Шишкин А.Г. Метод бесшовного обновления программного обеспечения // Международный журнал информационных технологий и энергоэффективности. – 2023. – Т. 8 № 7(33) с. 38–43

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004

МЕТОД БЕСШОВНОГО ОБНОВЛЕНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Шишкин А.Г.
Группа компаний «ИксКом» (ООО М-Инвест), Москва, Россия, (125212, г. Москва, Кронштадтский бульвар, д. 3А), e-mail: andrey.shishkin.spb@gmail.com

В статье предложен метод перехода на новое программное обеспечение в условиях, когда невозможно прекратить работу текущего программного обеспечения ни на минуту, а сам переход невозможно сделать одномоментно, то есть, когда требуется идти по процессу поэтапного внедрения новых узлов, при этом обеспечить стыковку частей старого и нового программного обеспечения. Данная проблема часто присутствует в средних и крупных компаниях, в которых присутствует сложная техническая платформа и которая уже перестала удовлетворять современным требованиям бизнеса. Текущую платформу развивать и поддерживать становится слишком дорого и бесперспективно. Требуется заменить программное обеспечение, но без вреда бизнесу и каким-либо процессам внутри компании.

Ключевые слова: Программное обеспечение, миграция данных, техническая трансформация, технологическая трансформация, микросервисная архитектура, модульная архитектура, обновление программного обеспечения.

SEAMLESS SOFTWARE UPDATE METHOD

Shishkin A.G.
X-Com group of companies (LLC M-Invest), *Moscow, Russia (125212, Moscow, Kronstadt Boulevard, 3A), e-mail: andrey.shishkin.spb@gmail.com*

The article proposes a method for switching to new software in conditions where it is impossible to stop the current software for even a minute, and the transition itself cannot be done at once. Those. when it is necessary to go through the process of phased introduction of new nodes, while ensuring the docking of parts of the old and new software. This problem is often present in medium and large companies that have a complex technical platform and which has ceased to meet modern business requirements. It becomes too expensive and unpromising to develop and maintain the current platform. It is required to replace the software, but without harm to the business and any processes within the company.

Keywords: Software, data migration, technical transformation, technological transformation, microservice architecture, modular architecture, software update.

Введение

Примерно с 2015 года в России все чаще стало звучать слово “трансформация”, которую почти всегда относят к сфере информационных технологий. Речь идет о крупных и средних компаниях, которые задались целью обновить свою техническую и/или технологическую платформу для реализации своих стратегий развития [1]. Это применимо и для компаний коммерческого сегмента, и для компаний с государственным участием. Разница лишь

необходимости следования требованиям ГОСТа и в степени возможности отклонения от ГОСТ, или в полном отказе от ГОСТ.

Термин “трансформация” подразумевает не просто обновление технологической платформы, а в том числе и значительное обновление функционала, алгоритмов работы, технологий и т.п. [2] Это все значительно сложнее, чем простое обновление программного обеспечения на новую версию [3]. Процесс трансформации крупной компании можно сравнить с процессом полной переделки автомобиля из одной модели в другую находясь внутри машины, да еще и в процессе ее движения (на ходу). Аналогия шуточная, но показательная. Видно, насколько сложно может быть процесс трансформации.

Автор статьи разработал и применил методику, позволяющую провести подобную трансформацию “на ходу”, без остановки бизнеса, с минимальными потерями в процессе переобучения персонала, вводом новых функций. И все это в разумные сроки, приемлемые для бизнеса. Методика была разработана и применялась в технологической трансформации крупной компании из рынка электронной коммерции, где не допускалось останавливать работу ни сайтов, ни мобильных приложений, ни любого из отдела компании.

Описание проблемы

Любая современная компания стремится развиваться. Для этого она закупает соответствующее программное обеспечение, которое со временем устаревает. В современной практике чаще всего применяется микросервисная архитектура работы программного обеспечения [4]. Эта архитектура максимально качественно обеспечивает все текущие требования бизнеса по всем видам масштабирования, отказоустойчивости, взаимозаменяемости, стоимости поддержки, обновления и т.п. Однако это не значит, что через 10 лет не появиться новая архитектура, которая придет на смену микросервисной.

До появления микросервисной архитектуры, широкое применение было у модульной архитектуры [5]. А до этого работала моноархитектура, когда весь программный код писался как единое целое.

Как видно, технологии развиваются. А значит нужно развиваться и компаниям. Но не все компании успели вовремя обновить свои платформу. Кто-то просто не заметил этот тренд и отстал, у кого-то сложность перехода уже тогда была высокой, и компания откладывала этот шаг как можно дальше. Есть еще не одна сотня причин, почему компании не обновляли свои платформы.

В итоге, очень многие коммерческие компании оказались в ситуации, когда старая (текущая) платформа поглощает огромные ресурсы на ее поддержание и попытки ее развития. При этом эффективность вложенных средств в развитие кратно ниже современных показателей.

Попытка внести изменение упирается в следующие проблемы:

- Невозможно выполнить доработку ввиду конфликта между новыми требованиями и текущей архитектурой.
- Вводимые доработки каскадом выдают ошибки во всех смежных системах.
- Обновление происходит долго, с перезагрузкой серверов. Есть риски потерять данные.

