Т. 8. №7. 2022
https://doi.org/10.33619/2414-2948/80

УДК 004.72 https://doi.org/10.33619/2414-2948/80/37

АНАЛИЗ ЭФФЕКТИВНОСТИ КЛИЕНТ-СЕРВЕРНОЙ АРХИТЕКТУРЫ

©Лиманова Н. И., *ORCID: 0000-0003-2924-5602, д-р техн. наук, Поволжский государственный университет телекоммуникаций и информатики, г. Самара, Россия*
©Селезнев И. А., *Поволжский государственный университет телекоммуникаций и информатики, г. Самара, Россия, il.samara@mail.ru*

ANALYSIS OF THE EFFECTIVENESS OF THE CLIENT-SERVER ARCHITECTURE

©Limanova N., *ORCID: 0000-0003-2924-5602, Dr. habil., Povolzskiy State University of Telecommunications and Informatics, Samara, Russia*
©Seleznev I., *Povolzskiy State University of Telecommunications and Informatics, Samara, Russia, il.samara@mail.ru*

Аннотация. В статье анализируется технология клиент-сервер, основные принципы построения данной архитектуры и ее сетевая инфраструктура. Описывается роль клиента и сервера, а также их взаимодействие. Выявляются ее сильные и слабые стороны, способы повышения эффективности и устранения неполадок. В работе показано, что использование такой архитектуры может быть целесообразным только при грамотном ее применении с учетом всех нюансов построении программного и аппаратного обеспечения.

Abstract. The article analyzes the client-server technology, the basic principles of building this architecture and its network infrastructure. Describes the role of the client and server, as well as their interaction. Its strengths and weaknesses, ways to improve efficiency and troubleshoot are identified. The paper shows that the use of such an architecture can be expedient only if it is properly applied, taking into account all the nuances of building software and hardware.

Ключевые слова: клиент-серверная архитектура, веб-приложение, система, технологии, клиент, сервер.

Keywords: client-server architecture, web application, system, technologies, client, server.

Введение

В настоящее время практически все веб-приложения построены на клиент-серверной архитектуре. Каждый день, делая покупки в интернет-магазинах или записываясь на прием в государственные учреждения, мы сталкиваемся с работой приложений, которые используют данную архитектуру. Однако, недочеты при построении этой системы могут приводить к постоянным сбоям, поэтому так важно понимать принципы работы клиент-серверной технологии и методы повышения эффективности работы ее программного обеспечения (https://clck.ru/qHZcR).

Материал и методы исследования

Основная цель исследования — проанализировать работу клиент-серверных приложений и выявить основные преимущества и недостатки данной архитектуры. Для этого надо выполнить следующие задачи:

Т. 8. №7. 2022
https://doi.org/10.33619/2414-2948/80

1. Изучить устройство клиент-серверной архитектуры;
2. Разобрать принцип ее работы;
3. Провести классификацию моделей по возможным параметрам;
4. Проанализировать клиент-серверную архитектуру двух веб-приложений и выявить способы повышения эффективности ее работы.

Результаты и обсуждение

Клиент-серверная архитектура — это вычислительная модель, в которой сервер управляет большей частью ресурсов и услуг, потребляемых клиентом. В этом типе архитектуры один или несколько компьютеров подключены по глобальной или локальной сети к главному серверу.

По своей сути клиент и сервер — это программное обеспечение, располагающееся, как правило, на разных вычислительных машинах и взаимодействующее между собой через сетевые протоколы. Однако, бывают случаи, когда клиент и сервер устанавливается на одну машину. В основном взаимодействие такой архитектуры при разделении клиента и сервера происходит через сетевой протокол http. Этот протокол состоит из двух основных команд: GET (получение информации с сервера) и POST (принимает данные для хранения). Такой простой, но в то же время эффективный набор правил идеально подходит для использования клиент-серверной системы.

При изучении клиент-серверной архитектуры (Рисунок) можно выделить три ее основных составляющих: клиент, сервер и сервер баз данных. Это и есть стандартная инфраструктура данной системы, иногда встречается устройство без сервера баз данных и двухуровневая архитектура. В таком случае сервер будет обрабатывать запрос от клиента и после обработки сразу отправлять ему ответ. В обратной ситуации, когда сервер баз данных присутствует (в трехуровневой архитектуре), сервер также будет обрабатывать запрос от клиента, но, помимо этого, он будет отправлять запрос к серверу базы данных, после чего сервер баз данных отправляет ответ обратно серверу и уже потом сервер приложения отправляет ответ клиенту.

