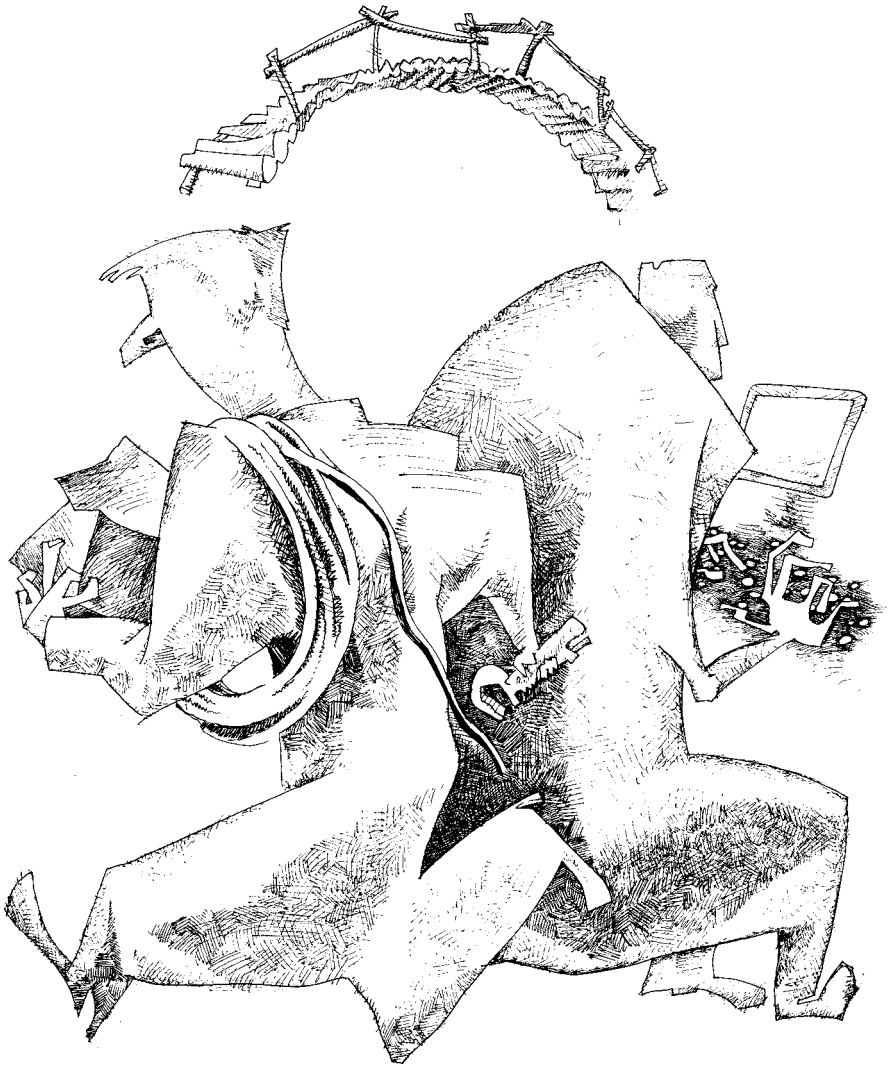
Труднее всего в жизни запомнить, какой мост надо перейти, а какой надо сжечь.
Бертран Рассел

Совместно с itSMF
СООБЩЕСТВО ПРОФЕССИОНАЛОВ ITSM



Наводим мосты между разработкой и эксплуатацией

Впервые опубликована в «Альманахе itSMF России 2013»

Несмотря на обилие материалов и по разработке, и по эксплуатации информационных систем, организация эффективного взаимодействия подразделений разработки и эксплуатации является одним из наиболее актуальных вопросов для ИТ-руководителей. О типичных проблемах в этой области и о практике их решения мы и поговорим в данной статье.

