15. УПРАВЛЕНИЕ ПРОБЛЕМАМИ – ЕЩЕ ПРОАКТИВНЕЕ

В публикациях ITEMS уже освещалось возможное деление задач и акцентов управления проблемами на реактивные и проактивные (выпуск от 6 октября 2008 года). Описанное там разделение предполагает, что «разница между реактивным и проактивным управлением проблемами видна из названия. Цель первого – реагировать на инциденты и не допускать их повторения. Цель второго – предотвращать их возникновение». Это соответствует тексту ITIL и практике большинства компаний, сознательно практикующих оба направления деятельности. Однако внимательный читатель не раз и не два встречал в тексте ITIL упоминания о «предотвращении *проблем*» в проактивной ветви процесса. Но даже этот внимательный читатель, скорее всего, не нашел объяснения тому, как problem management обеспечивает предотвращение проблем – ведь процесс имеет дело с *уже существующими* в инфраструктуре ошибками, и в самом проактивном из своих проактивных действий лишь предотвращает *инциденты*, инициируя устранение ошибок или снижение их влияния.

Итак, возможно ли в рамках управления проблемами обеспечить их *предотвращение*?

Суть процесса и его роль в управлении качеством ИТ сервисов

Процесс, известный как «управление проблемами», занимается поиском, анализом, и контролем ошибок в среде эксплуатации. Результатом контроля может явиться исправление этих ошибок путем проведения изменений.

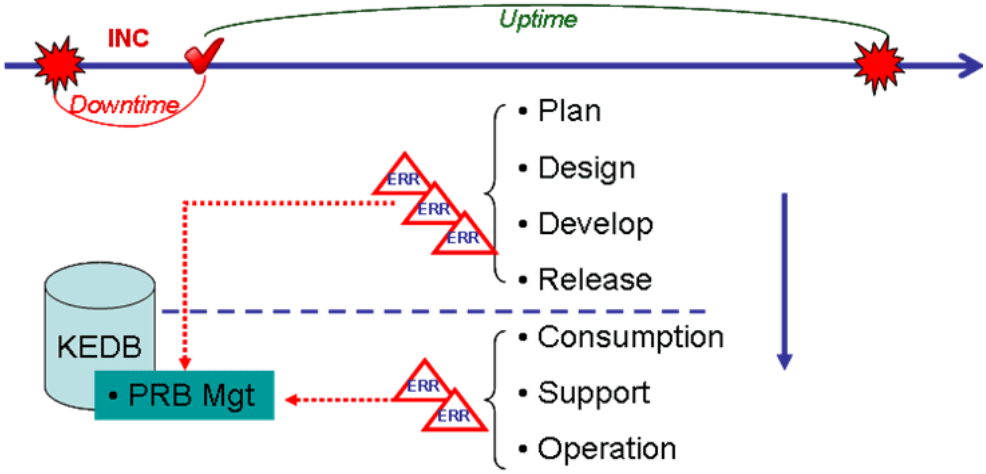
Роль процесса в жизненном цикле ИТ сервисов может быть проиллюстрирована следующим образом:

К сожалению, в жизни всякого ИТ сервиса на этапе эксплуатации можно выделить два состояния – нормальная работа и отклонение от нормы. Можно обозначит эти периоды как *uptime* и *downtime* соответственно – если мы договоримся понимать эти обозначения в контексте не только управления доступностью (доступен – недоступен), но максимально широко (соответствует норме – не соответствует).

Очевидно, что поставщик сервиса прилагает усилия к тому, чтобы сократить продолжительность периодов *downtime* и обеспечить максимальную длительность периодов *uptime*. Первое делается в рамках процесса управления инцидентами, а второе традиционно приписывается управлению проблемами. Однако если посмотреть на задачу обеспечения надежности чуть шире, мы увидим, что для того, чтобы сервисное решение дольше работало «правильно», необходимо, чтобы это решение было соответствующим образом спланировано и спроектировано, построено на основе надежных компонентов, корректно протестировано, описано и передано в эксплуатацию, - а там корректно сопровождалось, поддерживалось и потреблялось.

Если все перечисленные задачи выполнены хорошо и согласовано – вероятность наличия в инфраструктуре ошибок и, следовательно, возникновения инцидентов, будет относительно низкой. Однако, поскольку на практике все это, скорее всего, было сделано не лучшим образом, а исполнители друг с другом не встречались, в среде эксплуатации, вероятно, появились ошибки, сформированные на каждом из перечисленных этапов и – в особенности – на стыках этапов.

Те из ошибок service design, которые проходят фильтр управления релизами, попадают в среду эксплуатации – и в поле внимания управления проблемами[1]. Там к ним добавляются ошибки, возникающие вследствие некорректных сопровождения, поддержки и потребления.



Управление ошибками как рисками

Вне зависимости от способа обнаружения ошибок в среде эксплуатации и связанного с этим деления на реактивную и проактивную деятельность, в процессе управления проблемами делается примерно следующее:

- Идентифицированная (или вероятная) ошибка классифицируется с учетом вероятности проявления и возможного ущерба. Дальнейшая работа выполняется в отношении ошибок, преодолевших установленный барьер важности (приоритета).
- Проводится анализ ошибки и выработка планов восстановления (обходных решений для инцидентов) и предотвращения (изменений для устранения ошибки).
- С учетом результатов анализа ошибки и предложенных решений принимается решение о выборе модели дальнейшей работы с ней (подавать ли RFC, насколько

срочно, с какими ресурсами… - или продолжать жить с ошибкой, устраняя инциденты по мере их возникновения). Данные об ошибке публикуются в KEDB.
- Реализуется выбранная модель, контролируется ее эффективность.
- Если уязвимость устранена или угроза утратила актуальность – вследствие наших усилий или внешних факторов – ошибка закрывается.
- Если ошибка существует и актуальна, осуществляется периодический контроль эффективности выбранной модели деятельности.

