Фазы жизненного цикла информационных систем

Управление жизненным циклом информационных систем

3.5. Развертывание и внедрение

3.5. Развертывание и внедрение

- Закупка и настройка требуемой ИТ-инфрастуктуры
- Ввод начальных остатков
- Обучение пользователей
- Развертывание системы на рабочих местах
- Основные виды тестирование
- Опытно-промышленная эксплуатация
- Приемо-сдаточные испытания и интеграционное тестирование

Этап развертывания системы является одним из наиболее важных с точки зрения распределения ответственности за работы между заказчиком и подрядчиком. Исполнитель выполняет основную задачу — инсталляцию модуля на рабочий сервер с перенесением на него итоговой конфигурации с тестового сервера. Однако при верном последовательном обучении и вовлечении специалистов заказчика в процесс создания и внедрения системы остальные операции могут быть осуществлены ими:

- предшествующее работам обеспечение охвата всех автоматизируемых рабочих мест пользователей функционирующей ЛВС;
- настройка рабочего сервера;
- определение перечня и количества рабочих мест, которые необходимо развернуть для ОПЭ;
- развертывание рабочих мест пользователей с определением прав доступа;
- подготовка наиболее актуальной информации по каждому модулю для ввода в систему (включая подготовку, дополнение и выверку справочников, которые могли потерять актуальность за время между тестовой эксплуатацией и развертыванием системы);
- организация поддержания данных системы в актуальном состоянии (в том числе при интеграции с другими системами).

Целью данного этапа является подготовка модуля системы к опытно-промышленной эксплуатации — подготовка конечных пользователей, ввод начальных данных, подготовка приказов по предприятию заказчика о передаче модуля системы в опытно-промышленную эксплуатацию.

На этом этапе нужно:

- обеспечить безусловное выполнение условий готовности модулей системы к сдаче в опытно-промышленную эксплуатацию (закрепленных в отдельном документе);
- ввести и выверить начальные данные (например, начальные остатки) по каждому модулю для ввода информации в систему;

- отработать на рабочих местах пользователей модуля системы практические действия пользователей в параллельном режиме с имеющимися приложениями;
- разработать, согласовать с подрядчиком и утвердить регламент взаимодействия подразделений заказчика, участвующих в опытно-промышленной эксплуатации модуля системы;
- осуществить интеграцию модуля с другими модулями или внешними системами, внедренными ранее;

- подготовить и издать приказ по предприятию заказчика о передаче модуля системы в опытно-промышленную эксплуатацию, который должен содержать:

— наименование модуля системы, проходящего опытно-промышленную эксплуатацию;

— наименование компании-исполнителя;

— сроки проведения опытно-промышленной эксплуатации;

— список должностных лиц со стороны заказчика и исполнителя, ответственных за проведение опытно-промышленной эксплуатации модуля;

— перечень подразделений предприятия заказчика, участвующих в проведении опытно-промышленной эксплуатации;

— порядок и сроки перевода персонала заказчика на работу в условиях функционирования модуля системы.

К моменту ОПЭ уже должны быть проведены следующие работы:

- модули системы успешно перенесены и функционируют на рабочем сервере и автоматизированных рабочих местах пользователей (с разделением прав доступа);
- подготовлены, дополнены и введены недостающие справочники по каждому модулю системы, проведена выверка введенных в систему недостающих справочников;
- заказчиком утверждены итоговые документы.

3.5.1. Закупка и настройка требуемой ИТ-инфрастуктуры

Важно учитывать, что для всех предприятий чрезвычайно высока ценность хранимой и обрабатываемой информации, а также стоимость времени простоя бизнеса. Поэтому к техническому обеспечению ИТ-систем компании должны предъявляться жесткие требования:

- по производительности — обеспечение приемлемого времени реакции с точки зрения пользователя, использующего данную систему;
- надежности и доступности — пользователи должны продолжать работу при выходе из строя единичных устройств, система не должна простаивать и не должно быть потерь данных;
- катастрофоустойчивости — простой системы должен быть минимальным, а все данные должны быть сохранены; допускается некоторое ухудшение производительности и увеличение времени реакции системы;
- безопасности — система работает с данными, составляющими коммерческую тайну, и должна обеспечивать должный уровень защиты, вплоть до аттестации и сертификации решений и оборудования;

- масштабируемости — система должна иметь возможность приспосабливаться к увеличивающейся нагрузке со сравнительно малыми архитектурными изменениями или совсем без них.

Все эти требования к характеристикам решения в процессе подготовки ИТ-инфраструктуры детализируются на гораздо более подробном уровне. Ниже мы рассмотрим пример описания конкретных действий по обеспечению масштабируемости решения.

Пример

Основной объем обработки проводится при помощи рабочих процессов сервера приложений (диалог, фон, обновление, очередь или подкачка данных). Эти процессы запускаются на инсталляциях сервера приложений (первичная и дополнительная инсталляции дополнительного сервера приложений, за исключением процессов постановки в очередь, которые запускаются на инсталляции центральных сервисов). Вся обработка, связанная с базой данных, контролируется инсталляцией базы данных. Это значит, что критически важно обеспечить высокую масштабируемость инсталляций сервера приложений и базы данных. Масштабируемость этих инсталляций обеспечивается следующим образом.

- Сервер приложений масштабируется по горизонтали путем добавления дополнительных серверных хост-узлов, на которых выполняются дополнительные инсталляции приложений. Отправка запросов на инсталляции сервера приложений выполняется при помощи сервиса отправки сообщений, работающего на инсталляции центральных сервисов.
- Инсталляция базы данных масштабируется по вертикали путем адаптации необходимых аппаратных ресурсов (увеличения мощности процессора, расширение памяти и т.д.).

К ИТ-инфраструктуре в таком случае можно отнести следующие компоненты.

Вычислительная инфраструктура:

- базовые инфраструктурные сервисы. Описание и процесс организации таких инфраструктурных сервисов, как IP-телефония, электронная почта, резервное копирование, антивирусная защита и антиспам, служба каталогов AD, служба удаленных рабочих столов, сетевая печать и пр. Данные приложения автоматизируют вспомогательные задачи конечных пользователей, поэтому к этому классу в первую очередь относятся приложения, связанные с пользовательскими коммуникациями;
- серверное и пользовательское оборудование, системы хранения данных (СХД). Определение перечня пользовательского и серверного оборудования и систем хранения данных;
- серверные площадки и центры обработки данных (ЦОД). Планирование типовых центров обработки данных и серверных комнат, вплоть до составления некой усредненной модели серверной комнаты/ЦОД;
- системное ПО. К этой категории относятся описания операционных систем и программных платформ как для серверов, так и для оборудования, доступного конечному пользователю. Помимо этого, в данном разделе рассматриваются службы виртуализации.

Сетевая инфраструктура:

- инфраструктура локальных вычислительных сетей (ЛВС). Данный компонент должен представлять собой типовое решение по организации внутренней локальной вычислительной сети. Именно этот элемент является ключевым для обеспечения связи между оборудованием в офисах, филиалах, ЦОДах и других типах помещений. Важно, что в случае производственной или любой другой компании со значительной долей технологических систем проводится физическое и логическое разделение локальной сети на два сегмента, каждый из которых эксплуатирует только собственное сетевое оборудование. Таким образом, в корпоративной сети работают все бизнес-пользователи, а в технологической располагаются основные сервисы (оборудование, пользователи), поддерживающие технологическую составляющую и процессы компании;
- корпоративные сети передачи данных (КСПД);
- телефония, ВКС, подключение к сети Интернет.

Инженерная инфраструктура: устройства бесперебойного питания, электропитания, охлаждения, кабельная инфраструктура.

Можно упомянуть, что среди современных тенденций построения ИТ-инфраструктуры — использование нескольких уровней абстракции для вычислительных комплексов, сетей передачи данных с целью повышения гибкости и обеспечения регулирования потребления ресурсов (что снижает в конечном итоге суммарную стоимость оборудования).

Действия на этапе построения необходимой для системы ИТ-инфраструктуры складываются из следующего.

1. Закупка оборудования — данный шаг обычно значительно увеличивает время подготовки технической платформы, так как закупка может быть связана с проведением конкурсов, заключением договоров, ожиданием поставки оборудования. Все эти шаги могут приводить к потерям времени, от нескольких недель до нескольких месяцев.

2. Подготовка инфраструктуры — расчет возможности установки оборудования в серверное помещение, выделение и конфигурирование сетевых портов, планирование мощностей компонентов инженерной инфраструктуры серверных и пр.

