Научная статья
Original article
УДК 004

СПОСОБЫ ИНТЕГРАЦИИ

WAYS OF INTEGRATION

Копылова Яна Антоновна, Студент, МИРЭА-Российский технологический университет (РТУ МИРЭА)
Матвеев Владимир Евгеньевич, Студент, 3 курс, направление «Прикладная информатика», МИРЭА-Российский технологический университет (РТУ МИРЭА), 119454, Россия, г. Москва, проспект Вернадского, 78, Институт информационных технологий, Россия, г. Москва

Kopylova Yana Antonovna, Student, MIREA-Russian Technological University (RTU MIREA)
Matveev Vladimir Evgenievich, Student, 3 course, direction "Applied Informatics", MIREA-Russian Technological University (RTU MIREA), 119454, Russia, Moscow, Vernadsky Avenue, 78, Institute of Information Technology, Russia, Moscow

Аннотация: В современном мире редко можно встретить систему, состоящую из одного приложения или модуля. В связи с ростом потребностей пользователей, возросла и сложность программных продуктов, что привело к их разделению на несколько компонентов. Следовательно, возникла необходимость в соединении различных частей разрабатываемых комплексов.

Такой процесс называется интеграцией и может быть выполнен несколькими способами. Существует всего 5 основных методов для интеграции компонентов и каждый из них обладает своими преимуществами и недостатками, как с точки зрения программной реализации, так и со стороны бизнеса. Способ интеграции должен выбираться исходя из архитектуры программного комплекса и финансовых ресурсов компании, что позволит достичь максимальной эффективности и надежности.

Abstract: In the modern world, it is rare to find a system consisting of a single application or module. Due to the growing needs of users, the complexity of software products has also increased, which led to their division into several components. Consequently, there was a need to connect the various parts of the complexes being developed. This process is called integration and can be performed in several ways. There are only 5 main methods for integrating components, and each of them has its own advantages and disadvantages, both from the point of view of software implementation and from the business side. The integration method should be chosen based on the architecture of the software package and the financial resources of the company, which will achieve maximum efficiency and reliability.

Ключевые слова: интеграция, приложения, методы, точка-точка, система, сервис, подход, узлы

Key words: integration, applications, methods, point to point, system, approach, knots

Интеграция, в широком смысле понимания данного термина, означает процесс объединения частей в целое. В контексте интеграции приложений определение принимает следующий вид: «Интеграция – это возможность обмена данными между системами с последующей их обработкой». Таким образом, главная задача интеграции – предоставить возможность отдельным приложениям обмениваться данными с целью увеличения функциональности всей системы. Такой подход позволяет избежать необходимости повторной разработки одних и тех же функций, что повышало бы время, требуемое на

разработку финального продукта.

1. Интеграция при помощи метода «Point-to-point»

В качестве первого рассматриваемого подхода к интеграции можно выделить способ под названием «Точка-точка» (Point-to-point) [1]. Такой подход предполагает взаимодействие приложений напрямую, то есть каждое интегрируемое приложение должно подразумевать способ для общения с каждым из остальных. Таким образом, при изменении формата данных или способа работы одного из приложений необходимо также отредактировать способ связи с этим приложением у всех остальных компонентов системы.

Из вышесказанного следует, что применение подхода «Точка-точка» допустимо исключительно в системах с небольшим количеством составляющих, так как с ростом числа приложений растет и сложность системы, а значит и сложность ее изменения. Несложно вычислить, что при наличии хотя бы восьми компонентов, количество уникальных связей, требующих постоянного поддержания, будет равняться двадцати восьми. Пример такой системы приведен на Рисунке 1.

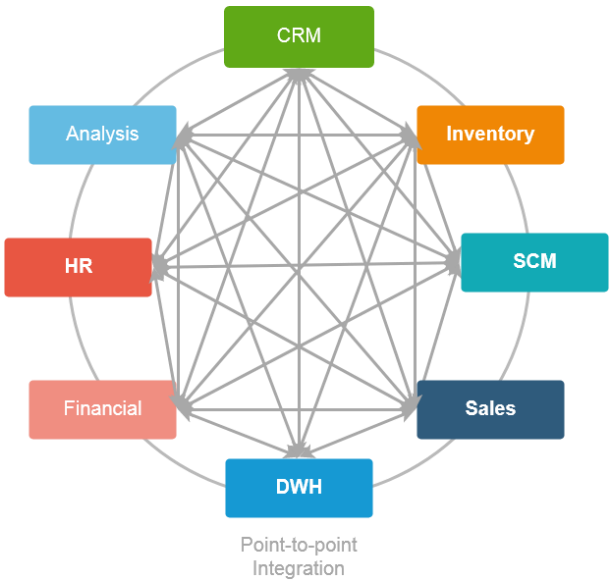

Рисунок 1 – Схема реализации подхода «Точка-точка» в системе с восьмью компонентами

Тем не менее, у данного подхода есть и положительные стороны. Во-первых, реализация данной модели обладает исключительной простотой при разработке, так как она требует лишь написания правил для взаимодействия приложений напрямую и не нуждается, например, в стандартизации формата данных для всей системы. Во-вторых, такой метод интеграции обладает высоким уровнем прозрачности, так как изменение одного из компонентов однозначно определяет необходимые изменения в остальных приложениях. Наконец, реализация метода «Точка-точка» не требует разработки дополнительного программного обеспечения, так как все элементы связаны напрямую друг с другом.

