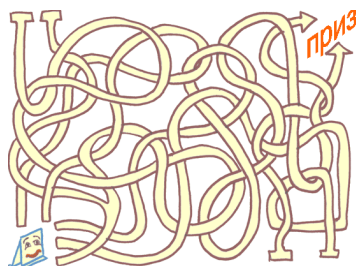




Выбираем типовое решение для автоматизации бизнес-процессов предприятия?

Хотим типовое решение без доработок!



Все комплексные решения так называемого «ERP-класса» обладают некоторой заявленной функциональностью. Состав таких систем очень схожий, но детали могут разительно отличаться. Навряд ли какая-либо из рассмотренных типовых систем подойдет идеально для Вашего предприятия. Но каждая из систем может подойти в большей или в меньшей степени. Как «примерить» на себя типовое решение? Как проанализировать наличие нужных функций, ранжировать их?

Что важно и принципиально, а чем можно пожертвовать и оставить на доработки? Процесс выбора похож на решение задачи со многими неизвестными.

В настоящей статье приводится концепция анализа типовых решений на предмет их применимости для предприятия. Эта концепция достаточно проста и не требует специальных знаний проектных технологий.

Лица, принимающие решение, сталкиваются с необходимостью оценки параметров и возможностей системы, в которых можно разобраться, лишь «прожив» аналогичный проект. А такой опыт есть далеко не у всех. Поэтому в ход часто идут более простые методы оценки типовых решений.

По нашей практике, решающим аргументом при выборе системы часто является демонстрация реально работающего типового решения на похожем предприятии. Логика здесь очень простая: «Если у соседа работает, значит, и у меня заработает». Но при таком подходе клиенты почему-то очень редко интересуются объемом доработок решения под специфику предприятия...

А ведь выбирается типовое решение – т.е. коробочный продукт, который предполагается использовать на предприятии. Но коробочный продукт – это только «исходная точка», задающая вектор развития системы. Где гарантия, что из выбранного коробочного продукта в Вашем случае удастся с приемлемым результатом построить систему, которая оправдает понесенные на нее расходы? Что доработки типового решения, выполненные конкретной командой внедрения, не пойдут вразрез с заложенной в него бизнес-логикой?

Принято считать, что альтернатива типовому решению – создание заказного решения. В этом случае уж точно вся специфика предприятия будет изначально заложена в систему. Минусы уже всем известны – высокие риски по качеству и полная зависимость от разработчика, устойчивость которого может быть под вопросом. Поэтому заказные системы давно уже «не в почете».



Клиенты стремятся «не изобретать велосипед», а внедрять типовые решения, рассчитывая на высокое качество головного вендора и его надежность. Насколько оправданы такие ожидания?

Дело в том, что любое типовое решение при внедрении дорабатывается, и становится в какой-то степени «Заказным». Переход от полностью «заказной» системы к полностью типовой – плавный! Чем меньше подходит выбранная система для предприятия, тем в большей степени она будет «заказной» после всех необходимых доработок.

Программа – необычайно податливый материал. Можно взять типовое решение и переработать его в процессе адаптации к предприятию на 100%. Например, благодаря открытости платформы 1С – все функции типового решения на внедрении можно переписать заново, если потребуется.

Отсюда аксиома:

«Внедрить» на конкретном предприятии для решения определенных бизнес-задач можно, при большом желании и определенной настойчивости любое типовое решение. Особенно это относится к решениям на платформе 1С. (Не в этом ли секрет распространенности 1С в России?)

Можно внедрить «1С-бухгалтерию» для планирования производства. А можно «Торговлю и склад» приспособить для финансового планирования...

Пару – другую рук и мозгов умелых программистов и постановщиков – и все будет работать!

Вопрос только в том, какое качество будет такой заказной системы. Каковы будут перспективы ее развития и гибкость. И во что такая разработка обойдется по расходам – платить программистам придется Заказчику. Что хорошо для программистов – есть где воплотить полет фантазии и причем за деньги клиента.

Потребители понимают, что если перед автоматизацией стоят типовые задачи, которые охватываются существующими типовыми решениями – разрабатывать что-то «свое» – элементарно нерентабельно. Это все равно, что строить автомобильный завод для производства двух-трех сотен автомобилей.