Дмитрий Исайченко

Директор по консалтингу компании Cleverics. Имеет опыт обследования и реорганизации работы подразделений ИТ ряда средних и крупных компаний: BSGV, ВТБ24, Raiffeisenbank, Банк «Санкт-Петербург», Внешэкономбанк, Метроком, М. видео, СУАЛ и других. Основная специализация — проведение консалтинговых проектов в сфере управления ИТ, развитие методики оказания консалтинговых услуг, программные средства автоматизации процессов управления ИТ. Член совета экспертов itSMF России. Ведёт свой блог на портале Real ITSM.

Предмет анализа

Начнем с определения организационного контекста, который поможет нам более четко сформулировать задачу. Для этого представим обычную ИТ-организацию, имеющую функциональную организационную структуру и применяющую не только готовые коммерческие продукты, но также активно использующую внутреннюю разработку. В этом случае в данной организации можно выделить два крупных функциональных блока – департаменты разработки и эксплуатации (см. рисунок).

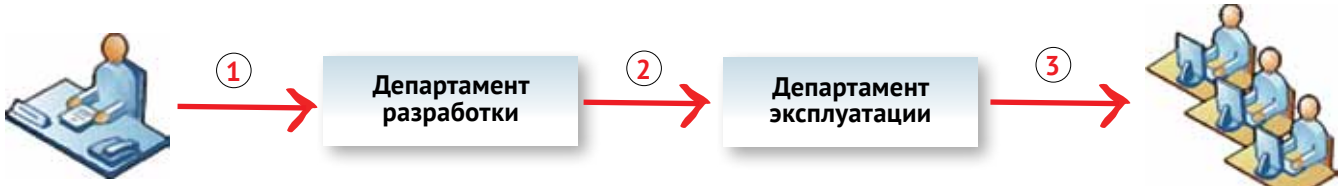
Департамент разработки отвечает за развитие информационных систем и обрабатывает запросы на разработку от бизнес-руководителей или выделенных сотрудников[1]. В своей деятельности департамент разработки в основном руководствуется подходом ALM (Application Lifecycle Management) и, возможно, даже использует соответствующее специализированное программное обеспечение для управления требованиями, распределения задач на разработку, определения содержания релизов, организации тестирования, отслеживания дефектов и прочего.

В свою очередь, департамент эксплуатации принимает от разработчиков новые релизы программного обеспечения (возможно, с участием выделенной функции тестирования, что для нас не принципиально), развертывает их в продуктивную среду и обеспечивает эксплуатацию информационных систем, включая администрирование, резервное копирования, мониторинг, поддержку пользователей, выполнение регламентных операций и многое другое. В своей деятельности департамент эксплуатации обычно в той или иной степени руководствуется концепцией ITSM, хотя, возможно, только в пределах базовых операционных процессов – управления инцидентами и запросами пользователей, в какой-то части управления конфигурациями, изменениями.

Два этих департамента управляются разными руководителями, используют для организации своей работы разные подходы и инструменты, по-разному взаимодействуют с бизнес-подразделениями[2]. Причем разработчики, как правило, выполняют запрос на доработку и переходят к следующему, передавая результаты своей работы сотрудникам департамента эксплуатации, которым с этими разработками предстоит жить долгие годы.

[1]Обычно департамент разработки также выполняет функции управления проектами, закупками и архитектурой, вовлеченные в проекты внедрения новых информационных систем, но в данной статье мы без этих функций можем обойтись.

[2]Отдельным интересным вопросом является применимость к представленной ситуации и организационные границы процесса управления уровнем ИТ-услуг, в частности, включение в сервисные отношения департамента разработки. Подробнее об этом можно почитать в статье «Управление уровнем ИТ-услуг. Часть 2. Каталог ИТ-услуг и процессы» Альманаха itSMF России 2013.



Бизнес-заказчик

1 Запрос на разработку новых функциональных возможностей
2 Передача решения в эксплуатацию
3 Предоставление услуг на базе разработанного решения

Пользователи

Рис. Разработка и эксплуатация в функциональной оргструктуре.