- Крайне острая зависимость от текущего штата программистов, т.к. поддержание работы платформы требует одновременного знания всех ее составляющих.

В итоге, компания вынуждена принять решение о полной трансформации. И чем позже это решение будет принято, тем более болезненный путь необходимо будет пройти.

Сама по себе трансформация, в общем понимании, не несет в себе никаких особых сложностей. Разрабатываются или покупается соответствующее программное обеспечение. Разворачивается на тестовых серверах. Публикуется после тестирования. Однако это легко только для небольших систем. Где допустимо полное одновременное замещение платформы.

А вот когда у компании ее платформа представляет из себя сложную многоуровневую структуру, то проблема обновления становится совершенно иначе. Невозможно себе представить, чтобы все сотрудники компании как-то раз придя к себе на рабочее место увидели новые интерфейсы и сразу, без проблем, начали работать. Как бы кто не тестировал новую платформу, всегда остаются незамеченные ошибки. Ряд ошибок будет скрыт на уровне бизнес логики, т.е. ее заметят только сами сотрудники. Новые интерфейсы всегда требуют, как минимум, этап привыкания, но чаще требуют незначительных доработок функционала.

В итоге, крупная технологическая платформа требует отдельного метода по трансформации.

Авторский метод трансформации

Ключевой особенностью метода автора статьи является этап эмуляции прежних интерфейсов, систем, процессов. Такой подход выгодно отличается от привычных и распространенных методов обновления программного обеспечения.

Этап 1 - Оценка.

Любой новый проект начинается с изучения текущей обстановки в компании. Изучается текущий ИТ ландшафт, процессы, взаимодействия сотрудников. Проводится опрос заказчиков, руководителей отделов, ответственных сотрудников с целью выявить текущие проблемы и сформулировать требования к новой системе.

Отдельный акцент следует сделать на такие отделы, как маркетинг, отдел развития, отдел продаж, отдел поддержки клиентов. Именно тут нужно собрать данные о перспективах развития компании на среднесрочную и долгосрочную перспективу. Новая технологическая платформа должна не просто заменить текущую, но и обеспечить развитие компании в некотором обозримом будущем.

Этап 2 - Разработка плана.

Создается документ-концепция, где предельно подробно прописывается план работ. Разработка подобного документа уже в больше степени относиться к отрасли проектного управления. Например, очень подробно данная тема изучается в методе управления проектами по PMBOK [6]. Поэтому в данной статье автор сделает лишь оговорку, что в плане, в том числе, нужно предусмотреть следующие важные детали:

1. Любой разрабатываемый модуль новой системы должен быть полностью совместим со старой системой, т.е. стать своего рода “клоном”. Это нужно, чтобы на старте работы системы можно было с минимальными болями пройти этап запуска системы благодаря тому, что сотрудники некоторое время будут работать в привычных интерфейсах и процессах.

2. Для процесса обновления заложить дополнительный этап, который можно назвать “эмуляция”. Именно этот этап позволит провести безболезненное обновление

Этап 3 - Разработка.

Довольно очевидный этап для любого ИТ процесса. Особенность тут заключается в работе архитектора системы. Он должен проработать новую архитектуру так, чтобы с одной стороны обеспечить синхронизацию с действующей системой, а с другой полностью выполнить требования для новой системы.

Не всегда такой подход возможно реализовать внутри одного модуля. Например, когда технические требования к новой системе радикально отличаются от текущей системы. Тогда рекомендуется продумать вариант двойного обновления, т.е. создания промежуточного состояния модуля, который позволит сперва запустить эмуляции, а уже потом обеспечить переход на модуль новой архитектуры.

Этап 4 - Эмуляция.

Новый модуль запускается сперва в тестовой среде, а затем в боевой, но в режиме эмуляции. Т.е. (в идеальном случае) ни один сотрудник компании, ни один из других участков системы не заметит включение нового модуля вместо старого.

Для такого бесшовного перехода требуется выполнить следующие условия:

- Провести миграцию данных [7]. Т.е. первично наполнить модуль историческими данными.
- Наладить процесс потокового параллельного распределения данных на старый и новый модули [8]. Это позволит новому модулю начать работать в боевом режиме, но на тестовом окружении, т.е. без риска повлиять на боевую среду.
- (если это возможно) Реализовать параллельную работу двух модулей. Часто это помогает протестировать новый модуль, а в случае нахождения ошибок в данных, их всегда можно поправить из старого модуля (при условии, что его функционал не вызывает сомнения в работоспособности).
- После завершения всех тестов, в том числе с боевыми данными, произвести финальное включение модуля в боевую среду. При этом старый модуль перенести на тестовые сервера, где хранить его в рабочем состоянии еще некоторое время, для перестраховки.

Последний пункт выполняется для модуля в режиме эмуляции. Главная цель этапа - включить модуль новой архитектуры с минимальными изменениями для бизнес-процессов компании.

Этап 5 - Масштабирование.