Но и использование типового решения не означает, что глубоких доработок не будет. На практике – чем больше предприятие, чем ближе решаемые задачи к управлению ресурсами и бизнес-процессами, оперативному учету и управлению – тем больше приходится дорабатывать типовое решение на внедрении.

Таким образом, заказной характер создаваемой системы (как ее качественный показатель) как правило, неизбежен. Независимо от того типовое или заказное решение внедряется.

Разумеется в типовом решении объем доработок можно минимизировать. С точки зрения функциональности – это главный и очевидный критерий выбора типового решения.

Минимизация доработок при этом – не самоцель. Потребитель рассчитывает на сопровождение основным вендором и сохранение качественных показателей, заложенных вендором в типовое решение.

Все вышеизложенные рассуждения – это предполсылки проблемы, они тривиальны и ни для кого уже не являются откровением.

Для многих алгоритм выбора типового решения на этом заканчивается. Анализируется функциональность и уже выполненные проекты. Казалось бы, что еще можно в этом направлении сделать?

Но если «копнуть глубже» (как убедимся далее, специальных знаний для этого не требуется) то можно обнаружить интересные вещи...

Какие именно доработки вредны для системы?

При выборе потребители рассматривают типовые решения на предмет минимальных доработок – это критерий. Но не всегда принимается во внимание, каков характер требуемых доработок! Далее мы покажем, насколько это важно.

Понятно, что минимизация доработок – это не самоцель.

Важно не количество доработок, а то, как именно эти доработки повлияют на сопровождаемость решения основным вендором. Насколько эти доработки способны «сломать» базовые механизмы системы.

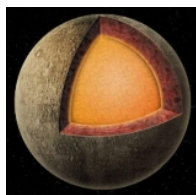
Кроме того, согласно закону перерастания количества в качество - при превышении некоторого порогового значения объема доработок качество системы может упасть недопустимо низко, а создание системы – окажется нерентабельным.

При попытке проанализировать некое типовое решение на предмет «предусмотрено/не предусмотрено» потребитель сталкивается с огромным количеством функций, форм, справочников, документов и т.д.

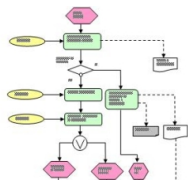
Мы часто видим, как потребители строят сравнительную таблицу функционала разных решений и после этого приступают к медитации над ней.

Пытаться проанализировать все это хозяйство «в лоб» - довольно затруднительно и по силу лишь ИТ-специалисту с глубокими знаниями как предприятия, так и типового решения. Как быть? Есть простой алгоритм.

Надо ранжировать все возможности типового решения на два уровня:



Уровень функциональности ядра, **возможность системы в принципе хранить и рассчитывать необходимые данные.**



Функциональность, направленная на обеспечение сервисов для выполнения бизнес-процессов и удобство пользователей. Это так называемая **«Функциональность взаимодействия между пользователем и программой»**. Например, это удобные для конкретных пользователей формы отчетности, копирование данных, автоматическое заполнение документов, подборы, предоставление нужной информации в нужные моменты в удобном виде, контроль за правильностью введенных данных на этапе ввода, подсказки пользователю, дружелюбность и интуитивная понятность интерфейса и т.д.

Конечно, программа с хорошо реализованным взаимодействием между пользователем и компьютером – очень привлекательна. Но это поверхностный подход, полностью оправдываемый при выборе программы, например, для мобильного телефона или веб-браузера для персонального использования. Пользователям в таких программах приятно работать – все нужные инструменты под рукой, интуитивно все понятно, легко, удобно и быстро.

Вот только в крупных корпоративных системах эти показатели отходят на второй план. Повышение комфортности работы «простых» пользователей – не цель очень дорогостоящего внедрения, не так ли? По крайней мере, в нашей стране.

Самое интересное в том, что требования к функциональности взаимодействия наиболее специфичны для разных предприятий, и здесь очень трудно найти предприятия со схожими требованиями даже в одной отрасли. Перечень анализируемых параметров системы здесь очень обширный. Анализировать – что подходит, а что не подходит в функциональности взаимодействия для предприятия – занятие неблагодарное. Для этого нужно проводить дорогостоящее функциональное моделирование бизнес-процессов предприятия в типовом решении. Что есть как раз первый этап проекта внедрения.

А вот требования к базовой функциональности (какие данные система способна хранить и обрабатывать) – более универсальны и более однозначны.

