12+

Кирилл Ледовский

ТОМ 3

ERP-проект. Как делать.

Технология проектирования сложной ERP-системы

МОЯ ERP ДЛЯ АНАЛИТИКА

Кирилл Ледовский
ERP-проект. Как делать

https://litres.ru/74374974
ISBN 9785007104913

Аннотация

Конечная точка этого тома тоже отличается от первых двух.

После него читатель должен видеть не только проблему и не только устройство правильной модели. Он должен видеть полный проектный маршрут и понимать, где находится его собственная задача: новое внедрение, переход на 1С: ERP, модернизация действующей системы, пересборка учёта, восстановление процессов, нормализация НСИ и т. д.

Содержание

ERP-проект. Как делать

Кирилл Ледовский

ISBN 978-5-0071-0491-3
Создано в интеллектуальной издательской системе Ridero

Введение. Почему ERP-проект распадается, даже когда каждый делает свою работу правильно

ERP-проект редко рушится в тот момент, когда кто-то совершает очевидную грубую ошибку. Намного неприятнее другая ситуация: каждый участник делает вполне профессиональную работу, документы появляются в срок, совещания идут по плану, аналитики рисуют процессы, методологи обсуждают учёт, разработчики закрывают задачи, а через несколько месяцев выясняется, что **все эти результаты плохо складываются в одну систему**. Процессная схема живёт своей жизнью, требования — своей, НСИ начинают обсуждать отдельно, в 1С: ERP уже появляются настройки и доработки, а тестирование проверяет сценарии, происхождение которых никто не умеет быстро восстановить.

На старте это почти незаметно. Аналитик получает задачу описать деятельность и проводит интервью. Финансовая служба формулирует требования к аналитике и отчётности. Специалист по НСИ собирает справочники и классификаторы. Разработчик открывает конфигурацию и ищет подходящие объекты. Руководитель проекта следит за сроками. Каждый решает реальную задачу, и в действиях каждого есть профессиональная логика. Проблема возникает между ними

— в местах, где **один результат должен стать основанием следующего**.

Например, руководителю нужен отчёт по финансовому результату в разрезе направлений деятельности. Учётная модель говорит: нужна аналитика. Прикладная команда отвечает: в 1С: ERP есть подходящий реквизит. Кажется, осталось сделать поле обязательным и передавать значение по документам. Но затем пользователь спрашивает: а в какой момент процесса я действительно уже знаю правильное направление? Если **значение требуется раньше, чем оно стало известно**, система заставит человека угадывать. Если позже — часть цифровой цепочки уже прошла без нужного признака. Вопрос о маленьком поле внезапно возвращает проект к процессу, хозяйственному смыслу, ответственности и отчётности одновременно.

Именно здесь становится видно главное ограничение традиционной организации ERP-проекта. **Отдельные профессиональные работы сами по себе не образуют связанную модель предприятия**. Хорошая BPMN-схема не гарантирует правильную аналитику. Хорошая учётная постановка не гарантирует, что нужный факт появляется в правильной точке процесса. Наличие подходящего объекта 1С: ERP не доказывает, что он выражает именно тот бизнес-смысл, который нужен предприятию. А зелёный технический тест не подтверждает сквозную работу, если его шаги не продолжают один и тот же фактический предмет и не

проверяют ожидаемый результат.

Поэтому эта книга начинается не с интерфейса 1С: ERP и не с выбора нотации. Она начинается с вопроса о связности самого проекта. Как сделать так, чтобы обследование не осталось архивом интервью, процессы не превратились в самостоятельный альбом схем, учёт не жил отдельным методологическим документом, НСИ не проектировалась в конце внедрения, а доработки не возникали из ближайшего удобного экрана? Как **провести один смысл через весь путь** — от реальной деятельности предприятия до данных, прикладной реализации и доказуемого результата?

Первые два тома подводили к этому вопросу с разных сторон: первый — через потерю объяснимости работающей ERP, второй — через предметный мир за документами и схемами. Подробно место третьего тома в серии разберём ниже; здесь достаточно одной мысли: **третий том переводит найденные различения в последовательность реального ERP-проекта**.