3. Аппаратное конфигурирование — получение оборудования, его конфигурирование и установка (в том числе тестовый сервер, рабочие станции, вспомогательное оборудование).

4. Установка системного программного обеспечения — система управления базами данных, операционная система, генераторы отчетов (либо установка базового программного обеспечения, реализующего слой виртуализации).

5. Конфигурирование пользовательских рабочих мест — рекомендуется определить несколько стандартных конфигураций пользовательских рабочих мест, которые позволяли бы обеспечивать приемлемое качество работы пользователей. Это дает возможность не только упростить поддержку клиентских рабочих мест, но и, например, настроить систему мониторинга для проверок ключевых параметров работы приложений на всех конфигурациях пользовательских рабочих мест, используемых в компании.

В результате команда внедрения со стороны подрядчика получает развернутую функционирующую стендовую локальную вычислительную сеть, удовлетворяющую требованиям ТЗ, и программно-аппаратную платформу, уже готовую к последующей инсталляции системы.

3.5.2. Ввод начальных остатков

Данный этап, актуальный для систем финансового управления, является логическим продолжением проведенной ранее миграции данных (со вводом справочников и первичной информации). Формирование начальных остатков проводится уже после настройки параметров приложения. На их основе составляется начальный баланс, с помощью которого проверяется, насколько корректно были введены данные. В зависимости от системы при вводе и выверке данных может проводиться либо редактирование сальдо счета (субсчета), либо формирование фиктивных проводок. Делаются записи по дополнительным регистрам. При этом проводится тщательный контроль реакции программы на все операции, проверки сбалансированности сальдо, проверки каждого счета (в том числе аналитических).

3.5.3. Обучение пользователей

Для минимизации сопротивления, которое может встречаться со стороны специалистов организации при внедрении «очередной» системы, обучение проводится в три основных этапа.

1. Выделение и обучение ключевых пользователей.
2. Обучение рядовых пользователей.
3. Повышение квалификации или переобучение.

Рассмотрим эти этапы подробнее. В первую очередь необходимо объяснение принципов работы уже настроенной системы перед первым тестированием с ключевыми пользователями.

Следующая веха, инсталляция на стендовый сервер, важна как промежуточный этап, на котором снова будут привлекаться ключевые пользователи, только уже для описания работы и интерфейсов в гораздо более детализированном виде, для идентификации и устранения всех остающихся критических замечаний перед ОПЭ. Разумеется, к этому моменту все ИТ-специалисты, которые будут заниматься поддержкой системы, уже должны в деталях знать внутреннюю архитектуру системы и быть способными проводить ее доработку (или формировать требования на доработку) в случае необходимости.

И наконец, наиболее масштабное обучение организуется уже для всех остальных сотрудников, которые будут работать с системой. В данном случае система представляется не с точки зрения внутренней архитектуры, а с точки зрения пользователя, т.е. описываются принципы ввода, обработки и получения данных.

Для организации процесса обучения необходимо выполнить следующие шаги.

Шаг 1. Выделение и обучение ключевых пользователей. В первую очередь выделяются ключевые пользователи — один-два специалиста от каждого структурного подразделения (деление организации может выполняться не с организационной, а с логической точки зрения).

Важно, что в зависимости от масштаба и специфики системы эти специалисты могут работать не только с пользовательскими аспектами системы, но и самостоятельно разворачивать функциональные компоненты и модули, конфигурировать их по требованиям пользователей. Именно поэтому их обучение, аттестацию и вовлечение в работу крайне важно проводить гораздо раньше ОПЭ, ведь основное число сотрудников увидят систему и ознакомятся с принципами работы в ней лишь во время опытно-промышленной и промышленной эксплуатации.

Проведение учебных курсов по основам работы с системой для сформированной группы ключевых пользователей (по функциональным направлениям) организуется специалистами исполнителя (включая аттестацию). В качестве формы организации обучения может быть выбрана очная (преподавателем от команды исполнителя в офисе компании заказчика или подрядчика) либо дистанционная (когда личное присутствие преподавателя нецелесообразно или невозможно). Все популярнее становится проведение вебинаров и телеконференций, позволяющих экономить временные и денежные ресурсы.

Их поддержка будет крайне важна на следующем этапе при проведении тренингов для остальных пользователей: помимо предоставления комментариев по пользованию системой и о необходимости доработок, они могут помочь остальным пользователям разобраться с работой в системе и не обращаться каждый раз к команде сопровождения со стороны подрядчика.

Шаг 2. Обучение рядовых пользователей. К этому моменту необходимо, чтобы достаточное количество специалистов группы внедрения от заказчика было способно самостоятельно разворачивать функциональные компоненты системы или модуля на рабочих местах пользователей, проводить обучение конечных пользователей, оказывать консультации по стандартной конфигурации системы, а также сопровождать ее в оперативном режиме.

Принципы проведения тренингов могут значительно различаться в зависимости от масштабов компании, проекта/системы и пожеланий заказчика, однако на практике они чаще всего проводятся несколько раз, не более чем для 10—15 человек единовременно. При этом группы обычно формируются под каждую отдельную роль пользователя. Основные темы включают общий обзор задач системы, рассмотрение ее интерфейса, основных возможностей, затем специфической для конкретной роли пользователя функциональности и секции «вопрос — ответ». Иногда после этого проводится настройка необходимых параметров (например, интеграции с учетными записями Outlook, настройка синхронизации) уже на местах пользователей, однако необходимость этого зависит от конкретной системы.

Шаг 3. Повышение квалификации и переобучение. Следует учитывать также необходимость последующего периодического обучения с целью повышения квалификации сотрудников.

Условно можно выделить три основные категории такого обучения.

1. Плановое обучение после изменений в системе (при добавлении дополнительных возможностей, внедрении нового модуля и т.п.).
2. Плановое повышение квалификации: для совершенствования знаний работников, уже обладающих определенными профессиональными навыками работы с системой.
3. Обучение по запросу — относится к сфере поддержки пользователей на этапе эксплуатации системы и проводится в случае, если пользователь в своей работе столкнулся с конкретными проблемами или нуждается в проведении методического занятия.

Отметим, что для работы с некоторыми классами систем (как правило, если их деятельность связана с управлением объектами, представляющими в случае выхода из-под контроля опасность для жизни и здоровья людей) просто проведения обучения недостаточно. В таких случаях организуются дополнительные тестирования знаний, успешное прохождение которых дает допуск к пользованию системой при предоставлении персонального сертификата. Примером могут являться системы, автоматизирующие управление безопасностью железнодорожного движения и авиаперевозок, запуск космических объектов, химическое производство и подобные виды деятельности.

И разумеется, при планировании обучения следует учитывать такие аспекты, как опыт сотрудников, предполагаемые и видимые знания, а также навыки руководства и управления, организационные способности.

3.5.4. Развертывание системы на рабочих местах

Следующим шагом организуется развертывание модуля для дальнейшей его опытно-промышленной эксплуатации. Успех данного этапа целиком зависит от проведенной ранее работы. Так, если уже был реализован пилотный проект в другом подразделении, филиале или в рамках другой функциональной области, этот опыт поможет компании воспользоваться рекомендациями по развертыванию системы и выполнить оптимальные настройки. Кроме того, инсталляция проводится еще до тестирования, а значит, результаты используются исключительно в целях совершенствования системы в так называемой песочнице (*sandbox*).

«Песочница» (*sandbox*) — закрытое для доступа извне виртуальное пространство, в котором можно работать с ПО без риска изменения системных файлов, в связи с чем они используются для запуска кода, еще не прошедшего тестирование.

Настройки «песочницы» и ее элементы дублируют основные элементы и возможности реальной системы для первоначальной отладки и проверки кода в виртуальной среде перед его переносом в продукт. Это помогает избежать значительного числа рисков и сбоя настроек сложной конфигурации системы.

Набор ресурсов системы, выделяемый для «песочницы», жестко ограничен в части доступа к объему памяти, к сети, обмену информацией с основной системой. Это обусловлено в первую очередь нестабильностью работы тестируемого кода и значительной нагрузкой на систему. По этой причине «песочницы» часто используют исключительно для проверки отдельных критических частей кода.

Сделанные в «песочнице» настройки используются при полноценном развертывании модуля системы после тестирования перед ОПЭ. Модули системы устанавливаются на стендовый сервер с проведением основных необходимых настроек под подлежащие автоматизации бизнес-процессы предприятия заказчика. По результатам этих действий подрядчик подготавливает описание оптимальных настроек сервера БД и рекомендации по развертыванию рабочих мест, которое и проводится следующим шагом.