Представим себе, что менеджеру закупок нужно создать автоматически заявки поставщикам на основании рассчитанной потребности в материалах. Простая формулировка функции таит за собой огромное количество вариантов ее воплощения и особенностей бизнес-процесса. Это функциональность взаимодействия пользователя с компьютером. В типовом решении может быть заложена только ограниченное количество моделей (или вообще только одна), и совсем не очевидно что эти модели Вам подойдут полностью без доработок.

С другой стороны, такое базовое требование как «Система должна хранить рассчитанную потребность в материалах на даты в количественном выражении в разрезе заказов клиентов» является

однозначным, и реализация этой функции в конкретном решении достаточно легко поддается анализу. Например, в какой-то системе может оказаться, что в разрезе заказов получить потребность в принципе нельзя – она рассчитывается как сводная по всем заказам. Это ни что иное, как ограничение системы (она не способна рассчитывать и хранить эти данные), которое придется преодолевать путем доработок.

Мы видим, что принципиальные ограничения типового решения, связанные с отсутствием нужной базовой функциональности, проще увидеть при анализе базовой функциональности.

Мы говорим именно «ограничения», поскольку доработки решения на уровне базовой функциональности (ядра) есть ни что иное как пересмотр **модели учета (управления)**, заложенной в систему. В какой-то степени такой пересмотр может быть ломкой модели, заложенной разработчиками в систему, или нарушением ее целостности.

Отсутствие же в типовом решении нужной **модели взаимодействия** пользователя с компьютером – как правило не являются серьезным ограничением. Доработки в этой сфере можно выполнять как правило на уровне интерфейсов (внешнего вида программы и диалоговых форм), не «ломая» модель базовой функциональности.

Поэтому мы предлагаем на этапе выбора типового решения основное внимание уделить наиболее важному уровню – ядру системы. На этом уровне перечень анализируемых параметров резко сокращается, анализируемые параметры более конкретны и понятны, и самое главное – эти параметры решения являются универсальными для всех отраслей и предприятий! И именно от переделок функций ядра в большей степени зависит сопровождаемость системы от основного вендора и надежность ее работы.

Наглядный пример. Можно купить автомобиль с нужными базовыми характеристиками (мощность двигателя, наличие кондиционера, внедорожные возможности). Это базовые свойства «ядра». Если вы ошиблись в выборе – двигатель уже заменить не получится, и не получится поставить кондиционер. Проще продать автомобиль и купить новый. А вот что касается дальнейшего тюнинга – делайте что хотите в известных пределах.

К сожалению, многие проекты по внедрению типовых решений сопровождаются доработками, похожими на замену бензинового двигателя на реактивный с попутным приделыванием крыльев... И это цена выбора типового решения. Потребители часто не отдают себе отчета в том, что в программе есть двигатель (его замена – очень дорого и снятие с гарантии), а что есть отделка салона, рейлинги и колесные диски...

Часто приходится видеть, что потребитель при выборе типового решения забыл какую-то мелочь. При внедрении он просит ее реализовать и натывается на астрономическую стоимость доработок.

Казалось бы, чего уж проще – учитывать текущие запасы полуфабрикатов при планировании потребности в материалах...

Доработки функций на уровне ядра связаны с наибольшими рисками и как правило связаны со снятием с сопровождения доработанных модулей. Эти доработки имеют наибольший риск несистемности. Именно по этой группе доработки типового решения нежелательны.

Отметим, что изменения законодательства часто затрагивает функциональность ядра, что приводит к необходимости обновления релизов от разработчика системы. Снятие разработчиком ядра с сопровождения влечет серьезные трудности и затраты на поддержку соответствия системы требованиям законодательства.

Зато по этой группе наиболее просто определить еще на стадии выбора системы нужную функциональность. Достоверный, исчерпывающий перечень требуемой функциональности (а именно – какие данные способна рассчитывать и хранить система) снижает риск несоответствия выбранной системы ожиданиям потребителя.

Влияние на целостность системы доработок сервисов бизнес-процессов, функциональности взаимодействия между пользователем и компьютером диаметрально противоположны доработкам ядра. Сервисные возможности по определению относятся к некоторой локальной области функций, это своего рода «тюнинг», и их модификация и доработки не разрушают функциональность ядра.