Полный маршрут книги состоит из десяти связанных этапов — от рамок и доказательной базы до сквозных сценариев, проверки и итоговой цифровой модели. **Подробная логика этих этапов будет раскрыта в первой главе**; во введении важнее увидеть их зависимость друг от друга.

Эта последовательность не означает жёсткий водопад, в котором запрещено возвращаться назад. Реальный проект постоянно уточняется. Новое интервью может обнаружить

скрытую систему. Новый источник — изменить понимание процесса. Прикладная привязка может показать, что аналитика вообще не имеет доказанной точки возникновения. Проверка способна обнаружить ошибку не в программе, а в исходной модели. Разница в другом: **возврат должен иметь адрес**. Команда должна понимать, какой слой требуется изменить, почему он изменяется и какие зависимые результаты после этого нужно пересмотреть.

Здесь особенно важна **цифровая нить**. В первой книге она появлялась как средство сохранить происхождение решения после ухода людей и смены версий. В проекте её роль ещё практичнее: она связывает существенный факт обследования с принятым решением, решение — с процессом и данными, данные — с прикладной реализацией, реализацию — со сценарием проверки, а последующее изменение — с теми частями модели, которые оно действительно затрагивает. Цифровая нить не заменяет документы и не превращает проект в бюрократию. Она не позволяет смыслу решения раствориться между документами.

Такой подход одинаково нужен при внедрении новой ERP и при модернизации уже работающей 1С: ERP. В новом проекте он помогает не дать техническим объектам слишком рано навязать архитектуру будущей деятельности. В действующей системе задача сложнее: часть модели уже материализована в документах, настройках, расширениях, интеграциях и пользовательских привычках, поэтому сначала приходит-

ся восстанавливать происхождение значимых решений и отделять действующую норму от исторических следов. Но **инженерная логика остаётся одной**: понять деятельность, связать её с экономическим смыслом и данными, определить прикладную реализацию и доказать результат.

Поэтому «ERP-проект. Как делать.» — не книга о том, как нарисовать идеальную схему и не каталог правильных настроек 1С: ERP. Она о другом: как не позволить одному ERP-проекту распасться на несколько параллельных проектов, каждый из которых по отдельности выглядит разумно. Важным результатом становится не количество страниц, моделей и настроек, а **способность пройти по связям и доказательно объяснить**: что предприятие решило, на каком основании, где этот смысл зафиксирован, как реализован и чем подтверждён.

Если такая связь сохраняется, **сложность перестаёт быть лабиринтом**. Система может быть большой, процессы — разветвлёнными, учёт — многослойным, конфигурация — существенно адаптированной. Но команда всё равно понимает, где находится действующая модель и как одно решение дошло до конкретного поведения ERP. Именно такую проектную дисциплину мы и будем собирать дальше — шаг за шагом, от границ проекта до проверяемого результата.

Место третьего тома в серии

Три книги серии отвечают на три разных вопроса одного и того же профессионального маршрута. Пер-

вый том ставит диагноз: почему сложная 1С: ERP может продолжать технически работать и одновременно становиться всё менее объяснимой для самого предприятия. Второй меняет способ смотреть на деятельность: показывает, почему документы, экраны и крупные процессные схемы являются проекциями более глубокого предметного мира. Третий переводит оба вывода в проектную работу: как из рамок, источников, обследования, предметов, процессов, учёта, данных и прикладных решений собрать один управляемый ERP-проект.

Первый том «**Мы теряем контроль над сложной 1С: ERP. Что делать?»** полезен прежде всего там, где система уже прожила несколько лет. Аналитики менялись, доработки накапливались, типовая конфигурация развивалась, а вместе с ней менялось само предприятие. В такой среде всё чаще приходится восстанавливать происхождение решений: почему это поле обязательно, почему действует именно такой статус, какой бизнес-смысл защищает старый код, какую версию проверял прежний тест. Том показывает, что **потеря контроля начинается не с технической аварии**, а с разрушения причинной памяти системы.

Второй том «**ERP-Лабиринты. Где выход?**» делает шаг назад — к моменту, когда эта память ещё только создаётся. Он показывает, как проектная команда сама может заложить будущий лабиринт, если слишком рано принять самый заметный документ за сам предмет управления. Заявка на ре-

монт начинает изображать весь ТОиР, заказ клиента — все продажи, ресурсная спецификация — всю технологическую подготовку. На схемах появляются десятки правильных действий, но предмет меняется по ходу стрелки, а границы процесса остаются условными. **Выход начинается с различения бизнес-предметов**, их состояний и предметных передач.

