11
Проектируем для людей

В предыдущих главах мы обсудили концепцию персон и убедились в важности преобладания целей над задачами. Только после того, как мы определили пользовательских персон и выяснили их цели, мы можем переходить к изучению задач, будучи уверенными, что они не нарушат процесс проектирования. Наш метод для объединения задач в единое целое мы называем «сценариями». Сценарий — это краткое описание действий пользователя, которые он предпринимает при взаимодействии с программным продуктом, пытаясь достигнуть своих целей. Далее в этой главе я остановлюсь на сценариях более подробно, а также расскажу о некоторых других инструментах, полезных при проектировании. Я также опишу кейс, из которого можно будет понять, как эти инструменты, в частности сценарии, работают на реальных проектах.

Сценарии

По мере того как в ходе процесса проектирования внимание смещается от общих концепций к проработке деталей, эффективность сценариев также повышается. Мы разыгрываем эти сценарии с нашими персонами, будто с актерами, изучающими свою роль, и через это проверяем правомерность наших предположений в отношении проектирования. Ничего удивительного, что такой процесс работы с нашими сценариями напоминает подход актера к постижению характера своего героя, когда актер вживается в образ персонажа, чтобы увидеть мир его глазами и ощутить его эмоции. Мы тоже пытаемся думать так, как это сделала бы наша персона. Мы на время

забываем о собственном уровне образования, способностях, степени подготовки и методах, которыми владеем, воображая себя человеком, обладающим опытом и восприятием, присущим *персоне*. Учитывая, что мы являемся проектировщиками, а не актерами, воплотить подобное без понимания контекста и знания нюансов бывает нелегко, потому сценарии оказываются мощным подспорьем. К примеру, если мы обладаем информацией о том, что Бетси создает веб-сайт для страховой компании, нам становится гораздо проще проникнуть в самую ее суть. Это может показаться странным, однако на самом деле это вполне нормальная ситуация. Это так же, как программисты проникают в самую суть компьютеров. Для программистов становится привычным даже описывать действия компьютеров от первого лица, например, от них можно услышать: «Я получил доступ к базе данных и теперь сохраняю записи в своем кэше». Хотя они употребляют местоимение «я», не они выполняют все действия, а компьютер. Однако, ставя себя на место компьютера, им легче понять, что происходит у него внутри и что ему требуется, а затем реализовать это в коде.

Сценарии обычно составляются исходя из информации, полученной на первом этапе проведения исследования. Из интервью с пользователями, а также в ходе прямого наблюдения за их действиями о задачах можно узнать довольно много. Целям свойственна стабильность, они практически не меняются, а вот задачи изменчивы, и нередко при компьютеризации деятельности часть этих задач отпадает. Проходя этап разработки сценариев, мы традиционно отыскиваем задачи, которые существуют лишь потому, что так исторически сложилось, а затем, по мере возможности, избавляемся от них.

Качественный сценарий в большинстве случаев является таковым благодаря ширине охвата, нежели глубине детализации. Другими словами, более важно, чтобы в сценарии присутствовало описание какого-либо процесса от начала и до конца, чем подробное описание каждого шага с мельчайшими нюансами.

Здесь важно прорабатывать только те сценарии, которые будут способствовать процессу проектирования, и не углубляться в исключения. Обычно мы создаем сценарии двух видов, хотя их может быть и больше, — это сценарии на основе ежедневных действий и сценарии на основе обязательных действий.

Сценарии ежедневных действий

Сценарии ежедневных действий представляются наиболее важными и полезными. В них описываются самые частые действия пользователя. Например, для систем отслеживания ошибок и неисправностей в программах типичным сценарием, выполняемым ежедневно, будет, соответственно, поиск ошибок и заполнение отчетов об ошибках, найденных в текущем периоде. Любой специалист технической поддержки выполняет две эти задачи по несколько раз каждый день.

В целом у большинства пользователей количество таких описанных ежедневных сценариев будет весьма ограниченным. Обычно число таких сценариев не превышает одного или двух. Больше трех их бывает крайне редко.

Сопровождение сценариев ежедневных действий в пользовательском взаимодействии должно быть невероятно качественным. Новым пользователям требуется осваивать такие сценарии очень быстро, а потому нужна хорошая встроенная система обучения. Это значит, что пояснения к применению различных функций должны быть описаны прямо в интерфейсе самой программы. Тем не менее чем чаще пользователь работает с программой, тем меньше ему в дальнейшем требуется использовать эти пояснения и инструкции. Вместо этого вскоре возникнет потребность в освоении быстрых сочетаний клавиш. Более того, еще чуть позже, по мере приобретения опыта работы с программой, у пользователей возникнет и потребность в индивидуальной подстройке программы под их персональный стиль действий с учетом личных предпочтений.

Сценарии обязательных действий

В сценариях обязательных действий описываются все действия пользователей, которые нужно исполнять непреложно, однако нечасто. К этому виду сценариев можно отнести такие действия, как чистка баз данных и создание исключительных запросов. Взаимодействия на основе обязательных действий также требуют качественного обучения. Тем не менее в данном случае пользователь никогда не перерастет их и не станет применять параллельные действия вроде сочетаний клавиш. Ввиду нечастого использования этих операций пользователь подстроится под настройки программы и не потребует персонализации. Таким образом,

разработчики будут освобождены от необходимости дорабатывать этот вид взаимодействия до того уровня, который требуется при использовании сценариев ежедневных действий. Приблизительно по той же причине салон нового «Ягуара» отделан со всей роскошью, в сравнении с грубой металлической окантовкой его моторного отсека.