2. Интеграция при помощи метода «Hub-and-spokes»

Вторым популярным подходом к интеграции различных приложений является так называемый “Hub and spokes” [1]. Основная идея этого метода заключается в том, что существует отдельное приложение – hub (основной узел), через которое осуществляется связь различных приложений. Так, отдельные компоненты системы связаны не каждый с каждым, а только с узлом, что позволяет существенно снизить количество существующих связей. Схематичное изображение интеграционного решения “Hub-and-spokes” представлено на Рисунке 2.

Рисунок 2 – Схема реализации подхода «Hub-and-spokes» в системе с восьмью компонентами

Такой способ интеграции не требует разработки множества видов взаимодействия для каждой пары приложений. Всё, что требуется для создания общей системы – это создать уникальный процесс для передачи данных от каждого отдельного приложения к узлу. Ещё одним достоинством такого метода является простота в расширении системы благодаря тому, что при добавлении нового приложения его необходимо связать только с hub-приложением.

Тем не менее, данный метод обладает некоторыми недостатками. Один из самых весомых – дороговизна при создании системы, так как необходимо разработать отдельное приложение для связи остальных. Из предыдущего следует также и сложность разработки системы, включая дополнительные затраты во времени. Помимо вышеперечисленного, если приложение-hub выйдет из строя, то вся система перестанет корректно работать. Наконец, значительным недостатком данного метода является то, что пропускная способность при обмене сообщениями между приложениями ограничена максимальной скоростью работы центрального узла.

3. Интеграция при помощи метода «Enterprise Service Bus»

Далее следует рассмотреть подход на основе «Серверной шины предприятия» (ESB – Enterprise Service Bus). Такой способ имеет много общего с предыдущим. Основная идея заключается в том, что помимо отдельных приложений, в системе присутствует так называемая шина – специальное решение, позволяющее передавать информацию между отдельными компонентами. Эта шина используется в следующем порядке: одно приложение отправляет данные или запрос в шину, а приложения-слушатели проверяют шину и отлавливают эти сообщения, после чего обрабатывают полученную информацию и могут также отправить свое сообщение в шину. Схематичное изображение данного метода представлено на Рисунке 3.

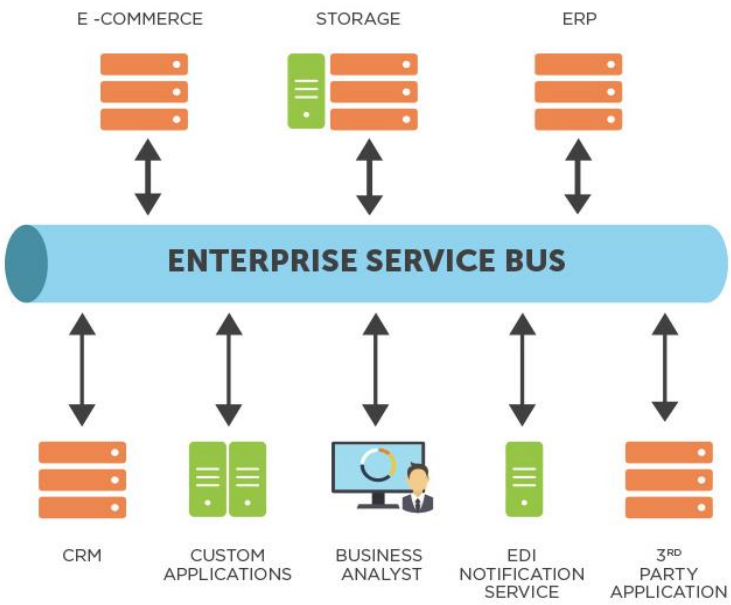

Рисунок 3 – Схема реализации интеграции приложений методом «Enterprise Service Bus»

Главное отличие данного метода от “Hub-and-spokes” состоит в том, что центральный узел в прошлом методе самостоятельно принимает решения о перенаправлении всех сообщений от одного приложения к другому, а шина выступает в роли потока сообщений, из которого приложения сами забирают нужные им сообщения. Это позволяет достичь следующего порядка действий: одно приложение отправляет сообщение, шина преобразует его в какой-либо уникальный формат, другое приложение пытается получить информацию из этого сообщения, шина преобразует данные в нужный для этого приложения формат и отправляет.

