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

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.

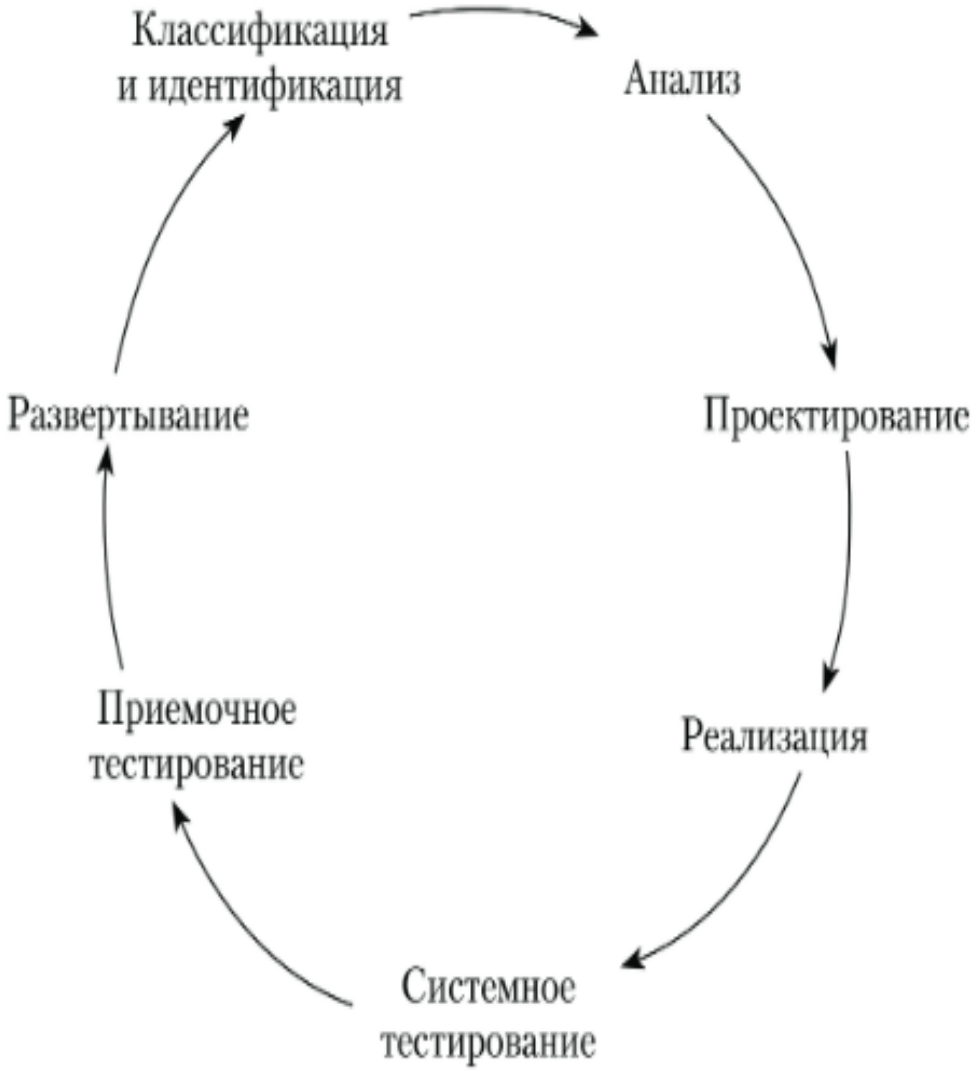
Техническая поддержка. Техническая поддержка системы начинается после приема в промышленную эксплуатацию и может продолжаться до снятия с эксплуатации и утилизации системы. Как правило, варианты осуществляемой поддержки варьируются по процентам от стоимости приобретенного ПО в зависимости от набора оказываемых 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 A 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).

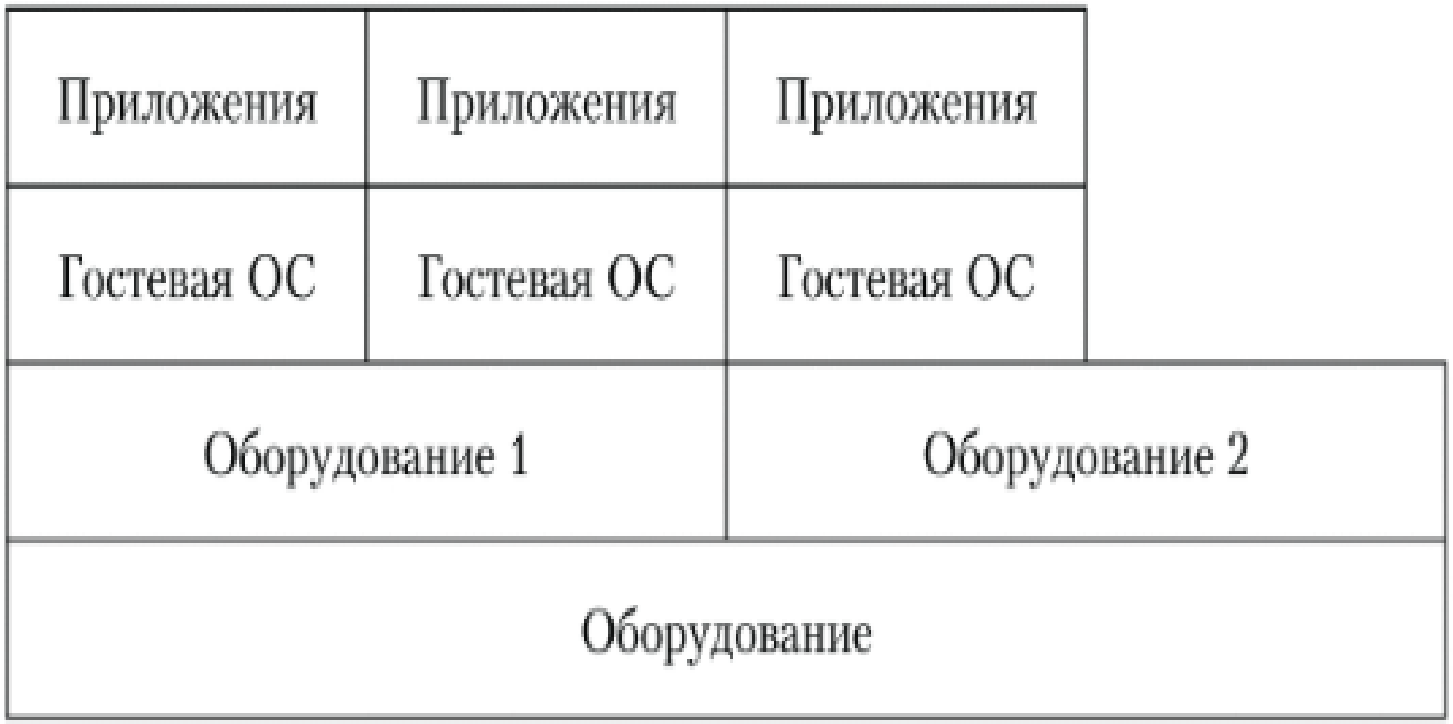


Рис. 3.3. **Эмуляция аппаратных средств**

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