Третий том «**ERP-проект. Как делать.**» находится ровно между этими двумя проблемами. Он начинается там, где аналитик уже увидел предметный мир, но ещё не знает, как провести его через полноценный ERP-проект. Предметная модель сама по себе не отвечает, как организовать доказательную базу, когда вводить экономико-учётный ракурс, как определить обязательную аналитику, чем обосновать точку её регистрации, как сопоставить требование со штатным механизмом 1С: ERP и как построить проверяемую сквозную цепочку. Для этого нужна **технология проектной сборки**.

Поэтому третий том «ERP-проект. Как делать.» не повторяет первые два. Он использует их выводы как рабочие предпосылки. Из первого берётся требование сохранять происхождение и цифровую нить решения. Из второго — требование не подменять деятельность её документами и техническими носителями. Дальше **эти два принципа соединяются в полном маршруте**: от того, что мы подтверждённо знаем о предприятии, к тому, что проектируем, реализуем и проверяем.

Эти книги можно читать и отдельно. Руководитель действующей сложной ERP может начать с первого тома, потому что узнает там собственную боль. Аналитик, который постоянно спорит о границах процессов, может войти через «ERP-Лабиринты». А специалист, которому уже поручили организовать обследование, проектирование или модернизацию, способен сразу открыть третий том. Но вместе **серия даёт более важную последовательность** : почему система теряет смысл → из какого предметного мира этот смысл должен быть собран → как провести его через реальный ERP-проект и не потерять по дороге.

По этой причине третья книга сознательно ближе к рабочему столу аналитика. Здесь меньше места занимает объяснение, почему предметный подход вообще нужен, и больше — вопрос, **какой результат должен существовать после каждого этапа**. Что мы знаем после рамок проекта? Чем доказательная база отличается от просто большой папки файлов? Что именно должно дать обследование? Когда операционная модель готова к учётному ракурсу? Какие данные должны возникнуть до того, как аналитик идёт в форму 1С: ERP? Что необходимо проверить, прежде чем говорить, что решение работает?

Конечная точка этого тома тоже отличается от первых двух. После него читатель должен видеть не только проблему и не только устройство правильной модели. Он должен видеть **полный проектный маршрут** и понимать, где нахо-

дится его собственная задача: новое внедрение, переход на 1С: ERP, модернизация действующей системы, пересборка учёта, восстановление процессов, нормализация НСИ, анализ старых доработок, подготовка сценариев или проверка результата. С этого момента разговор уже можно вести не о методе вообще, а о конкретном проекте.

Как читать эту книгу и использовать её в ERP-проекте

Эту книгу лучше читать не как стандарт, который нужно последовательно исполнить от первой страницы до последней, а как **карту взаимозависимостей ERP-проекта** . Десять этапов образуют полный маршрут, но реальный проект может начинаться с разной точки и иметь разную глубину. Новое внедрение обычно требует более последовательного движения от рамок к модели и прикладной реализации. При модернизации часть результатов уже существует, но их приходится проверять, восстанавливать и связывать между собой.

Полезно **держать рядом один живой пример из собственной работы**. Это может быть заказ клиента, ремонтная потребность, производственный этап, обязательный реквизит, старая доработка, управленческий отчёт или интеграционный сценарий. По мере чтения попробуйте каждый раз отвечать на один вопрос: какой слой проекта сейчас объясняет этот объект и чего о нём ещё нельзя утверждать? Такой способ быстро показывает разницу между «мы нашли доку-

мент», «мы поняли процесс», «мы определили экономический смысл», «мы нашли место реализации» и «мы действительно проверили результат».

Книга специально различает уровни готовности. **Описанный результат — это ещё не спроектированное целевое решение**. Спроектированное решение — ещё не доказанная привязка к конкретной ERP. Найденное место реализации — ещё не воспроизводимый сценарий. Сценарий — ещё не факт успешного выполнения. Это различение может показаться строгим, но оно снимает значительную часть конфликтов между аналитиками, разработчиками, тестировщиками и заказчиком: все говорят об одном и том же результате, но понимают его текущую степень определённости одинаково.