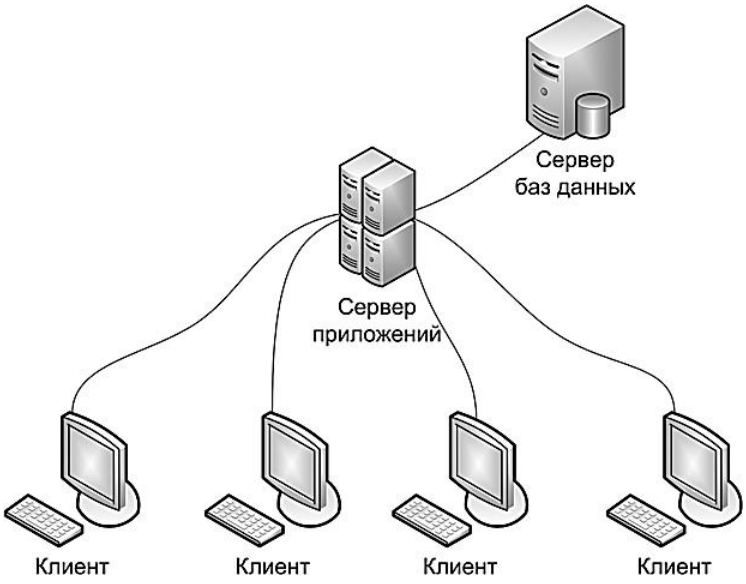

Рисунок. Архитектура клиент-сервер

Т. 8. №7. 2022
https://doi.org/10.33619/2414-2948/80

Важно понимать различие операций, которые проводятся на сервере приложения и сервере баз данных. Сервер баз данных будет возвращать наборы информации, а сервер приложения будет производить все необходимые вычисления с полученными данными.

Так как сервер может выполнять запросы от нескольких программ клиентов, то программу сервера размещают на специально выделенной вычислительной машине, следовательно, производительность данной машины должна быть очень высокой.

Системные требования к клиенту должны быть намного меньше, поскольку на нем производится гораздо меньше вычислений. Самым распространенным примером клиента является любой интернет браузер. В адресную строку вводится сайт, это и есть запрос, посылаемый от клиента к серверу, после чего сервер обрабатывает запрос и отправляет ответ на клиент в виде запрашиваемого сайта.

Клиенты бывают двух видов: толстый и тонкий. Тонкий клиент представляет собой программу или компьютер, который переносит все или наибольшую часть задач по обработке информации на сервер. Как правило, его основная цель отправить только запрос на сервер, поэтому системные требования к нему минимальные. Упомянутый выше браузер является примером тонкого клиента.

Толстым клиентом является приложение, которое обеспечивает расширенную функциональность, вне зависимости от центрального сервера. Чаще всего сервер в этом случае является только хранилищем данных, а вся задача по обработке этих данных переносится на машину клиента. В качестве примера можно привести программу «1С: Предприятие», где все вычисления бухгалтерского учета ведутся на клиенте, а на сервер передается лишь информация, которую необходимо сохранить и впоследствии получить для повторной обработки. Еще одним примером толстого клиента могут быть онлайн игры, в которых все действия с игровым процессом выполняются на компьютере пользователя, а на сервер отправляются лишь данные об изменениях, произошедших на аккаунте в результате действий пользователя.

Преимуществ у архитектуры данной модели несколько. Во-первых, из-за того, что все вычисления происходят на стороне сервера, то программам клиента не нужно дублировать код. Во-вторых, большинство вычислений выполняются на сервере, поэтому требования к клиенту приложения минимальные. В-третьих, из-за того, что все данные хранятся на сервере, риск утечки этих самых данных гораздо ниже, потому что как правило, клиент защищен гораздо хуже, чем сервер.

Рассмотрим недостатки данной технологии. Так как хранение всех данных и вычислений производятся на сервере, то при отказе работы серверной части вся архитектура становится неработоспособной. И еще одним недостатком является высокая стоимость оборудования. Если оборудование серверной части будет дешевым и обладать низкой производительностью, то это вызовет перегрузку сети с возможной последующей остановкой работы сервера.