«Тюнинг» системы:

- никак не влияет на целостность системы
- потенциально не способен разрушить ее целостность.
- требуют меньшей квалификации программиста и следовательно – дешевле.

Что можно отнести к «тюнингу»? Это разработка нового удобного отчета по заявке пользователя, ввода на основании, сервиса автозаполнения или подбора в документе, сервиса утверждений документа пользователями, печатные формы документов, этапность ввода документов, контроль ввода данных и т.п.

Вывод простой – все требования к АС должны быть поделены (ранжированы) на требования к ядру и требования к сервисам взаимодействия (надстройкам над ядром), и выбор решения должен быть в первую очередь основан на соответствии функциональности ядра требованиям, предъявляемым к ядру. С расчетом на то, что доработки сервисов взаимодействия обладают несущественными рисками и могут быть выполнены силами специфика предприятия.

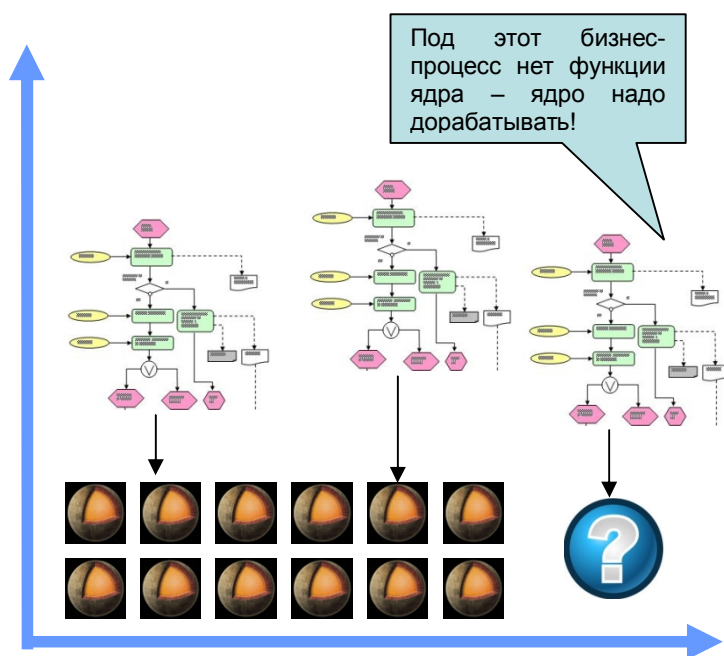
При этом именно доработки сервисов взаимодействия (надстроек) целесообразно выполнять под специфику предприятия, поскольку именно на сервисах специфика проявляется наибольшим образом и типовое решение как правило эту специфику никогда не покрывает в достаточном объеме.

Также, именно сервисы подвержены постоянным изменениям и доработкам командой ИТ предприятия в течение времени жизни АС. И наконец, модификация в области сервисов требует наименьшей квалификации программистов и в меньшей степени способна исказить данные в системе, разрушить их целостность и т.д.

Часто Заказчики заманиваются на проект красивыми сервисами взаимодействия (отчетами, навигаторами и т.д.), имеющимися в предложенном типовом решении.

Однако, после внедрения, все эти возможности часто превращаются в красивые безделушки, которые не удастся использовать только потому, что в них не учтена специфика предприятия или построенной системы. Либо клиент видит во внедренном решении глубоко заложенные изначально проблемы, которые значительно перевешивают внешнюю «красивость» программы.

Подытожим.



интерфейсов и организация взаимодействия пользователя с компьютером.

- **Функциональность ядра** — это фундамент корпоративной системы учета и управления, вмешательство в который крайне нежелательно. Это наполнение базы данных — *какие данные в принципе может рассчитывать и хранить система.*

- **Функциональность сервисов взаимодействия** — это «надстройка», которая допускает большую свободу действия (доработок, не очень сложных) со стороны потребителя без серьезных рисков. Это система