Благодаря вышесказанному такой подход более подвержен расширяемости, так как обработка данных с точки зрения передачи сообщений полностью происходит на стороне приложения-шины и нет необходимости встраивать данную функцию в каждое отдельное приложение.

В качестве одного из основных минусов данного метода можно также выделить единственную «точку провала», так как при некорректной работе шины связь между приложениями нарушается. Вторым существенным недостатком, как и в случае с предыдущим методом, является тот факт, что

наличие единственного связывающего приложения является проблемой из-за того, что при большом потоке сообщений скорость работы системы будет ограничена скоростью работы шины, что может в результате привести к значительному снижению общей производительности.

Таким образом, данный метод выглядит более предпочтительным, чем способ “Hub-and-spokes”, так как обладает практически теми же недостатками, но имеет более высокую расширяемость и производительность.

4. Интеграция при помощи метода «Middleware»

Следующий метод для интеграции приложений имеет название «Middleware», что можно перевести как «Связующее программное обеспечение» или «Промежуточное программное обеспечение» [2]. Этот способ основан на двух предыдущих, но решает некоторые из их проблем. Основная идея заключается в том, что существует несколько дополнительных программных решений, каждое из которых может как решать отдельную задачу, так и повторять функциональность другого такого решения.

Целью таких дополнительных программ является обработка сообщений от одного приложения и последующее перенаправление либо в другое приложение, либо на еще одно промежуточное программное обеспечение. Каждое из этих связующих ПО может быть реализовано как на основе метода «Hub-and-spokes», так и на основе метода «Enterprise Service Bus». В качестве второстепенных задач, такое программное обеспечение может отфильтровывать различные сообщения, возникшие по ошибке или, например, пришедшие от потенциальных злоумышленников.

Положительные стороны данного метода выделяют его на фоне остальных. Так, система, интегрированная при помощи промежуточного программного обеспечения, имеет более высокую надежность благодаря тому, что она может обладать запасными элементами на случай, если какие-либо выйдут из строя, а даже если перестанут работать все однотипные связующие ПО, то система продолжит функционировать, лишившись лишь одной из

частей. Из вышесказанного также следует, что такой программный комплекс намного удобнее и проще поддерживать, так как при возникновении каких-либо ошибок, процесс обнаружения поломки окажется незатруднительным за счет того, что отключится какой-либо определенный модуль. В качестве еще одного достоинства можно также выделить то, что при использовании способа «Middleware» интеграции устраняется проблема с пропускной способностью системы. Различные промежуточные приложения могут выполнять одни и те же операции, но равномерно распределять нагрузку между друг другом.

Конечно же, данный метод также обладает своими недостатками. Одним из ключевых является сложность и дороговизна первичной настройки интегрированной системы. Описанный выше способ также предполагает разработку большого количества дополнительного программного обеспечения, что требует большого расхода ресурсов, а иногда и отдельной команды, занимающейся непосредственно процессом интеграции. Помимо прочего, данный метод также предлагает довольно большое количество связей, хоть их настройка и становится проще чем в методе «Точка-точка». Данное свойство такого метода может накладывать свои ограничения на производительность системы, тем не менее достоинства интеграционного способа «Middleware» перевешивают этот недостаток.

5. Интеграция при помощи метода «Microservices»

Наконец, последний из рассматриваемых способов интеграции называется “Microservices Integration” («Интеграция при помощи микросервисов»). Данный подход основывается на понятии микросервиса, которое означает одно отдельное приложение (или независимый модуль), выполняющее конкретную небольшую задачу. Помимо самих сервисов в системе может также присутствовать от одного до нескольких дополнительных приложений-брокеров для перенаправления сообщений между компонентами программного комплекса.

Одно из основных отличий данного метода от всех описанных ранее заключается в том, что система программных приложений может изначально разрабатываться, используя микросервисный подход, даже в рамках одного приложения. Итоговый программный комплекс в таком случае будет состоять из базы данных, клиентского приложения и необходимого количества микросервисных решений.

Тем не менее, данный подход можно применить для интеграции существующих приложений. Так, каждое приложение будет связано с одним или несколькими микросервисами, которые будут проводить фильтрацию и обработку данных, после чего перенаправлять их либо в другие микросервисы, либо в приложения-получатели.