Взаимодействие подразделений разработки и эксплуатации обычно включает в себя:

- развертывание разработок в продуктивную среду;
- поддержку пользователей;
- устранение ошибок в программном обеспечении.

Как правило, для такой организации в большей или меньшей степени характерны следующие проблемы во взаимодействии подразделений разработки и эксплуатации:

- непредсказуемый для департамента эксплуатации поток работ по тестированию и развертыванию доработок (при этом, даже если тестирование выполняется выделенной функцией, департамент эксплуатации обычно принимает в нем участие, а также обеспечивает подготовку тестовых сред);
- заметное увеличение количества запросов пользователей после развертывания очередного релиза, особенно в случае развертывания на крупной бизнес-сети;
- отсутствие информации по изменениям в составе релиза, что затрудняет поддержку пользователей;
- недостаточно оперативная поддержка со стороны разработчиков при выявлении ошибок в программном обеспечении;
- несоответствие новых решений требуемому режиму эксплуатации.

> Для подразделения эксплуатации готовность очередной доработки является неожиданностью, плановые действия разработчиков, выливаются в плохо предсказуемую загрузку подразделений эксплуатации

Разберемся с этими проблемами более подробно. При этом будем отличать два типа изменений в информационных системах – крупные и небольшие.

1. **Крупные изменения** представляют собой внедрение новых информационных систем либо новых версий информационных систем. Обычно на такие изменения уходит от трех месяцев до нескольких лет, требуют значительных ресурсов, и поэтому к ним применяется практика управления проектами.
2. **Небольшие изменения** связаны с реализацией текущих доработок. Масштаб этих изменений относительно невелик, время реализации колеблется от одного дня до трех месяцев, зато число таких изменений превышает количество проектов в несколько раз. Например, в крупной организации может выполняться параллельно несколько десятков проектов, при этом количество доработок достигает одной-двух сотен в месяц (а иногда и больше).

Очевидно, что к крупным и небольшим изменениям целесообразно применять разные практики управления. Поэтому и трудности при их реализации, в том числе во взаимодействии разработки и эксплуатации, отличаются.

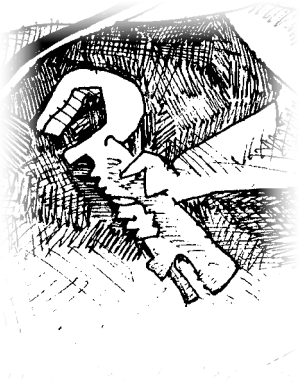
Причины проблем и решения

Непредсказуемый поток работ по тестированию и развертыванию разработок

Эта проблема более характерна для небольших доработок, поскольку проекты

планируются заранее, и вовлечение в них подразделений эксплуатации известно на несколько месяцев вперед.

Причина. Возникает проблема потому, что запросы на новую функциональность поступают напрямую разработчикам, и работы по реализации планируются ими самостоятельно. Поэтому для подразделения эксплуатации готовность очередной доработки является неожиданностью. А это означает: неожиданные работы по подготовке тестовых сред, проведению тестирования, развертыванию изменений. То есть действия, плановые для разработчиков, для подразделений эксплуатации выливаются в плохо предсказуемую загрузку. Это приводит к задержкам при развертывании изменений (время развертывания может колебаться от одного дня до двух и более недель) и нарушениям в выполнении плановых работ блока эксплуатации.



Решение. Рекомендации в данном случае зависят от объема разработок.

1. При небольшом потоке разработок эти вопросы могут решаться публикацией планов по разработке на доступном департаменту эксплуатации ресурсе (например, ряду сотрудников предоставляется доступ на чтение к ALM-системе или календарь разработки из ALM-системы автоматически публикуется на внутреннем web-портале). Возможно также проведение регулярных совещаний с обзором текущих планов по разработке.
2. Однако при большом количестве задач запланированные сроки начинают слишком часто меняться – как из-за поступления новых, более приоритетных, запросов, так и за счет внутреннего перепланирования работ, вызванного сложностями при реализации. Да и самих разработок становится так много, что работы по тестированию и развертыванию превращаются в сплошной поток.