Перед развертыванием системы необходимо формирование командой внедрения следующей информации:

- выделение основных блоков функциональности, их запусков, взаимозависимостей;
- выделение основных блоков работ, требуемых для запуска функциональности системы, и их взаимозависимости;
- расстановка контрольных точек для верхнеуровневого мониторинга процесса;
- информация по исполнителям работ (зонам ответственности).

При переходе к стадии эксплуатации должны быть осуществлены следующие действия:

- проверка необходимых предпосылок для перехода в продуктивную систему;
- описание мероприятий по переходу в продуктивную систему;
- описание критериев, на основании которых будет приниматься решение о начале работы пользователей;
- описание порядка действий в случае возникновения ошибок, сбоев или нарушения работы системы.

При развертывании системы на рабочих местах выполняется следующее:

- установка и подключение технических средств;
- установка и конфигурация ПО;
- развертывание баз данных, служебных программ;
- доработка интерфейса и процессов, настройка профилей пользователей, настройка отчетов;
- установка прав доступа;
- пуско-наладочные работы и окончательная отладка конфигурации.

Потенциальные сложности могут возникнуть также, если несколько смежных подпроектов (например, в части разных модулей системы) реализуются несколькими подрядчиками. Тогда следует уделять пристальное внимание управлению интеграцией всех проектов на протяжении всего времени их реализации и при подготовке к ОПЭ – в особенности.

3.5.5. Основные виды тестирования

Прежде чем приступать к промышленной эксплуатации системы, важно проверить ее работоспособность во всех предусмотренных спецификацией режимах, а также в экстремальных условиях. Если планируется одновременный распределенный ввод данных сотней пользователей, необходимо, как минимум, проверить правильность обработки системой одновременной активности именно не менее сотни пользователей. Подобная идея лежит в основе нагрузочного тестирования.

Нагрузочное тестирование необходимо для предсказания поведения системы в реальных и экстремальных условиях, выявления ошибок, отслеживания производительности и доступности при изменении различных параметров работы с системой. Среди других его задач:

- оценка производительности и работоспособности приложения на этапах:
  - разработки,
  - опытно-промышленной эксплуатации,
  - сопровождения и эксплуатации (выпуск новых релизов, патчей);
- оптимизация приложений;
- подбор оптимальной программно-аппаратной платформы и конфигурации сервера.

Перечислим еще несколько ключевых категорий тестирования.

Тестирование производительности (*load testing*) — моделирование ожидаемой интенсивности использования путем распределенной работы большого числа пользователей с различными модулями системы.

Среди возможных целей этого типа тестирования:

- определение времени выполнения операций с заданной интенсивностью (на разных нагрузках);
- определение максимально достижимой производительности системы.

Стрессовое тестирование (*stress testing*) — определение стабильности системы при интенсивности работы, превышающей плановые или стандартные значения. Стрессовое тестирование по своей сути проверяет, возвращается ли (и насколько быстро) система после запредельной нагрузки к нормальному режиму работы, тестируется способность системы к регенерации.

Стрессовое тестирование особенно важно для компаний, в которых стоимость отказа системы в экстремальных условиях может быть очень велика, а стандартного тестирования разработчиками недостаточно для эмуляции тех условий, в которых произойдет отказ системы.

Среди возможных целей этого типа тестирования:

- оценка динамики производительности при нестандартных ситуациях: аварийном изменении серверной конфигурации либо резком повышении числа пользователей и выполняемых одновременно операций;
- определение условий и скорости возвращения системы к нормальному режиму работы после окончания стрессового тестирования.

Тестирование надежности (*reliability testing*) — определение длительности бесперебойной и безошибочной работы системы. Цели этого типа тестирования следующие:

- оценка стабильности системы при многочасовом тестировании со стандартным средним уровнем нагрузки для определения:

— утечек памяти,

— некорректных конфигурационных настроек,

— случаев перезагрузки сервера и других требующих устранения проблем.

Конфигурационное тестирование (*configuration testing*) — оценка степени влияния на производительность изменений в конфигурации и различной балансировки нагрузок. Конфигурационное тестирование относится как к серверному уровню (совместимость с окружением), так и к клиентскому уровню (например, кросс-платформенное или кросс-браузерное тестирование). Цели этого типа тестирования:

- оценка обработки ошибок и исключительных ситуаций, а также «узких мест» отдельных модулей и компонентов системы при непропорциональных нагрузках;
- определение оптимальной конфигурации оборудования, которая бы обеспечивала требуемые характеристики производительности и отклика системы;
- проверка определенных компонентов системы на предмет совместимости с указанным в спецификации оборудованием, операционными системами и программными продуктами внешних поставщиков.

Последовательности действий для проведения данного вида тестирования такова.

1. Создается матрица покрытия с описанием всех возможных конфигураций системы.
2. Выполняется приоритезация конфигураций.
3. И наконец, в ходе тестирования в соответствии с имеющимися приоритетами организуется проверка всех основных конфигураций.

Объемное тестирование (*volume testing*) — тестирование программного обеспечения системы на предмет стабильности обработки определенного объема данных. Цели этого типа тестирования:

- измерение динамики выполнения операций при увеличении объемов данных в базе данных, постепенном росте числа запросов пользователей;
- определение максимального числа пользователей, которые могут работать с приложением без снижения производительности.

Отсутствие или ошибки проведения подобных видов тестирования значительно увеличивают риски и могут даже приводить к ситуациям, когда после внедрения срок эксплуатации систем не превышает даже года в связи с невозможностью конфигурации системы справляться с необходимыми задачами и нагрузками. Именно для нивелирования подобных рисков необходимо своевременно организовывать тестирование еще на этапе внедрения (хотя его проведение возможно (и часто проводится) и для уже эксплуатируемых систем).

3.5.6. Опытно-промышленная эксплуатация

Опытно-промышленная эксплуатация представляет собой тестирование в полной функциональности и полной нагрузке для определенного количества пользователей. Ее основной целью является апробирование работы пользователей в системе в реальных производственных условиях. Это означает, что если по спецификациям системы предполагается, что она будет

обрабатывать 500 000 записей в день, необходимо проверить корректность обработки именно этого количества записей. При этом важные задачи:

- тестирование модуля системы в условиях, максимально приближенных к реальным условиям промышленной эксплуатации (в том числе, при необходимости, в интеграции с другими модулями или внешними системами) — нагрузочное тестирование, рассмотренное ранее;
- проведение множественных расчетов по фактической производственной информации с применением соответствующих программно-аппаратных средств, предусмотренных техническим проектом (проектным решением);
- достижение наиболее полного охвата бизнес-процессов, автоматизируемых подразделений предприятия.

При условии соблюдения ранее определенных требований к предыдущим этапам и их успешном завершении в первую очередь необходимо провести:

- ввод наиболее актуальной информации по каждому модулю ввода в систему (подготовка которой проведена на предыдущем этапе);
- отработку действий пользователя в параллельном режиме с уже имеющимися приложениями (это проводится непосредственно на рабочих местах пользователей и включает дополнительную, но необходимую нагрузку для пользователей, которые в короткий период времени должны адаптироваться к работе в другой системе (а также помочь адаптировать ее для оптимальной и эффективной работы));
- интеграцию внедряемого модуля с уже существующими модулями и (или) внешними системами, внедренными ранее;
- разработку и согласование регламента взаимодействия внутренних подразделений предприятия-заказчика в рамках ОПЭ и дальнейшей промышленной эксплуатации (помимо схемы и порядка передачи функций с обязательным определением зон ответственности за актуальность и корректность данных).

Пример

Несмотря на то что официально «владельцем» системы будет являться одно подразделение (или даже одно ответственное лицо), ввод и редактирование информации будут осуществляться распределенно и часто одновременно различными категориями пользователей. Соответственно, существует практика определения зон ответственности за различными подразделениями, которые вне зависимости от источника ввода будут обладать исключительными правами на удаление и изменение информации. Этот механизм может быть организован многими способами, самым «простым» из которых является четкое определение алгоритма взаимодействия подразделений в рамках работы с системой (сотрудники будут сами знать, какую информацию в какой момент они будут вводить и обрабатывать). Во многом подобный механизм исходит из принципов «презумпции невиновности», предполагая что каждый специалист достаточно компетентен и отвечает за каждое совершенное им действие (как в целом в работе, так и в системе).

