посредством электронного устройства идентифицируемая веб-адресом и имеющая стандартизированные интерфейсы. Веб-сервис является единицей модульности в СОА. Одним из наиболее важных преимуществ веб-сервиса является его архитектура, делающая его независимым от платформы разработки и размещения. Также веб-сервисы основаны на базе открытых стандартов и протоколов. На высоком уровне абстракции веб-сервис можно рассматривать как сущность, имеющую входные и (или) выходные данные. Для обеспечения взаимодействия веб-сервисов используются следующие стандарты: XML, SOAP, REST, WSDL, UDDI, BPEL.

В работе [1] исследуется выбор рациональной композиции сервисов. Вводится новый критерий сравнения при выборе оптимальной композиции – чувствительность сервиса. Под чувствительностью здесь понимается критерий, на основе которого может быть оценена возможность обеспечения определенного уровня производительности веб-сервиса при возрастающей нагрузке.

Добавление нового критерия выбора оптимальной композиции к нефункциональным требованиям существенно усложняет и так не простой алгоритм.

Рассмотрим более подробно работу [4]. Основной идеей работы является создание компоновщика PlanICS. С целью упрощения процесс компоновки разбивается на три фазы: абстрактное планирование, поиск предложений и конкретное планирование. PlanICS использует унифицированное семантическое описание сервисов как часть предметной области. На первой стадии компоновки – абстрактном планировании – составляется абстрактная композиция на основе семантического описания сервисов. На данном этапе достаточно знать имеет ли атрибут объекта значение, абстрагируясь от конкретного значения данного атрибута.

На второй стадии компоновщик использует абстрактный план, построенный на первой стадии и репозиторий реальных сервисов, заменяя абстрактные сервисы, полученные из семантического описания с сервисами из репозитория. Также на данной стадии строится множество пользовательских ограничений к нефункциональным требованиям сервиса.

На третьей фазе – конкретном планировании – происходит поиск множества сервисов удовлетворяющих всем ограничениям, таким образом, чтобы максимизировать функцию качества. Согласно [5], задача конкретного планирования является NP-трудной задачей.

Следует заметить, что стадия конкретного планирования имеет наибольшие временные затраты в контексте процесса компоновки вебсервисов. В зависимости от сферы использования время поиска и составления композиции может играть ключевую роль процесса. Например, это может касаться систем реального времени, в которых временной параметр является ключевым.

Для последующего анализа разобьем нефункциональные характеристики на две основные группы: *производительность* и *качество*. Категория производительности включает критерии, характеризующие аспекты относящиеся к временным затратам по выполнению функций сервисами, такие как *пропускная способность* и *время выполнения функции*. Категория качества включает аспекты, характеризующие качество услуг предоставляемых сервисами, например, *доступность*, *время восстановления после сбоя*, *безотказность* и т.д.

Также можно рассматривать экономические аспекты, такие, как стоимость обработки запроса сервисом, лимит бесплатных запросов и т.д., однако в данной работе абстрагируемся от таких критериев.

Поскольку стадия конкретного планирования имеет наибольшую сложность в контексте процесса, рассмотрим некоторые шаги по ее упрощению. Основная сложность заключается в том, что используется как большое количество нефункциональных характеристик со стороны сервисов, так и нефункциональных требований со стороны клиентов (пользователей) сервисов. Выделение некоторой группы критериев с по-

следующей заменой на некоторый единый критерий позволило бы если не решить проблему выбора оптимальной композиции, то существенно ее упростить.

В контексте облачных вычислений открывается возможность сбора некоторой статистики об использовании веб-сервисов и хранения ее как отдельной нефункциональной характеристикой. Это вызвано, прежде всего, тем, что, во-первых, вне облака сервисы находятся в сфере ответственности частных поставщиков, что делает практически невозможным унифицированной и глобальной процедуры сбора статистики использования, во-вторых некоторые нефункциональные критерии, такие как чувствительность сервиса [1] или пропускная способность имеют совершенно другой смысл в контексте облачных вычислений: предоставляется возможность динамически добавлять или освобождать вычислительные ресурсы. Отметим, что подобная технология уже используется, например, в поисковой системе Google. При поиске используется алгоритм PageRank [6] для оценки полезности сайтов по количеству внешних ссылок на эти сайты.

Так как категория *производительности* является важной с точки зрения систем реального времени, заменим категорию критериев *качества* на единый критерий – *рейтинг веб-сервиса*, который может также включать количество использований сервиса.