Данный подход получает все больше распространения благодаря тому, что он позволяет создавать децентрализованные системы, которые достаточно просто поддерживать. Поиск и отладка ошибок выполняются сравнительно быстро за счет того, что все функции отделены друг от друга. Помимо прочего, использование данного метода не несет больших затрат, так как связующие приложения небольшие и выполняют конкретные задачи, а при условии использования данного метода, начиная с момента разработки всей системы, дополнительные затраты и вовсе не требуются. Также данные системы очень легко расширять, так как добавление нового приложения или функции, требует только написания соответствующего сервиса. Наконец, одной из самых выдающихся отличительных черт данного подхода является высокая «сменяемость». Так, если необходимо полностью изменить какой-либо модуль или связь, достаточно просто изменить работу одного сервиса, после чего внедрить его вместо старого микросервиса, а остальная часть системы продолжит работать неизменно.

Данный способ также обладает и негативными сторонами. Первым из недостатков можно считать относительную новизну данного метода, что влияет на эффективность работы команд, которые пока не привыкли к данному подходу, но, очевидно, данная проблема со временем исчезнет.

Вторая проблема заключается в том, что за счет разрозненности системы её безопасность может быть снижена, так как каждая связь (которых может быть довольно много) должна быть защищена. Конечно, если где-то защита системы будет нарушена, и злоумышленник получит доступ к части программного комплекса, решение проблемы и восстановление системы не займет много времени, но сам факт возникновения проникновения недопустим, особенно в крупном бизнесе.

Таким образом, при должном регулировании и управлении данный подход можно считать одним из наилучших в большинстве случаев, так как его достоинства сильно перевешивают недостатки.

Подводя итог, интеграция помогает ускорить и упростить разработку приложений, благодаря расширению функционала при помощи использования других, написанных ранее, решений. Существуют различные способы для реализации данного подхода, которые обладают своими достоинствами и недостатками. В каждом отдельном случае интеграции приложений стоит выбирать метод, исходя из целей, задач и потребностей итоговой системы. Помимо прочего, развитие в области объединения программных разработок не остановилось и до сих пор появляются новые подходы и решения.

Список использованных источников

1. Enterprise Application Integration (EAI). – Текст : электронный // Medium : [сайт]. – 2019. – URL: https://medium.com/accesa/enterprise-application-integration-eai-df3e0d660482 (дата обращения: 21.09.2022).
2. EAI (enterprise application integration). – Текст : электронный // Techtarget : [сайт]. – 2021. – URL:https://www.techtarget.com/searchapparchitecture/definition/EAI-enterprise-application-integration (дата обращения: 21.09.2022).

3. Беседина, К. В. Современные подходы к интеграции корпоративных приложений / К. В. Беседина. – Текст : электронный // Cyberleninka : [сайт]. – 2007. – URL: https://cyberleninka.ru/article/n/sovremennye-podhody-k-integratsii-korporativnyh-prilozheniy/viewer (дата обращения:16.10.2022).
4. Иваненчук, А. Ю. Методы интеграции приложений / А. Ю. Иваненчук, А. А. Малинин. – Текст : электронный // Cyberleninka : [сайт]. – 2007. – URL: https://cyberleninka.ru/article/n/metody-integratsii-prilozheniy/viewer (дата обращения:16.10.2022).
5. Интеграция программного обеспечения. Описание процесса от бизнес-консультанта. – Текст : электронный // Хабр : [сайт]. – 2014. – URL: https://habr.com/ru/company/trinion/blog/245615/ (дата обращения: 22.10.2022).

List of sources used

1. Enterprise Application Integration (EAI). – Text: electronic // Medium: [website]. – 2019. – URL: https://medium.com/accesa/enterprise-application-integration-eai-df3e0d660482 (date of access: 09/21/2022).
2. EAI (enterprise application integration). – Text: electronic // Techtarget: [website]. – 2021. – URL: https://www.techtarget.com/searchapparchitecture/definition/EAI-enterprise-application-integration (Accessed 09/21/2022).
3. Besedina, K. V. Modern approaches to the integration of corporate applications / K. V. Besedina. – Text: electronic // Cyberleninka: [website]. – 2007. – URL: https://cyberleninka.ru/article/n/sovremennye-podhody-k-integratsii-korporativnyh-prilozheniy/viewer (date of access: 10/16/2022).
4. Ivanenchuk, A. Yu. Application integration methods / A. Yu. Ivanenchuk, A. A. Malinin. – Text: electronic // Cyberleninka: [website]. – 2007. – URL: https://cyberleninka.ru/article/n/methody-integratsii-prilozheniy/viewer (date of access: 10/16/2022).

5. Software integration. Description of the process from a business consultant. - Text: electronic // Habr: [website]. - 2014. - URL: https://habr.com/ru/company/trinion/blog/245615/ (date of access: 10/22/2022).

Для цитирования: Копылова Я.А., Матвеев В.Е. СПОСОБЫ ИНТЕГРАЦИИ // Научный сетевой журнал «Столыпинский вестник» №9/2022