Другим, менее распространенным, способом управления взаимодействием подразделений (и главное — контроля данных) является распределение прав доступа таким образом, что у наиболее важной информации будет только один источник ввода/проверки. И хотя другие пользователи могут изменять поле (например, вносить данные об инвестициях и договорах), в системе эта информация будет «висеть» в режиме ожидания до момента ее одобрения/авторизации действия ответственным за поле пользователем. Однако, несмотря на значительную гарантию достоверности данных, подобная схема крайне неэффективна с точки зрения использования ресурсов, особенно в условиях большого числа и масштаба операций.

Основная часть (как по объему, так и по трудозатратам) проводимых в ОПЭ мероприятий предусматривает сопровождение заказчиком системы на рабочих местах пользователей с формированием перечня замечаний ключевых пользователей (при их «переводе» из бизнес-терминологии в более техническую область) и устранением этих замечаний.

Пример

Замечания пользователей:

В список партнеров нельзя добавить университеты (нет пункта в списке).

Требование к реализации подрядчиком:

Создать в типе организаций «Прочие» наследующий его свойства подтип «Университет» с дополнительными полями «Ректор», «Факультет», «Кафедра». Настроить общедоступный отчет по всем университетам.

Устранением замечаний и внесением изменений в систему на этом этапе (длящемся, как и в случае тестовой эксплуатации, не менее одного месяца) занимается команда внедрения со стороны исполнителя, которая помимо простого соответствия системы всем требованиям и бизнес-процессам должна удостовериться в устойчивости работы системы при нагрузке, максимально приближенной к реальной промышленной нагрузке этапа эксплуатации (с исключением риска отказа системы в момент ее пика). Чтобы перейти к полноценной промышленной эксплуатации, система также должна безошибочно функционировать в параллельном режиме и при интеграции с внешними приложениями в течение всего срока ОПЭ. При наличии аналогичных *legacy*-приложений об успешном внедрении можно говорить, если результаты обработки данных в них и в новой системе совпадают и (или) различия объяснимы и некритичны.

Для окончания ОПЭ необходимо подтверждение того, что модуль системы функционирует в параллельном режиме с существующими внешними приложениями (при их наличии) в течение срока опытно-промышленной эксплуатации. Вторым условием является подтверждение, что любые отклонения (в случае их обнаружения) признаны некритичными и объяснимыми.

3.5.7. Приемо-сдаточные испытания и интеграционное тестирование

Именно передача системы из ОПЭ в промышленную эксплуатацию (в частности, путем проведения приемо-сдаточных испытаний) является целью комплексного проекта по автоматизации деятельности компании. Приемо-сдаточные испытания такого типа называют также интеграционным тестированием. Такое тестирование обеспечивает комплексную проверку реализации проектных решений системы.

Цели интеграционного тестирования:

- проверка корректности функционирования бизнес-процессов и функций;
- уточнение настроек бизнес-процессов в системе;
- уточнение и доработка настроек пользовательского интерфейса;
- проверка функций начальной загрузки в систему основных и переменных данных;
- проверка полноты переносов настроек посредством транспортной системы;
- проверка работоспособности интерфейсов с внешними системами;
- проверка корректности проектной и эксплуатационной документации;
- определение правильности функционирования системы на реальном объеме данных в реальном времени.

Тестирование проводится на продуктивном наборе данных для проверки работоспособности всей системы и правильности настройки интерфейсов взаимодействия с внешними системами.

Как уже было сказано, именно этот этап является переходным между этапом ОПЭ и реальной эксплуатацией. С организационной точки зрения перед началом промышленной эксплуатации необходимо удостовериться в наличии следующего набора документов:

- рабочий журнал опытно-промышленной эксплуатации с перечнем замечаний ключевых пользователей и результатами их устранения (заполняемый командой внедрения и согласовываемый затем с представителями заказчика);
- протокол об окончании ОПЭ и готовности к промышленной эксплуатации совместно с Актом приемки-передачи работ;
- приказ по предприятию заказчика о приеме системы в промышленную эксплуатацию (с составом функций, описанием программно-аппаратной платформы, списком ответственных за работу модуля и списком подразделений-пользователей, а также с порядком и сроками перехода персонала на работу в системе, включая аспект разработки и принятия новых форм документов в случае необходимости).

После этого использование системы уже не имеет признаков проектной деятельности и ограничений по срокам, содержанию или стоимости. Однако такие последующие этапы ЖЦИС, как сопровождение эксплуатации, модернизация и утилизация системы, будут по своей сути являться проектами и могут также выполняться не только за счет внутренних ресурсов компании, но и (что случается гораздо чаще) с привлечением внешних подрядчиков.

3.6. Эксплуатация

3.6. Эксплуатация

- Сопровождение эксплуатации
- Модернизация

Этап эксплуатации системы является наиболее ожидаемым для бизнес-заказчиков, поскольку только в этот момент они начинают получать возврат инвестиций и видеть реальный результат внедрения. Однако это и самый опасный этап, так как полученный результат может (и скорее всего, будет) не соответствовать новым ожиданиям и представлениям как о функциональности, так даже и о внешнем виде и интерфейсе системы. Технические специалисты и подрядчик фокусируются на решении задач бесперебойной работы, обслуживания баз данных, а в случае необходимости — на донастройке системы (что означает одновременное наступление стадии модернизации, продолжающейся параллельно с эксплуатацией). Но бизнес-заказчик и спонсор системы, как и все участники процесса сбора требований, часто имеют собственное представление о целевом состоянии системы. Именно поэтому критически важно с самого начала вовлекать в проект специалистов, переводящих пожелания бизнеса в задачи для разработчиков (данная роль в английской терминологии получила название IT/Business Relationship Manager).

Этап сопровождения

Стадия сопровождения, приводящая к изменениям системы в процессе эксплуатации (по окончании приемки работ), охватывает несколько аспектов:

- устранение замечаний, не приводящих к изменению ТЗ;
- обновления (по сути — новые версии системы), выпускаемые при накоплении критического объема доработок;
- увеличение производительности системы.

Важно отметить, что на сегодняшний день основной задачей компании-подрядчика является не просто настройка системы, но передача накопленных и базирующихся на опыте множества проектов знаний и методик работы с внедренным программным решением. По окончании проекта специалисты компании-заказчика должны быть способны самостоятельно осуществлять полноценную поддержку и развитие системы. В первую очередь это относится к развитию уже созданных модулей, совершенствованию функциональности, реализации новых задач для новых категорий пользователей, формированию аналитических отчетов и панелей мониторинга. Разумеется, часто заказчик принимает решение о продолжении сотрудничества с подрядчиком, внедрявшим систему, в части ее сопровождения. Это относится как к базовым функциям (построению дополнительных типов отчетов или изменению параметров настроек), так и к технологической и методической поддержке.

3.6.1. Сопровождение эксплуатации

Авторский надзор. В условиях промышленной эксплуатации в течение определенного договором времени после сдачи модуля системы со стороны исполнителя-подрядчика проводится наблюдение за его функционированием и, по мере необходимости, оказание помощи специалистам заказчика по устранению замечаний. В свою очередь, уже обученные ответственные за систему сотрудники заказчика в тесном взаимодействии с ключевыми пользователями формируют перечень замечаний, осуществляют технологические обновления модуля (системы) и ведут документирование эталонных версий модулей. При этом предприятие-подрядчик включается в работу только при необходимости значительных доработок либо если таковое определено заключенным контрактом на поддержку. Это объясняется тем, что ключевая функциональность, необходимая для успешной и бесперебойной работы системы, и программно-аппаратная база уже были сформированы на предыдущих этапах и специалисты заказчика уже в состоянии осуществлять стандартную техническую поддержку на своем предприятии самостоятельно.

Именно в это время устраняются основные возникающие замечания, ошибки и неточности первоначальных расчетов нагрузки на систему. Как правило, длительность этапа авторского надзора составляет не менее года, и, в отличие от фактора масштаба и сложности проекта, отсутствие или несвоевременное представление замечаний не влияют на продолжительность этапа авторского надзора за промышленной эксплуатацией модуля.

Для этой и последующих стадий сопровождения могут применяться специальные **метрики оценки работ** со стороны предприятия-подрядчика, отвечающего за сопровождение. Четыре основные метрики (актуальные для всего жизненного цикла) были выделены Институтом программной инженерии университета Карнеги-Меллон (SEI CMU): *размер*, *усилия*, *расписание* и *качество*.

Однако они могут дополняться и другими параметрами:

- *анализируемость*: оценка не предусмотренных изначально усилий или ресурсов, необходимых для диагностики недостатков или причин сбоев, а также для идентификации тех фрагментов программной системы, которые должны быть модифицированы;
- *изменяемость*: оценка усилий, необходимых для проведения заданных модификаций;
- *стабильность*: оценка случаев непредусмотренного поведения системы, включая ситуации, обнаруженные в процессе тестирования;
- *тестируемость*: оценка усилий персонала сопровождения и пользователей по тестированию модифицированного программного обеспечения.