Не пытайтесь раньше времени перепрыгнуть к 1С: ERP. Это одна из самых естественных профессиональных привычек аналитика: увидев требование, сразу искать документ, реквизит, функциональную опцию или расширение. Иногда это действительно полезно как гипотеза. Но в маршруте книги **технический механизм не является источником бизнес-смысла**. Сначала должно быть понятно, какое состояние предприятия мы хотим удержать, какой факт для этого нужен, где он возникает и кто отвечает за его определение. Только после этого вопрос «каким объектом 1С это реализовать?» становится корректным.

Точно так же не следует считать интервью самостоятель-

ным результатом обследования. Запись встречи, транскрипция и ответы экспертов являются источниками. Они становятся частью проектной модели только после того, как отделены подтверждённые факты, гипотезы, противоречия и решения. Если два специалиста описали один процесс поразному, хороший проект не обязан немедленно выбрать более правдоподобную версию. **Расхождение сначала становится открытым вопросом**, получает владельца и основание для дальнейшего решения.

При чтении главы об источниках обращайте внимание не на количество документов, а на адресуемость. В зрелом проекте можно быстро ответить, **из какого материала и какой версии возникло существенное положение модели**. Если источник изменился, новая редакция не должна молча переписать проектное решение. Сначала нужно понять, что именно изменилось и какие части модели зависят от нового основания. Здесь цифровая нить начинает работать ещё до настройки ERP.

В главах об операционной и экономико-учётной модели полезно **сохранять независимость слоёв**. Процесс не должен превращаться в бухгалтерскую проводку, а учётная модель — диктовать структуру деятельности только потому, что так удобнее существующей системе. Сначала определяется, что произошло в деятельности, затем — какой экономический смысл имеет событие, когда он признаётся, в какой оценке и по каким аналитикам. Эта последова-

тельность особенно важна для сложных управленческих моделей, где один технически проведённый документ может нести несколько разных хозяйственных последствий.

Раздел о НСИ, параметрах, аналитиках и отчётности лучше **читать с обратной стороны — от результата**. Если руководителю нужен показатель, попробуйте пройти назад: из каких фактов он складывается, какие аналитики различают эти факты, где значения впервые становятся известны, кто их определяет и каким объектом они фиксируются. Такой обратный маршрут быстро обнаруживает красивую отчётность без исходных данных и обязательные поля, которые пользователи вынуждены заполнять раньше, чем знают правильный ответ.

Когда книга переходит к прикладной реализации, сравнивайте не «наш документ» и «типовой документ», а **две причинные конструкции**. Что именно требуется предприятию? Какое поведение поддерживает штатный механизм? Какие параметры и данные делают его применимым? Что остаётся организационным или информационным разрывом, а что действительно требует изменения конфигурации? Такой подход одинаково полезен и при новом внедрении, и при попытке заменить старую доработку современной типовой возможностью.

Сквозные сценарии используйте как проверку связности модели, а не как ещё одну форму описания процесса. Сценарий должен проходить через фактические объекты и продол-

жать одну бизнес-историю. Если заказ создан отдельно, реализация оформлена по другому заказу, а платёж взят из третьего примера, три зелёных результата не доказывают одну сквозную цепочку. Проверяемый сценарий должен **сохранять фактическую преемственность данных и объектов** от начального условия до конечного результата.

Глава о проверке нужна в том числе для защиты от слова «готово». Подготовленная тестовая инструкция ещё не является результатом испытания. Автоматизированный сценарий ещё не является успешным прогоном. **Протокол старой версии не доказывает текущую конфигурацию**. Проверенным можно считать только тот объём, который действительно выполнен в определённой среде, на определённых данных и с сохранёнными доказательствами результата. Всё остальное может быть описано, спроектировано или подготовлено к проверке — и это тоже полноценные результаты, просто другого уровня.

Не все проекты обязаны доходить до последней стадии. Иногда предприятию нужна только подтверждённая модель рамок и обследования. Иногда — операционная и экономико-учётная модель для передачи другому интегратору. Иногда главная задача — восстановить НСИ и аналитику, пересобрать прикладные решения или подготовить программу приёмки. **Остановка после отдельного этапа не делает работу незавершённой**, если результат имеет самостоятельное назначение и честно обозначает, чего он ещё не до-

казывает.