Поэтому при большом количестве запросов на разработку целесообразно введение практики управления релизами, что приводит к появлению заранее известного календаря работ. Помимо решения вопросов планирования работ в блоке эксплуатации, эта практика также позволяет обеспечить более предсказуемое информирование об изменениях конечных пользователей.

> Обычно внутренняя политика организации считает приоритетной задачей поддержку пользователей, однако никто при этом не подразумевает, что изменение планов разработки из-за необходимости решения инцидентов — нормальная практика

Оборотной стороной может стать увеличение среднего времени реализации изменений, однако этого можно избежать при правильном выборе частоты выпуска релизов. Так, при использовании agile-методов с коротким циклом разработки частота выпуска релизов может быть равна двум неделям и даже меньше[3]. Дополнительно время реализации изменений можно сократить, обеспечив заблаговременную подготовку тестовых сред, которые к моменту окончания сборки релизов уже полностью готовы к проведению тестирования.

Увеличение количества запросов пользователей после развертывания релиза

Следующей сложностью, характерной как для крупных, так и для небольших изменений, сгруппированных в релиз, является увеличение количества запросов в службу технической поддержки после развертывания в продуктивную среду.

Причина. При проектной организации работ, как правило, проводится обучение персонала, а переводу в продуктивную среду предшествует период опытной эксплуатации. В то же время при реализации небольших доработок ни обучения, ни опытной эксплуатации не выполняется, но изменений, включенных в релиз, может накопиться достаточно, чтобы удивить и пользователей, и службу технической поддержки. Особенно остро эта проблема проявляется при развертывании разработок на крупной бизнес-сети, поскольку одни и те же вопросы и ошибки начинают множиться пропорционально размеру сети (количеству магазинов, отделений банка,..).

[3]Некоторые наши заказчики практиковали выпуск плановых релизов по основным корпоративным системам один-два раза в неделю.

Решение. Здесь практика позволяет дать несколько рекомендаций.

- Во-первых, как мы и говорили выше, важно выбрать оптимальную частоту выпуска релизов. Увеличение частоты приводит не только к снижению среднего времени реализации изменений, но и к сокращению среднего количества изменений в одном релизе. А чем меньше релиз, тем меньше ошибок, ниже вероятность пропустить их при тестировании, быстрее проводится диагностика возникающих инцидентов. Напротив – при развертывании больших релизов (например, выпускаемых раз в квартал и включающих в себя несколько десятков доработок) организацию может «лихорадить» еще больше месяца;
- Во-вторых, в крупной сети разумно ввести практику поэтапного развертывания релизов. Как правило, выбирается и закрепляется перечень объектов сети, которые первыми принимают на себя новшества. В сетях, охватывающих несколько часовых поясов, рекомендуется включать в состав тестовых объектов восточные отделения, поскольку там раньше начинается рабочий день и в среднем ниже уровень нагрузки, чем в западных регионах. Однако это предполагает наличие дежурной смены инженеров и поддержку, готовую помочь пользователям, когда в Москве еще глубокая ночь.

Отсутствие информации по изменениям в составе релиза

Отсутствие информации по изменениям в составе релиза является проблемой и для эксплуатирующего подразделения, и для конечных пользователей. Причины такой проблемы очевидны.

Решение. При регулярном (и достаточно частом) выпуске релизов информация может быть представлена небольшим файлом описания релиза (release notes), который специалистам по поддержке посильно осознать за 15–20 минут. Еще удобнее, если это централизованный ресурс на web-портале, который позволяет видеть историю изменений по релизам и быстро корректируется в едином источнике при обнаружении ошибок в описании. А наличие ссылок, позволяющих быстро перейти от новой функциональной возможности к требованиям по ее реализации позволяет дополнительно сократить долю инцидентов, передаваемых на решение разработчикам. При этом во многих случаях эта информация может извлекаться из ALM-системы и публиковаться на доступный web-ресурс автоматически, не требуя дополнительных трудозатрат на обновление.