Техническая поддержка. Техническая поддержка системы начинается после приема в промышленную эксплуатацию и может продолжаться до снятия с эксплуатации и утилизации системы. Как правило, варианты осуществляемой поддержки варьируются по процентам от стоимости приобретенного ПО в зависимости от набора оказываемых услуг, а объем и периодичность выполнения работ на данном этапе определяются отдельным договором.

В качестве поддержки могут предоставляться следующие услуги в различных комбинациях в зависимости от прописанных в договоре условий:

- консультирование по «горячей линии» (телефон, *e-mail*, *skype*) по определенному в договоре графику;
- консультирование по вопросам программно-аппаратной платформы системы;
- создание новых учетных записей пользователей системы;
- настройка форм отчетов и панелей мониторинга для определенных групп пользователей заказчика;
- диагностика и устранение неисправностей (с учетом гарантии на сами программные решения);
- уведомление о выпуске обновлений и их загрузка через Интернет;

- замена ключей электронной защиты, переустановка ПО в случае необходимости (например, при замене компьютерной базы заказчика);
- миграция данных и настройка интеграции с новыми системами предприятия-заказчика;
- помощь в формализации требований к новым программным решениям предприятия заказчика в части интеграции с внедренной системой либо в постановке и формализации задач (на уровне ТЗ) по включению новых бизнес-процессов в систему.

Постгарантийное сопровождение. Постгарантийное сопровождение предполагает заказываемые у производителя работы, не предусмотренные изначальным контрактом на автоматизацию процессов или системную интеграцию (рис. 3.2). Сопровождение программного обеспечения определяется стандартом IEEE Standard for Software Maintenance (IEEE 1219) как «модификация программного продукта после передачи в эксплуатацию для устранения сбоев, улучшения показателей производительности и (или) других характеристик (атрибутов) продукта, или адаптации продукта для использования в модифицированном окружении». Таким образом, функционирование программного продукта поддерживается на протяжении всего периода его эксплуатации.

.2. **Работы процесса сопровождения по стандарту IEEE 1219**

В рамках заключаемого на этом этапе договора исполнитель оказывает помощь по устранению замечаний и выделяет в случае необходимости специалистов для выполнения доработок (например, в части новой функциональности или отчетов). Однако если в течение предыдущих этапов ЖЦИС исполнителем проводились мероприятия по обучению заказчика и вовлечению его в проект, на фазе постгарантийного сопровождения специалисты заказчика будут способны:

- сопровождать модуль системы в ходе промышленной эксплуатации на участках;
- формировать перечень замечаний ключевых пользователей;
- выполнять работы по устранению возникших замечаний (оказывать помощь по устранению замечаний);
- осуществлять своевременные обновления модуля системы в компании исполнителя (прежде всего — технологические);
- хранить и вести эталонные версии модуля системы и программной документации (постгарантийное сопровождение старых версий может продолжаться и после выпуска новых версий программных продуктов, однако чаще поддерживаются исключительно новые версии).

Со стороны компании-исполнителя в данном случае осуществляются постановка задачи, анализ проблем в предметной области, планирование и реализация необходимых доработок и прочие активности.

В договорах на постгарантийное сопровождение, как правило, прописываются условия, касающиеся:

- возможности приобретения дополнительных клиентских лицензий;
- возможности поддержки устаревших версий программных продуктов, на которых основана система;
- оплаты командировочных расходов компании-подрядчика при необходимости выезда в филиалы других регионов и государств;
- скидок и особых условий при продлении контракта.

При рассмотрении работ по сопровождению системы будем использовать руководство SWEBOK в области знаний «Поддержка ПО» (*Software maintenance*) как содержащее достаточно полный набор материалов на эту тему.

SWEBOK приводит ряд процессов (работ, практик), которые являются уникальными для деятельности по сопровождению.

- **Передача** (*Transition*). Внутренние коммуникации разработчиков и группы поддержки для грамотной координации процесса передачи программного решения на сопровождение.
- **Принятие/отклонение запросов на модификацию** (*Modification Request Acceptance/Rejection*). Рассмотрение таких характеристик, как: объем и (или) сложность требуемых изменений, необходимые для их реализации активности. Решения по принятию или отклонению запросов на модификацию могут также основываться на анализе приоритетности, оценке обоснованности, отсутствии ресурсов (в том числе отсутствии возможности привлечения разработчиков к решению задач по модификации при реальном наличии такой потребности), внесении в план к реализации следующих релизов.

- **Средства извещения персонала сопровождения и отслеживания статуса запросов на модификацию и отчетов об ошибках** (*Modification Request and Problem Report Help Desk*). Поддержка конечных пользователей, в том числе анализ приоритетности и стоимости модификаций, связанных с поступившим запросом или сообщенной проблемой.
- **Анализ влияния** (*Impact Analysis*). Анализ возможных последствий изменений, вносимых в существующую систему, при идентификации всех связанных с ней систем и программных продуктов, на которых эти изменения могут потенциально отразиться.
- **Поддержка программного обеспечения** (*Software Support*). Консультирование пользователей, проводимое по их информационным запросам (request for information), например, в отношении бизнес-правил, проверки содержания данных и получаемых сообщений о проблемах (ошибках, сбоях, непредусмотренном поведении, непонимании принципов функционирования системы).

- **Контракты и обязательства.** Закрепленные в договорной форме обязательства по определенным параметрам оказания услуг сопровождения (в том числе классическое соглашение об уровне предоставляемого сервиса, SLA).

Обновление и релизы. Неотъемлемой частью сопровождения является выпуск обновлений и релизов программ/конфигураций. В них, в частности, могут быть реализованы новые правила обмена данными, настроены новые формы регламентированной отчетности, усовершенствованы интерфейсы, адаптированы функциональные возможности и произведены другие доработки.

Подобного вида изменения вносятся в ПО либо подрядчиком в рамках постгарантийного сопровождения при накоплении критического числа замечаний, либо вендором программного продукта в рамках планового выпуска его очередной версии. Это может проводиться в рамках подготовки нового технического дистрибутива, в рамках выпуска локализованной под конкретную страну или регион конфигурации, в рамках отраслевого решения, в рамках обновления определенного модуля и пр.

Если речь идет о *выпуске версий при накоплении критического числа замечаний*, то сопровождение подобного рода, как правило, складывается из следующих этапов.

1. Формирование заявок для специалистов, работающих в рамках сопровождения.
2. Планирование работ (выявление проблем и требований, объема и содержания задач).
3. Оказание услуг (согласование/уточнение требований, реализация работ, консультации, обучение, доработки).
4. Сдача-приемка работ.

Пример

В 2012 г. 1С выпустила новую конфигурацию «Бухгалтерия предприятия» (редакция 3.0), и в пресс-релизе были отмечены следующие внесенные изменения:

1) повышение удобства работы. Новые пиктограммы в интерфейсе пользователя, возможность настройки панелей действий и навигации. Появление внутренних ссылок для каждого из объектов информационной базы;

2) развитие функциональности учета заработной платы. Начисление заработной платы, НДФЛ и страховых взносов стало проводиться одним документом автоматически при начислении заработной платы. Разделы «Зарплата» и «Кадры» объединены в один раздел. за счет чего все кадровые документы стали доступны на карточке сотрудника;

3) развитие прав доступа. Добавлена функция доступа к информационной базе в режиме просмотра без прав на внесение изменений;

4) работа через Интернет по модели SaaS. Для работы начиная с версии 3.0 более не требуется установка платформы и информационных баз на компьютере пользователя. Вместо этого доступен запуск программы через веб-браузер с сайта компании-поставщика «облачного» сервиса;

5) повышение удобства работы при выполнении длительных операций. Возможность выполнения операций в фоновом режиме;

6) новые средства защиты персональных данных. Изменения внесены в связи с требованиями Федерального закона.

Таким образом, важность уделения должного внимания выпускаемым обновлениям несомненна, и активности по их реализации также несут в себе все признаки проектной деятельности.

Увеличение производительности системы. Во время проектирования и внедрения системы для уточнения ее возможностей определяется, как быстро приложение должно работать, какой объем памяти и какую загрузку процессора использовать, какие другие характеристики системы можно считать приемлемыми.