И последнее правило чтения: не превращайте эту книгу в ещё один архив знаний. Её ценность проявляется только тогда, когда принципы возвращаются в текущий проект. Выберите один спорный участок, пройдите его по маршруту от рамок и источников до прикладной реализации и проверки, отметьте **места, где ответ приходится угадывать**, и не маскируйте их красивым текстом. Именно эти точки показывают, где проект теряет связность.

Дальше книга будет двигаться тем же способом. Сначала мы установим рамки: что именно считается объектом работы и с какой глубиной. Затем разберёмся с источниками и обследованием. После этого соберём модель деятельности и экономический ракурс, определим нужные данные и только затем перейдём к 1С: ERP, сквозным сценариям и проверке. Каждый следующий раздел будет отвечать не только на вопрос «что сделать», но и на два более важных: **какой результат должен остаться после этого шага** и чего этот результат пока ещё не доказывает.

Глава 1. Один ERP-проект вместо десяти параллельных работ

Когда каждый сделал свою часть, а целого всё равно нет

На проектном совещании лежат вполне приличные результаты. Аналитики принесли схемы процессов. Финансовая служба подготовила требования к учёту и отчётности. Специалисты по данным собрали справочники и аналитику. Разработчики уже показывают формы 1С: ERP и предлагают варианты реализации. Отдельно существует план тестирования. Если смотреть на каждый материал по отдельности, **работа выглядит профессиональной и завершённой**.

Проблема обнаруживается, когда руководитель задаёт один сквозной вопрос. Например: почему показатель прибыли по направлению деятельности должен формироваться именно так? Процессная схема показывает действия. Учётный документ объясняет формулу. В 1С: ERP найден подходящий реквизит. Разработчик показывает, где он заполняется. Тестировщик демонстрирует сценарий. Но **связать всё это в одну доказуемую историю** внезапно оказывается трудно: кто и в какой точке процесса впервые определяет значение, почему именно это значение считается правильным, как оно проходит по цепочке и какой результат под-

утверждает его корректность.

Так возникает парадокс большого ERP-проекта. В нём может быть много правильных документов и одновременно отсутствовать одна правильная модель. Процессная команда отвечает за процессы, учётная — за экономический смысл, техническая — за систему, а тестирование — за проверку. Пока **между этими результатами нет преемственности**, каждое следующее направление вынуждено частично начинать исследование заново и создавать собственную версию того, что предприятие имело в виду.

Снаружи это похоже на обычную проблему координации. Кажется, достаточно чаще проводить общие совещания, завести единый каталог файлов и назначить ответственных. Такие меры полезны, но они не решают главного. У проекта должен быть не только общий архив, но и **общий путь происхождения решений**: из какого факта возникло требование, в каком процессе оно имеет смысл, какие данные нужны для его исполнения, где оно материализовано в ERP и чем подтверждён результат.

Именно этот путь превращает набор материалов в **связанную цифровую модель предприятия**. Под цифровой моделью здесь понимается не одна гигантская схема и не отдельная программная база. Это согласованный комплекс представлений, между которыми можно пройти по смысловой связи: рамки проекта, источники, подтверждённые сведения обследования, модель деятельности, экономико-учёт-

ный ракурс, данные и НСИ, прикладные решения, сквозные сценарии и результаты проверки. Каждый слой отвечает на свой вопрос, но не живёт отдельно от остальных.

Десять этапов одной модели

Полный маршрут удобно представить как десять последовательных этапов. Сначала определяются рамки проекта: что именно рассматриваем, с какой глубиной и что сознательно оставляем за пределами текущей работы. Затем формируется доказательная база — понятный и адресуемый корпус источников. После этого проводится обследование, которое превращает материалы и экспертные знания в **подтверждённую фактическую основу**.

Из подтверждённых сведений строится **операционная модель предприятия**: направления деятельности, бизнес-предметы, процессы, процедуры, роли и передачи. На неё накладывается экономико-учётный ракурс, чтобы каждое значимое хозяйственное событие получило понятный экономический смысл, момент признания, аналитику и правила отражения. Только после этого можно обоснованно проектировать НСИ, параметры, реквизиты, статусы, точки регистрации данных и отчётность.