Еще один отрицательный фактор – это необходимость плановых работ. При проведении технического обслуживания для изменения программного или аппаратного обеспечения сервера требуется его отключение, на время которого все посылаемые клиентом запросы не будут обрабатываться. Как правило, такие работы не превышают 12 часов, но даже такой сравнительно небольшой промежуток времени может оказаться критическим для работы некоторых приложений.

Т. 8. №7. 2022
https://doi.org/10.33619/2414-2948/80

В основном, все проблемы решаются грамотно установленным оборудованием и логически верным написанным кодом. Поэтому правильный подбор аппаратного и программного обеспечения минимизирует риск проявления сбоев работы системы.

С целью выявления основных причин возникновения ошибок в работе клиент-серверной архитектуры лучше всего провести сравнительный анализ работы двух систем: с высокими показателями отказоустойчивости и с низким.

Для разбора примера с наличием ошибок в проектировании системы подходит портал государственных и муниципальных услуг Российской Федерации (*далее — Госуслуги*), а в качестве стабильно работающей системы — видео хостинг YouTube (https://www.vesti.ru/article/2667235; https://clck.ru/MYBvX).

Один из самых грубых недочетов в построении системы — это несоблюдение принципа централизации и выполнение на клиентской части операций, которые должны производиться на сервере. Так, в июле 2021 года сайт Госуслуг добавил счетчик обратного времени, ставящий пользователя в очередь. Но, как выяснилось позже, счетчик можно сбросить через код страницы, отредактировав время, тем самым пользователь моментально получал доступ к сайту (что не предусматривала система). То есть, пользователь получал доступ к ресурсам выше своих прав, что в корне не соответствует принципу работы веб-приложения. YouTube ни разу не был замечен в подобных упущениях при построении централизации веб-приложения. Клиент всегда ограничивался только тем спектром возможностей, которые предоставляет ему система (https://clck.ru/qHaiq).

Еще один важный аспект — это высокий уровень защиты. Чем более конфиденциальной и важной информацией о пользователе обладает приложение, тем чаще оно будет подвержено атакам. Нередки ситуации получения персональных данных пользователей Госуслуг путем мошеннических схем, а также случаи утечек исходного кода и данных о пользователях. YouTube в подобных инцидентах замечен не был, из-за своего подхода к информационной безопасности. Одна из основных причин — использование аккаунтов сервисов Google для аутентификации в системе, которые известны своей повышенной безопасностью за счет двухфакторной аутентификации (https://habr.com/ru/post/357848/).

Также, немаловажным аспектом является выбор аппаратного обеспечения сервера. Если серверное оборудование не будет удовлетворять необходимым требованиям, то веб-приложение будет работать с перебоями, так как запросы, полученные сервером, будут обрабатываться медленнее. Информации по поводу серверов, используемых госуслугами в открытом доступе, нет, но исходя из факта того, что сайт работает с частыми сбоями, можно сделать вывод, что аппаратное обеспечение не выдерживает полную нагрузку. В этом плане YouTube тоже не является идеальным, но общие показатели гораздо выше, чем у Госуслуг. Каждое загруженное на хостинг видео хранится в одном из 14 центров Google по обработке данных, расположенных по всему миру. Такая система позволяет поддерживать работу с нагрузкой до 1,9 млрд. пользователей ежемесячно, а также на YouTube приходится 37% всего мобильного трафика в мире (по статистике на 2019 год). В то время, как число пользователей Госуслуг в 2020 году достигло только 56 млн человек.

Заключение

Исходя из вышеперечисленных наблюдений, можно сделать вывод что, несоблюдение принципов построения клиент-серверной архитектуры и нарушение ее основных правил,

Т. 8. №7. 2022
https://doi.org/10.33619/2414-2948/80

может привести к получению некачественного программного продукта, в котором технология клиент-сервер не может быть реализован со всей эффективностью.

Работа поступила
в редакцию 12.06.2022 г.

Принята к публикации
18.06.2022 г.

Ссылка для цитирования:

Лиманова Н. И., Селезнев И. А. Анализ эффективности клиент-серверной архитектуры // Бюллетень науки и практики. 2022. Т. 8. №7. С. 392-396. https://doi.org/10.33619/2414-2948/80/37

Cite as (APA):

Limanova, N., & Seleznev, I. (2022). Analysis of the Effectiveness of the Client-Server Architecture. *Bulletin of Science and Practice, 8*(7), 392-396. (in Russian). https://doi.org/10.33619/2414-2948/80/37