По описанный выше схеме запускаем работу всех других частей новой системы. Так мы обеспечиваем постепенное замещение старых частей системы на новые.

Этап 6 - Развитие.

Завершив интеграцию всех новых модулей (или их значительной части) можно начинать развиваться. Переводить модули в новый режим работы, когда включаются все заложенные в него функции по новым требованиям компании. Включение этих функций происходит при плотной поддержке всех сотрудников компании, с обучением, где это необходимо. Если есть возможность запуск делать участками, то лучше так и сделать. Разовый запуск новой архитектуры хоть и не вызовет технических проблем, ведь модули уже и так работают в

боевом окружении, но почти наверняка вызовут изменения в процессах компании, в интерфейсах и т.п. Начнутся ошибки со стороны сотрудников, к этому нужно быть готовым.

Как видно из этапов, они все описаны больше со стороны менеджеров компании. Разумеется, основная нагрузка ложится на исполнителей данного заказа. Но именно менеджеры должны в целом понимать, как именно они могут решить свою задачу, какие есть варианты решения.

Заключение

Предложенный метод проведения технической трансформации компании является наиболее безболезненным для любой компании. Он крайне рекомендуется в тех случаях, когда в компании большой штат, сложная и устаревшая техническая платформа, и где переход на новую платформу не допустим в общем режиме.

Однако метод не идеален. Всегда существует баланс между ценой, сроком и качеством. Данный метод в большей степени нацелен на качество работы, жертвуя сроками и ценой. Хотя вопрос цены спорный, т.е. растягивая сроки проекта можно растянуть нагрузку на бюджет, а также задуматься о работе со штатными программистами вместо подрядчика.

Сроки работ по указанному методу увеличиваются минимум на 30% по отношению к привычным методам обновления. Это вызвано необходимостью пройти этап эмуляции, а возможно и этап создания промежуточных версий модулей.

Список литературы

1. Крымов С. М., Кольган М. В. Методология инновационных обновлений предприятий на основе информационных технологий //Креативная экономика. – 2018. – Т. 12. – №. 6. – С. 787-804.
2. Ильин И. В., Лёвина А. И., Дубгорн А. С. Цифровая трансформация как фактор формирования архитектуры и ИТ-архитектуры предприятия //Научный журнал НИУ ИТМО. Серия «Экономика и экологический менеджмент». – 2019. – №. 3. – С. 50-55.
3. Исаев Е. А., Коровкина Н. Л., Табакова М. С. Оценка готовности ИТ-подразделения компании к цифровой трансформации бизнеса //Бизнес-информатика. – 2018. – №. 2 (44). – С. 55-64.
4. Мычко С. И. Микросервисная архитектура //Информационные технологии. – 2019. – С. 166-168.
5. Казначеева Е. О. Модульный и компонентный подходы в программировании //Альманах научных работ молодых ученых университета итмо. – 2018. – С. 150-152.
6. Аксенова Е. А., Ординян В. С. Повышение качества проектной документации при внедрении технологии информационного моделирования и методологии планирования PMBOK //Фундаментальные и прикладные исследования в науке и образовании: сб. ст. по итогам Междунар. науч.-практ. конф. Уфа. – 2019. – С. 124-129.
7. Баканович К. Миграция данных: что проще? //Storage News. – 2011. – №. 4. – С. 48.
8. Королева Ю. А., Маслова В. О., Козлов В. К. Разработка концепции миграции данных между реляционными и нереляционными системами БД //Программные продукты и системы. – 2019. – Т. 32. – №. 1. – С. 63-67.

References

1. Krymov S. M., Kolgan M. V. Methodology of innovative renovation of enterprises based on information technologies. Creative Economy. - 2018. - T. 12. - No. 6. - pp. 787-804.
2. Ilyin I. V., Levina A. I., Dubgorn A. S. Digital transformation as a factor in the formation of the architecture and IT architecture of an enterprise // Scientific journal NRU ITMO. Series "Economics and Environmental Management". – 2019. – no. 3. - pp. 50-55.
3. Isaev E. A., Korovkina N. L., Tabakova M. S. Assessing the readiness of the company's IT department for digital business transformation // Business Informatics. – 2018. – no. 2 (44). - pp. 55-64.
4. Mychko S. I. Microservice architecture //Information technologies. - 2019. - pp. 166-168.
5. Kaznacheeva E. O. Modular and component approaches in programming // Almanac of scientific works of young scientists of ITMO University. - 2018. - pp. 150-152.
6. Aksenova E. A., Ordinyan V. S. Improving the quality of project documentation when implementing information modeling technology and PMBOK planning methodology // Fundamental and applied research in science and education: coll. Art. following the results of the International scientific-practical. conf. Ufa. - 2019. - pp. 124-129.
7. Bakanovich K. Data migration: what is easier? //Storage News. – 2011. – no. 4. - pp. 48.
8. Koroleva Yu. A., Maslova V. O., Kozlov V. K. Development of the concept of data migration between relational and non-relational database systems // Software products and systems. - 2019. - T. 32. - No. 1. - pp. 63-67.