Следующий этап связывает уже принятые требования с конкретной ERP: объектами, формами, механизмами, правами, интеграциями и необходимыми изменениями. Затем отдельные элементы модели собираются в сквозные сценарии, где один конкретный случай проходит через процес-

сы и прикладные объекты от исходного условия до результата. После этого появляется возможность проверки — ручной или автоматизированной — и только затем формируется **итоговая конфигурация связанной модели** с честно обозначенным уровнем подтверждения.

Этот порядок важен не как ритуал. Он выражает **зависимость результатов**. Невозможно окончательно решить, каким реквизитом фиксировать факт, пока неясно, какой факт требуется. Невозможно определить полный состав фактов, пока не понятен экономический результат. Невозможно корректно интерпретировать экономическое событие, если сама операционная ситуация описана расплывчато. А обследование неизбежно становится бесконечным, если заранее не установлено, какая часть предприятия относится к текущему проекту.

В реальной работе этапы могут частично перекрываться. Аналитик способен заглянуть в 1С: ERP уже во время обследования, чтобы проверить техническую гипотезу. Учётный специалист может заранее обозначить критическую аналитику. Разработчик может обнаружить ограничение типового механизма ещё до завершения всей модели. Это нормально, пока найденная **информация возвращается в тот слой, которому принадлежит решение**, а не превращается в самостоятельное техническое правило только потому, что его удалось быстро реализовать.

Почему следующий этап не должен начинать про-

ект заново

Самая дорогая потеря в большом проекте происходит не тогда, когда информация вообще не собрана. Она происходит, когда информация собрана, но **следующая команда не может использовать её как устойчивый вход** и вынуждена заново восстанавливать смысл. После обследования появляется отчёт на двести страниц, но процессные аналитики снова проводят интервью. После процессного проектирования появляется набор схем, но команда учёта снова выясняет, что именно происходит в хозяйственной операции. Прикладные специалисты затем повторяют тот же путь, пытаясь понять, зачем нужен реквизит или статус.

Поэтому каждый крупный этап должен иметь понятный вход, собственное преобразование и результат, пригодный для следующего шага. Не обязательно, чтобы все участники работали в одном файле или одной системе. Важно другое: **следующее решение должно опираться на уже принятый результат**, а не на личное понимание его автора. Тогда операционная модель действительно наследует подтверждённые факты обследования, экономико-учётная модель — операционные события, модель данных — экономические и управленческие требования, а прикладное решение — уже определённый смысл.

При этом рабочая модель проекта и документ, который видит заказчик, не обязаны быть одним и тем же объектом. Рабочая модель может быть значительно структурнее:

в ней хранятся связи, версии, источники, состояния и зависимости. Человекочитаемый документ нужен для обсуждения, проверки и принятия решений. Опасность начинается тогда, когда **правка красивого отчёта незаметно становится правкой исходной модели**. Существенное замечание должно возвращаться в тот слой, которому принадлежит, изменять его принятую версию и уже после этого порождать новую человекочитаемую проекцию.

Здесь особенно важна **цифровая нить**. Она отвечает не на вопрос, где лежит файл, а на вопрос, как решение прошло через проект. Если новый документ или новая версия источника меняют существенный факт, сначала фиксируется новое основание. Затем определяется, какой слой затронут, кто отвечает за его пересмотр и какие последующие результаты зависят от изменения. Проверка, обнаружившая дефект, тоже не должна напрямую переписывать процесс или учётное правило: она создаёт доказательство проблемы, после чего корректируется владеющая часть модели.

Именно так проект перестаёт быть цепочкой передач «вот наш отчёт, дальше разбирайтесь». **Следующий участник получает не только файл, но и определённый смысл**, границу применимости и контекст происхождения. Это кажется небольшим организационным изменением, но именно оно позволяет большой системе переживать смену аналитиков, подрядчиков и версий, не превращая каждое новое изменение в археологию.