В течение срока эксплуатации и сопровождения системы ее окружение меняется, и прежние параметры производительности становятся недостаточными для успешной работы. В частности, это может относиться к таким аспектам, как:

- *время отклика* — время, необходимое серверу для выдачи ответа на произведенный запрос;
- *пропускная способность* — число запросов, которые могут быть обработаны за единицу времени; часто при измерении данного параметра определяется число запросов / логических транзакций в секунду;
- *использование ресурсов* — потребление приложением серверных/сетевых ресурсов (ЦП, память, дисковое пространство);
- *рабочая загрузка* — общее количество пользователей (либо только активных пользователей), объем данных и транзакций.

Соответственно, рано или поздно владельцы системы сталкиваются с необходимостью совершенствования этих характеристик. Иногда подобный подход является проактивным, когда принимаемые меры направлены на оптимизацию, снижение операционных расходов. Однако чаще эти действия являются уже реактивными и проводятся, когда стоимость владения системой и затраты на нее становятся неприемлемыми и поднимается вопрос о целесообразности ее эксплуатации.

Проводимое в рамках сопровождения системное улучшение параметров получило название *инженерии производительности* (*performance engineering*). Изначально данная область разрабатывалась в рамках направления системотехники и включала в себя единый набор ролей, знаний, практик, инструментов и результатов каждого этапа ЖЦ системы. Основная цель состояла в том, чтобы гарантировать соответствие создаваемого программно-аппаратного решения нефункциональным требованиям к его производительности.

В настоящее время область применения инженерии (и реинжиниринга) производительности в значительной части сфокусирована на проектах сопровождения и модернизации системы, когда требования уже проверены временем в ходе эксплуатации и могут быть определены более конкретно.

Среди задач инженерии производительности:

- повышение окупаемости бизнеса с помощью обеспечения своевременной обработки необходимых объемов транзакций;
- своевременное выявление потенциальных проблем масштабирования и производительности и их устранение;
- снижение стоимости поддержки ПО и внеплановых затрат через своевременное выявление проблем производительности;
- предотвращение задержек, вызванных проблемами с производительностью, при развертывании систем.

При планировании производительности необходимо понимать, какие ресурсы системы доступны в текущий момент. Для этого оцениваются:

- *сетевое обеспечение* (в том числе пропускная способность);
- *аппаратное обеспечение* (характеристики серверов, рабочих станций);
- *зависимости ресурсов* (число доступных соединений баз данных, веб-сервисов);
- *совместно используемые ресурсы* (в том числе принципы распределения ресурсов между разными элементами системы);
- *проектные ресурсы* (финансовые и человеческие ресурсы) для поддержки системы и осуществления проекта по повышению ее производительности.

В процессе могут быть задействованы администратор системы, системный архитектор, разработчики, тестировщики и другие члены проектной команды. Именно они производят моделирование/прототипирование желаемого результата, проектирование новых параметров системы, доработку программного кода, мониторинг, создание надежных тестов производительности, различного рода тестирование и настройку системы до необходимых значений.

Пример

Вполне конкретное требование к определению планового значения отклика системы на запрос может выглядеть следующим образом:

«Для сценария использования А отклик системы на корректный запрос пользователя не должен превышать:

- 5 секунд для нагрузки в 250 активно использующих систему пользователей (200 онлайн-пользователей в 95% случаев);
- 10 секунд для пиковой нагрузки в 500 активных пользователей (400 онлайн-пользователей в 90% случаев)».

В общем случае для формирования подобных требований определяется типичный бизнес-день, разбитый на часы, для формирования представления о том, какова динамика нагрузки на систему в течение дня. В результате можно получить конкретные предложения по улучшению (например,

«распараллелить работу модуля, запустив ее на четырех процессорах вместо одного») и конкретные результаты (например, «повысить таким образом эффективность загрузки ресурсов модуля, получив прирост производительности в 5 раз»).

3.6.2. Модернизация

Стратегии управления *legacy*-системами. В течение периода эксплуатации системы происходят неизбежные изменения как вне организации (в том числе изменение рынка, контрагентов), так и внутри организации (к примеру, смещение бизнес-приоритетов в другую индустрию, на другой рынок и пр.). Растет объем обрабатываемой информации, объем файлов, снижается пропускная способность, оборудование выходит из строя и появляется новое — все эти факторы неизбежно влияют на актуальность самой системы и решаемых ей задач. В то время когда системы уже не способны с должной степенью эффективности решать задачи бизнеса, говорят об *«унаследованных»* системах.

Унаследованные системы (англ. *legacy systems*) — системы, более не соответствующие текущим потребностям бизнеса, но по-прежнему эксплуатирующиеся компаниями. Как правило, в их основе лежат устаревшие технологии и платформы, и не представляется возможным далее совершенствовать и видоизменять их для соответствия требованиям безопасности, новому аппаратному обеспечению, обновленным операционным системам.

Чаще всего модификация подобных приложений требует значительных затрат времени и финансовых средств, в то время как их замена может требовать даже реинжиниринга бизнес-процессов организации. В таком случае требуется доработка системы, для чего необходимо заново пройти все фазы жизненного цикла — от сбора и формирования требований до сопровождения и эксплуатации — и, соответственно, снова определять сроки, ресурсы и содержание. Результат подобной доработки будет уникальным, и потому можно говорить об отдельном проекте по модернизации. Этим проектом может заниматься как команда внедрения, так и абсолютно другие компании-подрядчики (особенно если речь идет о модернизации системы, созданной несколько лет назад, ведь за этот срок организация-подрядчик может уже прекратить свое существование).

Модернизация системы подобного рода позволяет продлить срок ее эксплуатации, снизить совокупную стоимость владения (TCO), расширить функциональные возможности, усовершенствовать программно-аппаратную платформу минимальными доступными средствами *без* остановки процесса эксплуатации системы. В ходе модернизации устраняются многие проблемы, возникавшие при эксплуатации. Однако необходимость реализации определенных доработок, число которых уже достигло критического значения, является лишь одной из возможных причин старта модернизации системы.

Другим важным фактором (триггером) является аудит информационных систем, проводимый организацией самостоятельно или с привлечением внешних экспертов и консультантов. Результаты аудита могут сочетаться с реинжинирингом бизнес-процессов компании, разработкой новой целевой прикладной архитектуры или же обновлением ИТ-стратегии в целом. Принятию решения о модернизации в обязательном порядке должно предшествовать определение сроков, стоимости и состава работ по доработке, так как возможна ситуация, когда списание системы и внедрение новой в конечном итоге окажется лучшим вариантом, чем изменение существующей. Ведь в целом задачи модификации, будучи на первый взгляд простыми (хотя и трудоемкими), представляются компаниям, как правило, несвоевременными либо неперспективными. Поколения сотрудников (как заказчика, так и компании-подрядчика) меняются, работавшие при внедрении специалисты уходят, а в случае некорректного документирования системы при ее создании результаты попыток совершенствования ПО могут быть достаточно плачевными.

Таким образом, среди внешних и внутренних факторов, которые могут служить своеобразными индикаторами необходимости модификации систем, можно перечислить следующие (на примере экономической ИС):

- изменения в системе документооборота, формате и структуре формируемых документов;
- изменения в перечне задач, принадлежащих к основной функциональности системы, или в методах их решения;
- мнения пользователей о работе с системой, качестве обработки данных и результирующей информации;
- информация специализированного ПО (например, в составе СУБД и ОС) по статистике работы системы, ее быстродействию и отказоустойчивости;
- характеристики организации и внешней среды (выпуск новых изделий, появление новых технологических ограничений);
- изменение законодательства и требований регуляторов (например, в части стандартов учета);
- уход из компании и недостаток на рынке специалистов со знаниями, необходимыми для поддержки конкретной *legacy*-системы;

- слишком высокая стоимость поддержки системы или совокупная стоимость владения;
- отсутствие технических возможностей (высокая сложность) интеграции с новыми, внедряемыми на предприятии системами;
- рыночная конкуренция и бизнес-приоритеты (например, интернет-магазин, который фокусируется на удовлетворении потребностей пользователей, может принять решение о модернизации пользовательского интерфейса и системы принятия и обработки заказов клиентов);
- осуществляемые процессы слияний и поглощений M&A (так, в случае объединения информационных систем с другой компанией требуется полный пересмотр существующего ландшафта систем и частичная или полная модернизация).

Сами действия по модификации и совершенствованию системы могут представлять собой: трансформацию исходного кода вплоть до смены среды или языка программирования, совершенствование архитектуры приложения и принципов взаимодействия его модулей на основе анализа качества кода. Чаще всего выделяются следующие варианты проведения модернизации.

Миграция (*migration*). Переход на более новую версию платформы, ОС, СУБД или языка программирования. Обеспечивается достаточно эффективный с точки зрения соотношения стоимости и качества способ трансформации устаревающих систем.