Важно выбрать оптимальную частоту выпуска релизов. Увеличение частоты приводит к сокращению среднего количества изменений в одном релизе, а значит, к уменьшению ошибок и ускорению диагностики возникающих инцидентов

Недостаточно оперативная поддержка со стороны разработчиков

Причина. В большинстве организаций, с которыми мы работали, средняя длина очереди на разработку была равна нескольким месяцам (довольно часто – от полугода до года). Поэтому разработчиков всегда торопит план реализации очередных запросов. Отвлекаться на решение инцидентов – значит сокращать ресурсы на выполнение плановых работ. А характер работы (необходимость концентрации на одной решаемой задаче) мешает быстрому переключению на вопросы поддержки пользователей. Однако при обнаружении инцидентов, решить которые самостоятельно эксплуатирующее подразделение не в состоянии, оперативное привлечение разработчиков необходимо.

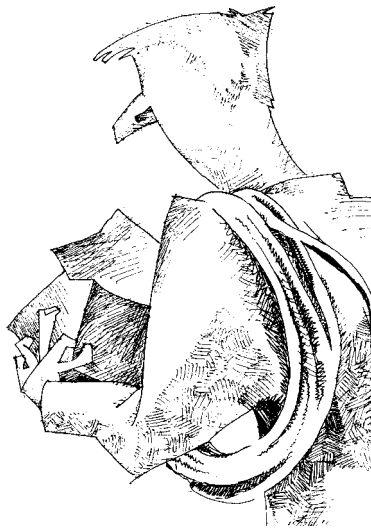
Обычно внутренняя политика организации в той или иной форме считает приоритетной задачей поддержку пользователей (если есть инцидент, мешающий работе, он должен быть устранен как можно быстрее). Однако никто при этом не подразумевает, что изменение планов разработки из-за необходимости решения инцидентов – нормальная практика.

Решение. Для решения ресурсного конфликта между разработкой и поддержкой в ряде случаев фиксируется максимальная доля времени разработчиков, выделяемого на поддержку, обычно на уровне 20–30 %. При этом зафиксированная величина на самом деле не является жестким пределом сверху, а скорее дает некоторый ориентир для планирования работ. Это действительно обеспечивает относительно оперативное решение инцидентов, но этого недостаточно.

Дело в том, что в ряде случаев можно повысить надежность информационных систем и сократить трудозатраты на их эксплуатацию оптимизацией технических решений (например, реализацией удобных средств администрирования учетных записей и прав), а также устранением проблем, которые приводят к повторению инцидентов (например, оптимизации запросов к базам данных для повышения быстродействия). Но когда департамент эксплуатации инициирует соответствующие запросы, они попадают в общую очередь разработки, где и живут годами, постоянно вытесняемые более приоритетными запросами бизнеса. То есть привлечь ресурс разработчиков к решению вопросов эксплуатации за рамками конкретного инцидента бывает почти невозможно. При этом разработчики, как правило, никак не мотивированы на решение этой проблемы.

Здесь можно попробовать зафиксировать минимально допустимую долю времени, выделяемого разработчиками на развитие по запросам эксплуатирующего подразделения и ввести контроль фактического использования этого времени. Тогда получим, например, те же 30 % на вопросы эксплуатации, из них 20 % – на решение инцидентов и 10 % – на развитие по запросам департамента эксплуатации. Контроль расходования времени должен обеспечить фактическое выделение этих 10 %, если только есть соответствующие запросы. При этом определять приоритеты запросов на изменения, поступающих от эксплуатации, может комитет по управлению ИТ-услугами, что позволяет установить более четкую связь инициатив по оптимизации с качеством и стоимостью услуг.