Результат этапа должен иметь самостоятельную ценность

Полный маршрут не означает, что каждый ERP-проект обязан дойти до последней точки. Иногда предприятию нужно только надёжно определить рамки и провести обследование. В другом случае требуется операционная модель, которую затем реализует собственный проектный офис. Третий проект заканчивается экономико-учётной постановкой и требованиями к данным. Четвёртый действительно проходит весь путь до фактической проверки в 1С: ERP. **Глубина определяется задачей, риском и договорным объёмом**, а не желанием обязательно пройти все ступени.

Поэтому **крупный этап должен оставлять самостоятельный результат**. Подтверждённое обследование можно передать другой команде как фактическую основу. Операционная модель полезна для перераспределения ответственности и дальнейшей автоматизации даже без готовой настройки ERP. Экономико-учётная модель может использоваться как основание новой отчётности. Сквозные сценарии способны стать программой приёмки, даже если автоматизированное тестирование не входило в проект.

Но **самостоятельность результата нельзя путать с универсальностью его доказательств**. Операционная модель показывает, как устроена деятельность в принятых рамках, но не доказывает, что именно так система уже работает. Прикладная привязка показывает, где предполагается

реализовать требование, но не доказывает корректность всей цепочки. Проверяемый сценарий описывает, как доказать результат, но сам ещё не является доказательством успешного выполнения.

Эта граница особенно полезна при споре о готовности. Вместо общего слова «готово» появляется точный ответ: какая часть проекта описана, что уже спроектировано, где найдено место реализации, что подготовлено к проверке и что фактически подтверждено. Разговор перестаёт зависеть от впечатления о количестве страниц и **переходит к состоянию результата**.

Пять уровней готовности результата

В сложном проекте слова «описано», «спроектировано», «реализовано» и «проверено» часто звучат как почти равные. Из-за этого стороны способны честно согласиться, что задача завершена, а через неделю обнаружить, что имели в виду разные вещи. Чтобы этого избежать, полезно **различать пять уровней**.

Первый уровень — описано. Это означает, что определённый аспект предприятия зафиксирован на основании принятых источников и решений. Второй — спроектировано: **определена целевая конструкция**, то есть уже не просто восстановлено текущее состояние, а принято, как должно быть. Третий — привязано к системе: для требования найдено конкретное техническое место реализации в целевой ERP или смежной системе.

Четвёртый уровень — проверяемо. Здесь существует воспроизводимый сценарий, который позволяет однозначно отличить ожидаемый результат от ошибки. Пятый — фактически проверено: сценарий действительно выполнен в определённой среде, на определённых данных, и результат зафиксирован. **Каждый следующий уровень требует дополнительной работы**; он не возникает автоматически из предыдущего.

Такое различение хорошо дисциплинирует ожидания. Согласованная схема процесса — ценный результат, но она не становится доказательством настройки. Найденный типовой механизм — важный шаг, но он ещё не доказывает сквозное исполнение. Написанный тест — не успешный прогон. И наоборот, если проект сознательно завершён на уровне спроектированной модели, это не «недоделанный проект», а **честно определённый объём с понятной границей**.

Что в итоге называется связанной цифровой моделью

Итоговая цифровая модель не обязана содержать все возможные слои. Она содержит те, которые действительно были разработаны в текущем проекте, и **сохраняет между ними смысловые связи**. В одном случае это рамки, доказательная база, обследование и операционная модель. В другом добавляются экономико-учётные решения, данные, прикладная реализация, сценарии и фактические результаты проверки.

Ключевое свойство здесь — **не полнота списка, а преемственность**. Факт обследования должен быть узнаваем в операционной модели, операционное событие — получать экономическую интерпретацию, экономическая потребность — формировать требование к данным, а данные — иметь определённое место возникновения и использования в ERP. Модель становится источником сценария, сценарий — источником проверки, а существенное изменение по цифровой нити возвращается в тот слой, который оно действительно меняет.

Если эти переходы сохранены, предприятие получает не просто комплект проектной документации. Оно получает объяснимую конструкцию, в которой можно двигаться как вперёд — от нормы к реализации, так и назад — от странного поля, цифры или доработки к причине её появления. Именно такая способность понадобится дальше, когда мы начнём последовательно разбирать каждый этап. Первый из них кажется самым простым и поэтому особенно часто выполняется поверхностно: **нужно точно определить, что именно входит в ERP-проект**.