Таким образом, модифицируя стадию конкретного планирования, а именно – задачу оптимизации по абстрактным планам получаем следующие результаты: уменьшается размерность задачи оптимизации, т.к. используется единый критерий для определения качества сервиса, следовательно, очевидным образом, сокращается время на поиск оптимальной композиции.

Зачастую одна и та же композиция строится и используется довольно часто, особенно при использовании социально-поисковых сервисов. Облачные вычисления, в силу гибкости использования ресурсов, позволяют хранить историю использования сервисов в композиции. Если композиция используется довольно часто, то есть смысл сохранить композицию как новый сервис, при этом используя стандарты СОА. То есть в WSDL-описании использовать не атомарный веб-сервис, а уже скомпонованный, путем автоматического составления WSDL-документа по результатам компоновки и подменой адреса сервиса на адрес скомпонованного (путем создания промежуточного уровня, содержащего скомпонованные сервисы). Данный подход решает проблемы выбора оптимальной композиции на третьей стадии построения композиции – конкретном планировании. Похожее предложение было рассмотрено в [7].

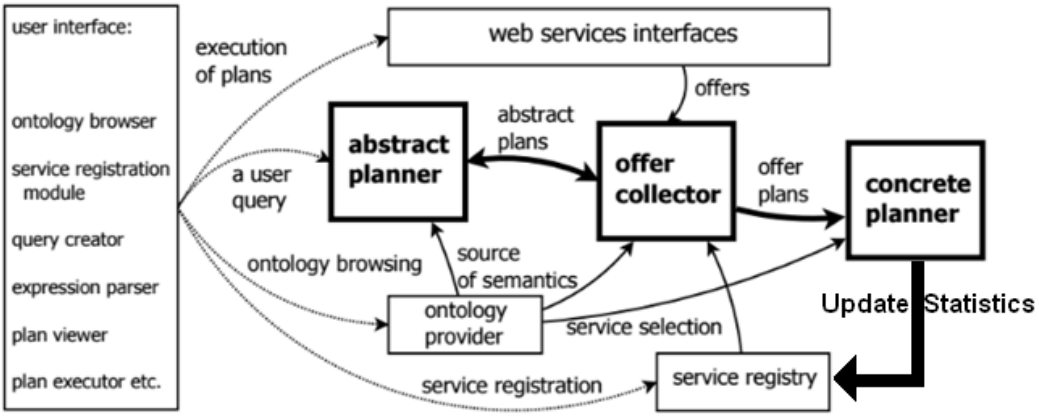

Рис. 1. Улучшенная архитектура PlanICS

Таким образом, в работе были получены следующие результаты:

- Проанализированы подходы компоновки веб-сервисов
- Предложен подход, основанный на статистике использования веб-сервисов
- Предложено хранить некоторые композиции как отдельные веб-сервисы.

Литература

1. *Душкин Д.*Н. Методы и алгоритмы выбора композиции веб-сервисов в системах с сервисно-ориентированной архитектурой: дис. … к-та техн. наук: 05.13.15 / Д.Н.Душкин – Москва, 2013. – 115 с.
2. *Карр Н. Дж.* Великий переход: что готовит революция облачных технологий / Н. Дж. Карр – Манн, Иванов и Фербер, 2013. – 272 с.
3. Web Service [Electronic resource] / Web Service – 2016. Mode of access: https://en.wikipedia.org/wiki/Web_service. – Date of access: 17.05.2016
4. *A. Niewiadomski, W. Penczek, and J. Skaruz*, “Genetic algorithm to the power of SMT: a hybrid approach to web service composition problem” in Service Computation 2014 : The Sixth International Conferences on Advanced Service Computing, 2014, pp. 44–48.
5. *A. Niewiadomski, W. Penczek, and J. Skaruz*, “SMT vs genetic algorithms: Concrete planning in PlanICS framework,” in CS&P, 2013, pp.309–321.
6. *Brin Sergey, Page Lawrence*. The anatomy of a large-scale hypertextual Web search engine // Comput. Netw. ISDN Syst. – 1998. – Vol. 30, no.1-7. – P. 107–117. – URL: http://dx.doi.org/10.1016/S0169-7552(98)00110-X.
7. *Литвин А. В., Войтешенко И. С.* Автоматизированная компоновка приложений служебно-ориентированной архитектуры // Международная научно-практическая конференция «Веб-программирование и Интернет-технологии WebConf09»: Сб. матер. Междунар. науч.-практич. конф. Минск, 8–10 июня 2009г. – Мн.: Институт математики НАН Беларуси, 2009. – с. 27–28.