Несмотря на то что почти у всех программных продуктов количество сценариев обязательных действий невелико, их количество все же превышает число ежедневных сценариев.

Сценарии исключений

Конечно же, есть и третий вид сценариев — сценарии исключений. Программисты по природе своей уделяют внимание исключительным ситуациям, тем не менее при проектировании такими сценариями можно пренебрегать. Это не значит, что соответствующий функционал можно удалить из программы, — это значит, что проектировать взаимодействие по этим сценариям можно грубо и прятать его поглубже в интерфейс. От способности обрабатывать исключения зависит лишь качество кода, а вот успех самого *программного продукта* обусловливается способностью справиться с ситуациями, описанными в сценариях ежедневных и обязательных действий.

Если пользователь выполняет какую-то задачу постоянно, соответствующее взаимодействие должно быть спроектировано на высшем уровне качества. То же справедливо и для задачи, которая выполняется редко, но неизбежно. Взаимодействие для нее хотя и преследует иные цели, также должно быть качественно спроектировано. А вот проектирование взаимодействия для задач, выполнение которых необязательно или происходит не каждый день, не требует такого скрупулезного подхода. Ресурсы в виде времени и денег всегда ограниченны, а потому это отличная возможность сэкономить и перераспределить ресурсы на более полезные вещи. Готовиться нужно ко всем видам сценариев, но детально проектировать взаимодействие следует под те из них, которые наиболее важны или происходят чаще остальных.

* * *

Персоны, цели и сценарии — тяжелая артиллерия в нашем арсенале проектирования. Прежде чем переходить к изучению кейса практиче-

ского применения сценариев, я бы хотел рассказать еще о нескольких полезных концепциях проектирования, таких как адаптивный интерфейс, вечная середина, терминология, мозговой штурм и латеральное мышление.

Адаптивный интерфейс

Взаимодействие всегда можно сделать еще проще — для этого достаточно просто удалить часть опций, уменьшив тем самым общую функциональность продукта. Нередко такая тактика не оправдывает себя, однако бывают и исключения. Задача проектирования более сложного уровня требует простоты в использовании продукта без принесения в жертву его функционала и производительности. Добиться подобного непросто, но вполне возможно. Для этой ситуации потребуется применить технику, которую я называю *адаптивным интерфейсом*.

Несмотря на то что в программе должно быть множество самых разных функций, у конкретного пользователя в каждый конкретный момент времени нет потребности использовать их все одновременно. Под каждый сценарий персоне пользователя понадобится использовать лишь небольшое подмножество элементов управления и связанных с ним данных, хотя это подмножество каждый раз может быть разным в зависимости от выполняемой задачи. Интерфейс станет в разы легче для восприятия, если нужные для реализации ежедневных сценариев элементы управления и данные разместить на видном месте, в то время как все прочие элементы будут перемещены на второй план, за пределы зоны видимости.

Интерфейсы многих серьезных программ схожи с меню в китайском ресторане, где каждая страница испещрена сотней вариантов выбора. Это может казаться подходящим при заказе ужина, но вот в высокотехнологичных продуктах такая особенность крайне мешает восприятию.

Например, на стандартной панели инструментов Microsoft Word расположены значки загрузки, закрытия и печати документа. Пользователь обращается к этим задачам довольно часто, поэтому такое размещение вполне оправданно. Однако тут же рядом с ними можно обнаружить и значки для генерации схемы документа и внедрения электронных таблиц. Компания *Microsoft* разместила значки этих опций на основной панели инструментов, чтобы мы смогли оценить всю мощь программы.

Только, к несчастью, большинству пользователей эти опции не пригодятся никогда, а даже если и пригодятся, они не будут пользоваться ими постоянно. Этих функций не должно быть на панели инструментов, потому что панель инструментов — это компонент интерфейса, традиционно предназначенный только для самых востребованных функций.

Вечная середина

Как правило, самые мощные из наших инструментов проектирования позволяют нам лучше понять личности наших пользователей, представить, как они выглядят, и проникнуть в самую их суть. Одна ментальная модель, которую мы применяем постоянно, называется «вечная середина». Большинство пользователей нельзя отнести ни к неопытным, ни к экспертам — они находятся где-то посередине. Вспомните Рупака, Шэннон, Декстера и Роберто, об уровне компьютерной грамотности которых шла речь в главе 9 «Проектируем для удовольствия». Несмотря на то что их опыт и специализация весьма различны, всех их можно отнести к категории вечной середины.

Опыт людей, которые взаимодействуют с интерактивными системами, можно изобразить с помощью кривой нормального распределения (колоколообразной кривой). Если мы построим такой график для любого электронного продукта, изобразив по вертикали количество пользователей, а по горизонтали их уровень компьютерной грамотности, мы увидим, что в крайней левой и крайней правой позиции будет небольшое число неопытных и продвинутых пользователей соответственно, а преобладающее количество средних значений займет всю центральную часть.

Вечная середина