Глава 2. Сначала определить, что именно мы проектируем

Один завод — несколько разных границ

На старте проекта заказчик часто говорит очень просто: **«Нужно внедрить ERP на предприятии»**. Формулировка кажется достаточной, пока команда не начинает уточнять детали. Выясняется, что юридических лиц несколько, производство расположено на двух площадках, центральный склад обслуживает обе, транспортный участок пока остаётся в старой системе, столовая нужна только для учёта затрат, лаборатория участвует в качестве, но работает во внешнем решении, а часть ремонтных функций вообще выполняет подрядчик.

В этот момент появляется соблазн **провести одну общую границу**: всё, что существует внутри организации, считать частью проекта. Это выглядит аккуратно и даже безопасно — вроде бы ничего не забыли. На практике такая граница создаёт противоположную проблему. Проект начинает одинаково глубоко обсуждать вещи, которые нужны ему по совершенно разным причинам. Производство действительно требуется моделировать и автоматизировать. Столовую, возможно, достаточно учесть экономически. Внешнюю лабораторию — описать как интерфейс. Транспорт — оставить в

другой системе, но сохранить обмен данными.

Именно поэтому **понятие «входит в проект» слишком грубое**. Для большого предприятия нужно несколько независимых рамок. Один объект может входить в организационный охват, но не требовать детального процессного проектирования. Процесс может быть нужен для понимания хозяйственных событий, но остаться ручным. Система может вообще не меняться, но участвовать в интеграции. Реализованное решение может быть шире, чем согласованный объём проверки.

Это не бюрократия и не попытка заранее усложнить проект. Наоборот, **независимые рамки позволяют не тащить в глубокую разработку всё предприятие** только потому, что оно существует. Они дают возможность говорить о конкретном объёме работ без постоянной подмены: «мы же знали, что у нас ещё есть этот участок, значит он тоже должен быть автоматизирован».

Шесть рамок, которые не обязаны совпадать

Первая рамка — **организационная и юридическая**. Она отвечает на вопрос, какие юридические лица, филиалы, площадки и организационные единицы относятся к текущему проекту. Принадлежность одному холдингу сама по себе ничего не решает. Две компании группы могут жить на одной ERP, но иметь совершенно разные процессы; или наоборот, один процесс может проходить через несколько юридических лиц.

Вторая рамка — операционная. Она определяет, **какие виды деятельности и процессы нужно обследовать и моделировать подробно**. Например, транспортный участок может находиться внутри предприятия, но в текущем проекте нас интересует только факт оказанной транспортной услуги и её связь с производством. Тогда не требуется разбирать всю диспетчеризацию, маршруты и эксплуатацию автопарка до операций.

Третья рамка — **учётная**. Она часто шире операционной. Предприятие может не проектировать внутреннюю технологию столовой, медпункта или вспомогательного участка, но обязано понимать, какие расходы, активы, обязательства и аналитики оттуда попадают в общую экономическую модель. Если такой контур игнорировать только потому, что он не автоматизируется глубоко, финансовая связность проекта окажется нарушена.

Четвёртая рамка — автоматизационная. Она показывает, что именно будет реализовано, изменено или перенесено в ERP. **Наличие процесса в модели ещё не означает, что его нужно автоматизировать**. Иногда достаточно определить интерфейс с внешней системой. Иногда ручная процедура осознанно остаётся ручной, но её результат должен быть зарегистрирован в ERP в конкретной точке.

Пятая рамка — интеграционная. Она отвечает за внешние системы и организационные контуры, с которыми проектируемая часть должна обмениваться предметами, данны-

ми или состояниями. Внешняя система может не изменяться вообще, но **интерфейс с ней является частью проекта**: нужно знать, что передаётся, когда, в каком состоянии, как связываются экземпляры и что происходит при ошибке.

Шестая рамка — рамка проверки. Она определяет, **какой объём решения должен быть подтверждён сценариями и фактическими испытаниями**. Проверка может охватывать не всё, что было описано и даже не всё, что было реализовано. Для критичных цепочек потребуется глубокий сквозной тест, а для второстепенных функций — ограниченная проверка или вообще только документированное решение. Главное, чтобы это различие было принято заранее.

Конец ознакомительного фрагмента.