Пример

В процессе миграции могут изменяться:

- платформа: миграция с IBM Mainframe COBOL/CICS/DB2/VSAM на Intel-платформу Windows Micro Focus COBOL (с БД Oracle);
- язык программирования: конвертация кода на C#;
- пользовательский интерфейс: переход от desktop-версии к веб-интерфейсу;
- аппаратное обеспечение: переход с DEC VAX-обеспечения на Intel-серверы и рабочие станции (что, в свою очередь, требует перевода ПО с VAX OpenVMS на MS Windows);
- миграция баз данных: переход с IDMS на Oracle.

Реинжиниринг (*re-engineering*). Чаще всего применяется при адаптации сервисно-ориентированной архитектуры SOA, при построении прежних приложений на основе новых технологий и платформ, с прежней или же расширенной функциональностью.

Рефакторинг кода. Преобразования/реорганизация кода. Рефакторинг может быть обоснован необходимостью реализации новой функции, которая не соответствует текущему архитектурному решению. Однако в особенности подобные активности необходимы, когда логика программы слишком сложна для понимания и требует структуризации и совершенствования в целом.

Пример

Дублирование кода, слишком длинный список параметров метода или код самого метода, «мертвый код», лишние переменные, несгруппированные данные и пр.

Для устранения таких недостатков кода используются различные методы, например, инкапсуляция полей, делегирование, замена операторов полиморфизмом и пр.

Следует отметить, что несмотря на то, что рефакторинг относится к одному из вариантов модернизации, он должен проводиться на протяжении всего цикла разработки (к примеру, именно такая логика лежит в основе концепции экстремального программирования и ряда других методологий).

Смена хостинга (*re-hosting*). Чаще всего проводится в качестве промежуточного шага при предстоящей замене аппаратного обеспечения (например, на UNIX/Wintel-платформах).

Внедрение «коробочного» решения (*package implementation*). Замена устаревшего ПО целиком или частично на целые семейства решений (SAP, Oracle Apps).

Виртуализация как стратегия модернизации решений. Одним из возможных решений при модернизации ИС является *виртуализация*.

Виртуализация — предоставление вычислительных ресурсов, абстрагированное от аппаратной реализации и изолирующее вычислительные процессы конкретных физических ресурсов. Таким образом, при виртуализации создаются различные уровни абстракции.

Сама концепция виртуализации появилась еще в 1970-х гг., когда IBM виртуализировала интерфейсы своего оборудования. VMS запускается непосредственно на основном оборудовании, позволяющем создавать множество виртуальных машин (VM), каждая из которых может обладать своей собственной операционной системой. Другим вариантом виртуализации в то время была симуляция процессора (выполнение псевдокода на виртуальной машине вместо реального оборудования).

В настоящее время виртуализация не теряет своей актуальности и становится все более и более популярным решением в силу следующих причин:

- происходит оптимизация расходов на ИТ;
- увеличивается степень контроля над ИТ-инфраструктурой;
- сокращаются плановые и внеплановые простои.

Среди вычислительных ресурсов, для которых доступна виртуализация, операционные системы и сети передачи данных, программно-аппаратные платформы, серверы/системы хранения данных. Один из наиболее сложных видов виртуализации обеспечивается эмуляцией аппаратных средств (когда виртуальные машины аппаратных средств создаются на хост-системе, чтобы эмулировать интересующее оборудование) (рис. 3.3).

Рис. 3.3. **Эмуляция аппаратных средств**

Виртуализация приложений обеспечивает изоляцию и безопасность ресурсов, которые предоставляются приложениям контейнерами. Каждый контейнер обслуживает содержащиеся в нем приложения при определенном уровне рабочей нагрузки. В некотором роде каждый контейнер имеет сходство с физическим сервером, однако виртуализация происходит на уровне приложений, а не на аппаратном.

При этом создается возможность выполнять приложение в виртуальном пространстве, которым можно управлять как отдельной единицей. Приложения могут выполняться в потоковом режиме на сервере приложений или на клиентском устройстве. Поэтому виртуализация отлично подходит для компаний, обладающих широким спектром приложений с относительно небольшим потреблением ресурсов центра данных, но используемых большим количеством сотрудников.

Виртуализация серверов маскирует от рядовых пользователей детали использования ресурсов (количество и основные данные серверов, процессоров, операционных систем), с которыми ведется работа. Она призвана обеспечить централизованное управление всеми серверными мощностями с максимально возможной гибкостью и высокой степенью масштабирования, а также быстрое развертывание виртуальных машин из готовых шаблонов, оперативное восстановление работоспособности серверов, мониторинг текущей загрузки физических и виртуальных серверов.

Одновременно с этим снижается зависимость эффективности сервиса от старения оборудования, так как появляется возможность осуществлять модернизацию практически без прерывания сервиса, что критично для крупных территориально распределенных компаний.

Крайне ярким примером виртуализации серверов является услуга *VPS-хостинга* (*виртуальный выделенный сервер — virtual private server*, VPS). В подобных случаях в сети могут создаваться несколько независимых друг от друга виртуальных серверов с собственным набором служб и уникальными характеристиками, которые должны существовать как независимые узлы сети.

Виртуализация систем хранения данных с точки зрения пользователя переносит данные с разрозненных сетевых устройств хранения на единое, управляемое централизованно.

Основной задачей в данном случае является повышение эффективности и надежности, гибкости хранения данных и их мобильности. Дополнительное отделение систем хранения от серверного оборудования уровнем виртуализации позволяет прозрачно перемещать данные между системами хранения и упростить процессы обслуживания и поддержки.

Виртуализация представлений — предоставление вычислительных ресурсов терминальным сервером и выполнение всех операций клиентских приложений на нем. Несколько компьютеров могут быть представлены как один отдельный компьютер (серверный кластер/*grid computing*). Пользователь при этом лишь видит собственное представление, что значительно снижает сложность администрирования, так как к подобной терминальной сессии всегда можно подключиться, избегая установки дополнительных программных средств. К тому же, несмотря на высокую стоимость подобных мощных серверов, итоговая стоимость проекта может оказаться более низкой, нежели при установке многочисленных локальных рабочих станций.

Особенности проектов по модернизации. Бóльшая часть задач по модернизации решения может проводиться параллельно с эксплуатацией (не дожидаясь ее прекращения), однако это требует повышенного контроля всех параметров и данных системы, вероятность ошибок в которых резко повышается. С другой стороны, успешное применение подобного способа позволяет избежать необходимости миграции данных после модернизации и поиска «промежуточного» решения на время модификации ИС, заменяющего по функциональности и удобству основную систему.

Перед началом модернизации следует рассмотреть ряд аспектов.

Целесообразность модернизации. Иногда лучшим выбором может быть продолжение использования системы с фокусом на снижении затрат на поддержку. Однако любая *legacy*-система со временем будет требовать все бóльших вложений за счет необходимости ее сопровождения сотрудниками с необходимым набором компетенций и опытом работы с подобными системами.

Аутсорсинг систем и процессов. Все большее число компаний выбирает вариант рехостинга *legacy*-систем (особенно в условиях наличия многочисленных вариантов планов аутсорсинга, предлагаемых поставщиками). При этом модификации самой системы можно избежать.

Целесообразность перехода на другую платформу. Для компаний, активно участвующих в процессах слияний и поглощений (M&A), эффективной альтернативой модернизации может стать консолидация и перевод *legacy*-систем на единую корпоративную платформу (в идеале — с хорошей масштабируемостью). Подобные действия могут значительно повысить затраты в краткосрочном периоде, но долгосрочные выгоды оказываются очень существенными.

Минимизация числа изменений. В некоторых случаях предприятия могут вносить изменения во внутренний дизайн приложения, сохраняя его функциональные возможности. Так, рефакторинг кода программного решения иногда позволяет модифицировать приложение гораздо более эффективно, нежели масштабная модернизация.

Минимальная кастомизация. Многие вендоры положительно относятся к проектам по кастомизации/локализации их программных решений, когда это означает более высокую выручку и выход на новые рынки. Так, в большинстве случаев компании приобретают бизнес-платформы и адаптируют исходный код под собственную специфику (либо выпускают патч для более старой версии, например, с учетом изменений налогового законодательства).

Однако в тот самый момент, когда предприятие примет решение о выпуске новых релизов, ему придется столкнуться с множеством кастомизированных доработок и отклонений от исходной версии, что увеличит затраты труда и времени на модификацию.