В некоторых организациях дополнительной мерой может стать оплата разработки по задачам оптимизации в интересах подразделения эксплуатации из своего бюджета, то есть фактически включение коммерческих сервисных отношений между разработкой и эксплуатацией. Это стимулирует разработчиков брать задачи по эксплуатации в работу и выполнять их, а не просто отчитываться о выделенном времени. В свою очередь, подразделение эксплуатации не будет запрашивать работы, не дающие соизмеримого оплате эффекта.



Несоответствие новых решений требуемому режиму эксплуатации

Эта проблема более характерна для новых решений, как правило, реализуемых в рамках проектов.

Причина. Она проявляется в том, что при проектировании и разработке не учтены:

Таблица.
Проблемы взаимодействия департаментов разработки и эксплуатации и варианты решений

Проблема	Возможные решения
Непредсказуемый поток работ по тестированию и развертыванию разработок	Введение календаря выпуска релизов для систем с плотным потоком доработок
Увеличение количества запросов пользователей после развертывания релиза	Ограничение размера релиза, повышение частоты выпуска релизов, поэтапное развертывание релизов
Отсутствие информации по изменениям в составе релиза	Публикация описаний релизов, ссылки на требования к реализации функциональных возможностей в составе релиза посредством интеграции с ALM-системой
Недостаточно оперативная поддержка со стороны разработчиков	Выделение фиксированного времени разработчиков на решение вопросов эксплуатации, контроль фактического выделенного времени, оплата разработок, направленных на оптимизацию эксплуатации
Несоответствие новых решений требуемому режиму эксплуатации	Фиксация состава эксплуатационных требований в проектных документах, разработка технологического стандарта, вовлечение эксплуатирующих подразделений с целью контроля выявления и реализации эксплуатационных требований, а также соблюдения технологических стандартов

- эксплуатационные требования, такие, как обеспечение требуемых характеристик быстродействия при одновременной работе заданного числа пользователей, требования к времени завершения критичных операций (cut-off time) и так далее. Реализация многих эксплуатационных требований, если они не были известны на этапе проектирования решения, после окончания проекта может быть почти невозможна;
- текущая практика эксплуатации (включая используемые технологии разработки, компетенцию специалистов, практику мониторинга, ...) и присущие технические ограничения (например, ограничения каналов передачи данных, быстродействия рабочих станций, разрешения мониторов на рабочих местах, ...).

Рекомендуется организовать вовлечение подразделений эксплуатации в согласование архитектурных и технических решений по проектам, что гарантирует своевременное выявление и учет эксплуатационных требований, а также соответствие утвержденным технологическим стандартам

В результате решение может эксплуатироваться только с существенными отклонениями от принятых практик. Это приводит к конфликтам при принятии решений в эксплуатацию, а также многократно обостряет проблему привлечения подразделений разработки к решению вопросов эксплуатации после окончания проекта, рассмотренную нами в предыдущем пункте.

Решение. В такой ситуации можно рекомендовать:

- определить фиксированный состав эксплуатационных требований, которые должны быть учтены при создании новых решений, и включить эти требования в шаблоны проектных документов;
- разработать технологический стандарт, содержащий базовые архитектурные и технические требования для обеспечения соответствия практике эксплуатации и ограничениям среды. В частности, требования к возможностям мониторинга и резервного копирования, ведению журналов, процедурам настройки рабочих станций пользователей, возможностям персональных компьютеров, средствам интеграции приложений и другие значимые для вас требования;
- организовать вовлечение подразделений эксплуатации в согласование архитектурных и технических решений по проектам, что гарантирует своевременное выявление и учет эксплуатационных требований, а также соответствие утвержденным технологическим стандартам.

Выполнение перечисленных рекомендаций позволит получить решения, в существенно большей степени готовые к эксплуатации с учетом особенностей конкретной организации.

Подводим итоги

В заключение суммируем перечисленные нами проблемы и варианты решений (см. таблицу). Практика многих компаний показывает, что, следуя перечисленным рекомендациям, можно если не полностью решить, то по крайней мере в существенной степени ослабить проблемы, связанные с взаимодействием подразделений разработки и эксплуатации.