Очевидно, что работа по управлению проблемами удобно укладывается в любую формальную модель управления рисками. Единственная особенность – процесс работает со специфическими уязвимостями, то есть ошибками в инфраструктуре ИТ.

Про проактивность

Часто оказывается, что ***расследуя причины инцидентов, аналитик может не только определить ошибку в инфраструктуре и оценить ее распространенность и влияние, но и высказать предположение о ее вероятном происхождении.*** Это может быть ошибка разработчиков, или ошибка планирования, или ошибка в документации по эксплуатации, или некорректная инсталляция – возможные источники были описаны выше. Однако ***управление проблемами обычно не имеет полномочий для внесения корректировок*** в перечисленные сферы деятельности. Используя терминологию ITIL v3, можно сказать, что процессы Service Operations не уполномочены напрямую инициировать коррективы в областях Service Design и Service Transition. Можно добавить: равно как и в деятельности разработчиков. Поэтому в жизни нередко бывает так, что ***устраненные ошибки впоследствии «возвращаются»*** - или возникают вновь, например, на других площадках.

Вероятно, авторы ITIL v3 тоже это заметили. В этой версии библиотеки есть и такая полезная часть, как Continual Service Improvement. Было бы логично предусмотреть в работе CSI такие процедуры:

- анализ ошибок, выявленных в процессе управления проблемами,
- оценку предположений аналитиков о природе появления этих ошибок (и/или проведение собственного анализа)
- инициацию корректирующих мер в отношении соответствующих процессов и функций для предотвращения повторного появления ошибок в среде эксплуатации.

У CSI есть для этого и полномочия, и инструментарий. Можно предположить, что нечто подобное имели в виду авторы книги Service Operations:
«Управление проблемами состоит из двух основных процессов:

- **Реактивного управления проблемами, которое обычно выполняется как часть Service Operations, и потому описано в данной книге**
- **Проактивного управления проблемами, которое инициируется в Service Operations, но управляется как часть Continual Service Improvement.»**[2]

К сожалению, эта идея никак не отражена в книге Continual Service Improvement.

В реальной жизни, тем не менее, можно предусмотреть следующие механизмы для предотвращения повторного возникновения ошибок в среде эксплуатации:

Предотвращение «своих» ошибок

Если вероятный источник ошибок – в области, за которую отвечают «свои» специалисты (разработчики, проектировщики, бизнес аналитики…), следует предусмотреть в процессе управления проблемами интерфейсы к этим областям. Механизм действия этих интерфейсов может быть аналогичен механизму обработки жалоб: ясная точка контакта, четкие требования к создаваемой записи, гарантированные сроки рассмотрения и форма ответа, наличие процедур эскалации.

При этом никаких гарантий внесения изменений – только возможность их предложить и получить обратную связь.

Предотвращение «чужих» ошибок

Если вероятный источник ошибок – вне сферы нашего контроля (сторонние поставщики и разработчики), можно предотвратить повторное «заражение» среды эксплуатации с помощью дополнительных фильтров в процессе управления релизами. Этот метод работает и в случаях, когда переданные «своим» данные об ошибках не привели к позитивным изменениям в передаваемых в эксплуатацию решениях. Опубликованные в KEDB данные о выявленных ошибках и способах их исправления должны использоваться не только процессами управления проблемами и инцидентами, но и процессом управления релизами.

Рассмотрим пример. Силами процесса управления проблемами была выявлена ошибка в ПО стороннего разработчика, и был найден патч, исправляющий ее или снижающий ее негативное влияние. Управление изменениями обеспечило эффективное применение патча для всех инсталляций этого ПО. Контрольный период показал, что инциденты, связанные с данной ошибкой, не повторяются. Ошибка признана устраненной и закрыта.

К сожалению, далее обычный сценарий выглядит так: при инсталляции очередного экземпляра ПО мы используем дистрибутив, предоставленный поставщиком. Поскольку он содержит ошибку, вновь возникают инциденты. Управление инцидентами устраняет их с помощью обходных решений из KEDB. Когда влияние инцидентов достигает предусмотренного порога, регистрируется проблема, из той же KEDB извлекается опробованный ранее способ устранения ошибки, устанавливается патч, инциденты прекращаются.

А вот другой возможный сценарий: при инсталляции очередного экземпляра ПО управление релизами обращается к KEDB и анализирует информацию о ранее обнаруженных в этом ПО ошибках. Включает в план построения и тестирования установку патча и проверку (не)проявления соответствующих инцидентов. Как следствие, в среде эксплуатации ни такая ошибка, ни такие инциденты повторно не возникают.

Таким образом, в процессе управления проблемами можно предусмотреть механизмы**, **позволяющие предотвращать** не только повторное и первичное проявление инцидентов, но и **повторное возникновение ошибок в среде эксплуатации.

Консультанты IT Expert

Подробнее эта тема обсуждается на следующих курсах:

- Руководство и контроль в ИТ. COBIT 4, ISO 20000, SOX
- Сертификационный курс Менеджер ИТ сервисов

Данная заметка отражает мнение автора, которое может не совпадать с уважаемыми первоисточниками (ITIL v2, ITIL v3, ISO 20000 COBIT, MOF и проч.). Комментарии и предложения темы для следующей заметки можно отправлять на items@itexpert.ru.

---

1. Конечно, к ним могут добавиться собственные ошибки управления релизами – ошибки компоновки, документирования, инсталляции и т.д.
2. Service Operations, глава 4.4.5