Длительные проекты внедрения значительно повышают риски. В современных условиях, когда жизненный цикл программных решений постоянно сокращается, а новые технологии и стандарты появляются с завидной регулярностью, система, разработанная в сложном длительном проекте внедрения с многочисленными итерациями тестирования и доработок, может оказаться неактуальной еще до окончания проекта.

Интеграция систем на основе SOA не является панацеей от всех проблем. Все чаще вендоры рекомендуют настройку SOA-интерфейсов *legacy*-систем, когда «монолитные» приложения разбиваются на определенные компоненты, которые затем по отдельности реализуются по SOA-принципам. Адаптеры инкапсулируют различные технологии для реализации интерфейса между приложениями, а для подключения используется единая сервисная шина.

Однако следует отметить, что данный подход никоим образом не снижает сложность приложения и затраты на его сопровождение, а лишь обеспечивает его более эффективное взаимодействие с окружающей средой.

Возможности обновлений. Необходимо взвешенно принимать решение об обновлении ПО: обновлять его при выпуске каждой новой версии, либо же с определенными промежутками, только после выпуска самых значительных обновлений. Компании, заключающие контракт на сопровождение, чаще всего ограничиваются обновлением приложений до новой версии в случае необходимости. Однако возможна ситуация, когда приобретенное приложение больше не выпускается и не поддерживается и организациям приходится сталкиваться с масштабной заменой программного решения.

Планирование архитектуры приложений. Замена системы может решить только небольшую часть проблем, важно в целом эффективно планировать ландшафт приложений, формировать принципы принятия решения о внедрении, модернизации и утилизации систем, согласовывать действия со стратегическими целями ИТ и предприятия в целом.

3.7. Утилизация

Не существует вечных систем, а значит, необходимо предусмотреть условия, при которых понадобится отказаться от системы и вывести ее из эксплуатации. Эта деятельность, в свою очередь, также является проектом, ключевая цель которого — снятие системы (или ее отдельных модулей) с эксплуатации, что чаще всего проводится уже после внедрения новой системы. Рассмотрим классификацию необходимых в рамках данного проекта действий[1]. Обратите внимание, что деятельность по выбору альтернативной ИС и принятие решения по замене не входят в эту фазу. Фаза утилизации предусматривает только практические действия по выводу ИС из эксплуатации, так как при формировании ТСО затраты на анализ альтернатив и выбор новой системы будут отнесены на новую систему. Допустим, на предприятии принято решение о внедрении системы CRM и выводе из эксплуатации «самописной» программы, которая поддерживала учет выставленных коммерческих предложений заказчикам. Функциональность новой системы CRM будет шире. Конечно, бухгалтерия отнесет затраты по обследованию, постановке задачи, процессу выбора ПО, доработке и внедрению на новый программный продукт.

Технические аспекты

Аппаратное обеспечение. Существующую платформу (компьютерное и серверное оборудование, сетевые и телекоммуникационные устройства, параметры сети и безопасности, например, выделенные IP-адреса и настройки сетевого экрана) можно использовать для развертывания других систем (например, изменив некоторые параметры или конфигурацию), продать, расторгнуть контракт лизинга.

Программное обеспечение. Основные программное решение, компиляторы и среды разработки, коннекторы, внутренние библиотеки и прочие компоненты ПО, относящиеся к системе, можно сохранить в качестве резервной копии.

Данные. Для данных (как необходимых для работы с системой, так и созданных в процессе ее эксплуатации) необходимо создать резервные копии как в «родном» формате системы, так и в читаемых на других платформах форматах (pdf, xml, ...).

Документация. Бизнес- и техническая документация системы (в виде детального описания алгоритмов и бизнес-правил, заложенных в систему) также должны быть сохранены в резервных копиях внутреннего формата системы и общедоступных (pdf, docx, ...).

Замещающие системы. К моменту вывода системы из эксплуатации должны быть предусмотрены все средства плавного перехода к эксплуатации другого решения (включая аспекты управления изменениями).

Технические аспекты/2

Пример

При смене системы «Учет кадров компании “Парус”», которая ранее эксплуатировалась в компании, на систему 1С решение о снятии с эксплуатации первой системы (утилизации этой подсистемы) принимается только после полной миграции данных, одновременного сопровождения двух систем и полномасштабного ввода соответствующего модуля 1С в эксплуатацию.

Зависимые/интегрированные системы. Этот пункт предполагает два основных аспекта. Первый — вопрос прекращения передачи и получения данных в интегрированные системы и из них, а также необходимость предусмотреть другие источники данных для них. Особенно это касается внешних приложений и интерфейсов работы с системами контрагентов. Второй — вопрос определения стратегии действий для вспомогательных систем (БД, средства отчетности, приложения автоматизации бизнес-процессов и пр.), необходимость в которых со снятием системы с эксплуатации может исчезнуть или потерять актуальность.

Организационные аспекты

Сотрудники ИТ-департамента компании. К моменту прекращения необходимости поддержки эксплуатации и модернизации системы ИТ-специалистами необходимо предусмотреть их занятость в других проектах с учетом оптимального применения приобретенных ими за время проекта знаний и умений

Подрядчики и специалисты, нанятые по контракту. Внешние специалисты, на постоянной или временной основе занятые в проекте внедрения и поддержки системы, могут остаться в компании (продлив контракт или перейдя в штат) или покинуть ее по окончании контракта. Соответственно,

в случае необходимости использовать их компетенции на других проектах необходимо своевременно предусмотреть им замену и организовать передачу критически важных знаний новым сотрудникам.

Конечные пользователи. Если на смену утилизируемому ПО внедряется альтернативное решение, пользователи перед этим в обязательном порядке должны пройти необходимое обучение.

Юридические аспекты

Контрактные вопросы. В процессе эксплуатации системы могли заключаться договоры с подрядчиками по поддержке, сервисами веб-хостинга, дата-центрами и прочими организациями, а значит, каждый из контрагентов должен быть своевременно оповещен о планируемых изменениях и принято решение о продолжении или прекращении совместной работы.

Лицензирование. Закупленные лицензии на ПО могут быть дезактивированы либо «заморожены» (в этом случае право на владение лицензиями сохраняется, но право пользования ими временно утрачивается до заключения дальнейших соглашений с поставщиком (например, при обмене лицензий на другую версию или другой программный продукт)).

Отчетность. Любые обязательства компании по раскрытию информации, ранее осуществлявшиеся при помощи выводимой из эксплуатации системы, должны быть предусмотрены в заменяющей ИС (включая передачу исторических данных).

Юридические аспекты/2

Пример

В качестве еще одного напоминания, каких ситуаций следует избегать, приведем следующий пример.

В одной из организаций используемое решение (по учету транспорта) было разработано 5—10 лет назад, и главные разработчики давно перешли на работу в другую компанию. Тем временем, по мере роста масштабов бизнеса, понадобилось внедрение приложения с расширенной функциональностью, отвечающего новым потребностям бизнеса. Однако миграция данных оказалась невозможной в силу отсутствия открытой для пользователя функции экспорта и недоступности аккаунта администратора системы, как и документации, включая исходный код со спецификациями, единственные версии которых были только у разработчиков системы.

Контрольные вопросы и задания

1. Какие виды действий требуется совершить на фазе планирования проекта?
2. Что содержат в себе такие документы, как отчет об экспресс-обследовании, технико-экономическое обоснование и оценка целесообразности проекта?
3. Какая деятельность происходит на стадии анализа и постановки задачи?
4. Что понимается под информационным обследованием предприятия?
5. При помощи каких нотаций и программных продуктов осуществляется моделирование бизнес-процессов?
6. На основании каких стандартов производится классификация требований к ИС?
7. Какие аспекты включает в себя фаза проектирования ИС?
8. Что происходит на стадии разработки информационной системы?
9. Какие действия совершаются при настройке конфигурации, создании ролей пользователей, миграции данных и разработке контрольного примера?
10. Какие цели преследует проведение тестовой эксплуатации?

11. Как осуществляется развертывание и внедрение информационной системы?
12. Почему особую важность приобретает обучение пользователей?
13. Какие основные виды тестирований существуют?
14. Как производятся приемно-сдаточные испытания информационной системы?
15. В чем заключается важность фазы эксплуатации ИС?
16. Какие виды сопровождения эксплуатации существуют и чем они различаются между собой?
17. Зачем проводится модернизация информационной системы?
18. Какие существуют стратегии управления *legacy*-системами?
19. На чем основывается концепция виртуализации и как она применяется на фазе модернизации ИС?
20. Какие аспекты фазы утилизации вы отметите? Какими причинами вызвана потребность в данной фазе?

Спасибо за внимание