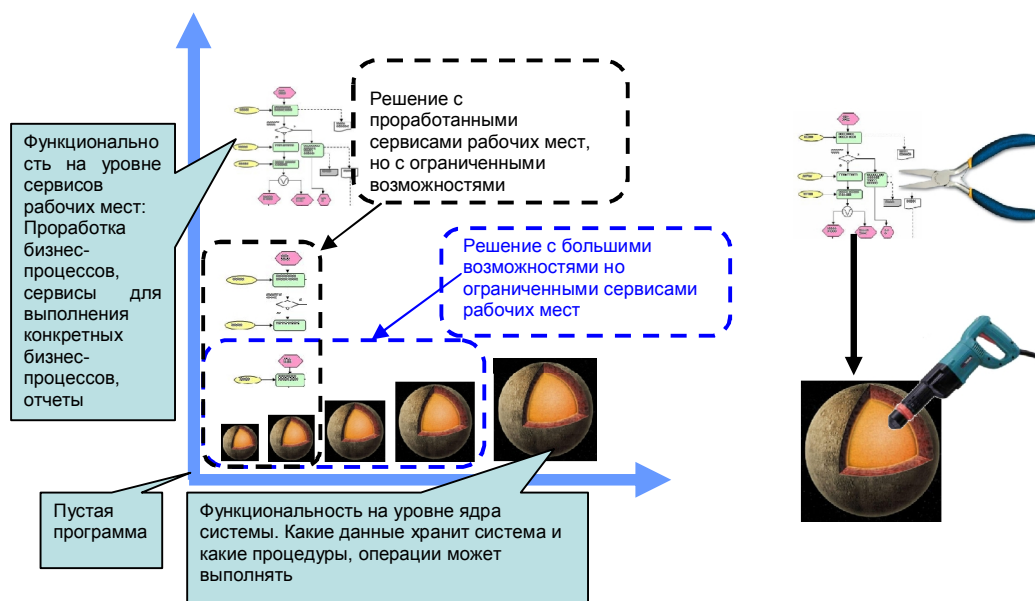
Разумеется, «надстройка» может существовать только на соответствующем ей фундаменте. Если в ядре нужной функциональности нет, то и о надстройке не может быть речи. Придется вмешиваться в ядро!

Практические рекомендации

Какая практическая ценность приведенных выше рекомендаций? Ну хорошо, предположим, мы убедились в результате проведенного анализа, что предложенное типовое решение требует доработок «ядра». Дальше-то что? Дорабатывать и снимать с сопровождения все равно придется!

Как раз необязательно!

Ведь разные типовые решения обладают разной базовой функциональностью! Отсюда следует, что нужно выбирать между разными типовыми решениями, сравнивая базовую функциональность, ну и, не забывая про надежность разработчика решения.



Выше было показано, что вмешательство и доработки в сервисах взаимодействия наименее болезненны в плане снятия с сопровождения типового решения и не приводят к ломке бизнес-логики системы. Однако в разных системах технологические возможности доработок сервисов взаимодействия реализованы по-разному.

Если сервисы взаимодействия плотно интегрированы в базовую поставку и неотделимы от него, то доработки этих сервисов опять приводят к снятию с сопровождения типового решения. В других типовых решениях интегрированные в базовую поставку (программу) сервисы взаимодействия являются лишь примером их реализации или сервисами по умолчанию. И потребитель может подключать к системе собственные реализованные сервисы без снятия базовой поставки с сопровождения.

Каким образом?

Многие современные типовые решения, в т.ч. на платформе 1С 8 обладают большими возможностями по подключению внешних программных модулей (т.н. «Библиотек бизнес-процессов»), в которых реализуются специфические для предприятия сервисы и надстройки взаимодействия пользователя с ядром системы. Такие внешние модули можно создавать и модифицировать на проекте силами собственных программистов. Подгрузка новой версии основной программы от основного вендора не вызывает в этом случае никаких проблем!

В результате ядро системы будет сопровождать основной вендор, а сервисы взаимодействия – будут сопровождаться и развиваться ИТ-отделом предприятия.

Пример типового решения от известной компании на платформе 1С 8, в котором реализовано подобное разделение функциональности на уровень ядра и на уровень сервисов взаимодействия – «ИТРП:Процессное производство 8».

Мы обращаем внимание на то что при выборе типового решения и анализе базовой функциональности правильнее рассматривать несколько альтернатив.

Может оказаться так, что потребитель о наличии альтернативных вариантов может просто не догадываться или не придавать этому значения. Он будет ориентироваться на типовое решение, наиболее распространенное на рынке,

вследствие больших средств, вложенных какой-нибудь крупной компанией в его маркетинг, продвижение, построение партнерской сети. Такая неосведомленность может дорого обойтись предприятию. Ведь монополизированный продукт – не всегда самый подходящий для именно Вас продукт.