Встаньте!

Уди Дахан

Для многих из нас карьера архитектора начиналась с какой-либо чисто технической должности, где успех определялся в основном способностью общаться с компьютерами. Однако в роли архитектора нам приходится общаться главным образом с другими людьми. Обсуждаете ли вы с разработчиками преимущества того или иного шаблона или объясняете руководству плюсы и минусы покупки промежуточного программного обеспечения, залогом успеха являются ваши навыки общения.

Объективно измерить степень влияния архитектора на проект довольно сложно, но одно совершенно ясно: если разработчики постоянно игнорируют указания архитектора, а руководство не придает значения его рекомендациям, «правильность» действий архитектора никак не повлияет на развитие его карьеры. Опытные архитекторы понимают, что они должны «продвигать» свои идеи, а эта задача требует умения эффективно общаться.

На тему межличностного общения написано множество книг, но я хотел бы предложить вашему вниманию один простой и практичный прием, который радикально повысит эффективность вашего общения и соответственно упрочит ваш успех в роли архитектора. В какой бы ситуации вам ни приходилось объяснять свою позицию сразу нескольким слушателям, встаньте. Неважно, где вы при этом находитесь – на официальном анализе проекта системы или неформальном обсуждении пары диаграмм. Встаньте – особенно если все остальные сидят.

Когда вы встаете, окружающие автоматически начинают воспринимать вас как авторитетного и уверенного в себе человека. Вы становитесь центром внимания. Вас будут реже перебивать. Все это окажет существенное влияние на обсуждение независимо от того, будут приняты ваши рекомендации или нет.

Также следует отметить, что стоящий человек более интенсивно использует жестикуляцию и мимику. Если вы общаетесь с группой из 10 и более человек, вставание поможет установить зрительный контакт со всей аудиторией. Зрительный контакт, жестикуляция, мимика и другие визуальные элементы играют важную роль в общении. Кроме того, положение стоя изменяет тональность и громкость голоса, а также темп речи: вы начинаете говорить так, чтобы вас было слышно в большом помещении, делая паузы для выделения важных моментов. Все эти элементы вносят существенный вклад в эффективность общения.

Хотите повысить эффективность передачи своих идей в процессе общения более чем вдвое? Это очень просто: встаньте.

Уди Дахан (Udi Dahan) – Шаман от Программирования; его заслуги были признаны корпорацией Microsoft и в течение трех лет оценивались престижным званием «Наиболее ценного специалиста» (Most Valuable Professional) в области архитектуры решений. В качестве советника по коммуникационным технологиям Уди работает с компанией Microsoft над WCF, WF и Oslo. Он также участвует в работе консультативных комитетов Microsoft Software Factories Initiative и проекта Prism группы Patterns & Practices. Он оказывает консультации в области современных архитектур и обучает клиентов по всему миру. Его основной специальностью является проектирование сервис-ориентированных, масштабируемых и безопасных архитектур на базе платформы .NET.

Вы ведете переговоры чаще, чем вам кажется

Майкл Найгард

Все мы попадали в «бюджетектурные» переделки, когда разумные технологические решения «хоронятся» ради экономии. Разговор проходит примерно так:

«Нам действительно так необходимы X?» – спрашивает спонсор проекта.

На место X можно подставить практически всё жизненно необходимое для работы системы: лицензии на программные продукты, избыточные серверы, внешние резервные копии или источники питания. Вопрос всегда задается отеческим тоном, словно вы спускаете все карманные деньги на комиксы и жвачку, тогда как серьезным взрослым людям нужно думать о покупке новых ведер, в которых они будут таскать свою будущую прибыль.

Правильный ответ на этот вопрос звучит так: «Да. Абсолютно необходимы». Но *так* почему-то почти никто не отвечает.

В конце концов, у нас техническое образование, а любая техническая дисциплина – это искусство компромисса. Понятно, что экзотика вроде источников питания никому не будет нужна, если поставить в центре обработки данных несколько беличьих колес и запустить в них практикантов. И вместо того чтобы сказать «Да, абсолютно необходимы», мы говорим что-то вроде: «Вообще-то без второго сервера можно обойтись, если вы согласны смириться с простоями из-за профилактики и при каждом сбое памяти. Хотя если мы купим память с автоматическим контролем четности, то и эту проблему можно обойти. Так что остаются только сбои операционной системы в среднем через каждые 3,9 дня, а значит, по ночам сервер придется перезагружать, но это вполне могут делать практиканты, когда устанут крутить колеса».

И все это может быть совершенно справедливо, но говорить так ни в коем случае не следует. Спонсор перестает вас слушать уже после слов «вообще-то».

Проблема в том, что вы рассматриваете происходящее с технической точки зрения, а ваш спонсор четко понимает, что он ведет переговоры. Вы занимаетесь совместным поиском решения, тогда как он проводит тактический маневр типа «выйдет/не выйдет». А в любых переговорах ни в коем случае не следует делать уступки по первому требованию. Правильный ответ на вопрос «Нам действительно так необходимы X?» звучит примерно так:

«Без второго сервера вся система будет «падать» примерно три раза в день, особенно в периоды максимальной нагрузки и при демонстрации на собрании совета директоров. На самом деле нам нужно четыре сервера, чтобы одна независимая пара обеспечивала сохранение 100-процентной работоспособности, даже если другая пара неожиданно перестанет работать».

Конечно, вы прекрасно знаете, что третий и четвертый серверы на самом деле не нужны. Это тактический гамбит, который заставит вашего спонсора перевести разговор на другую тему. Вы повышаете ставку и показываете, что система и так работает на минимальной, рискованной конфигурации. Кроме того, если вам каким-то чудом удастся получить дополнительные серверы, вы всегда можете передать один под тестирование (чтобы среда тестирования полностью соответствовала рабочей среде), а из другого получится отличная машина для сборки.

Биография автора приведена на стр. 31.

Сделать наспех и сбежать – преступление

Никлас Нильссон

Время близится к вечеру. Команда дружно корпит над новой функциональностью, запланированной для текущей итерации; кажется, даже воздух в комнате пульсирует в рабочем ритме. Однако Джон немного спешит: его ждет свидание. Впрочем, он успевает дописать свою часть кода, компилирует ее, регистрирует в системе управления исходным кодом – и поспешно уходит. Несколько минут спустя загорается «красный свет»: сборка приложения нарушена. У Джона не было времени на автоматизированные тесты, поэтому он поступил по принципу «сделать наспех и сбежать», из-за чего застопорилась работа всей команды.

Ситуация изменилась – рабочий ритм сбился. Теперь все знают, что при обновлении кода из системы контроля версий неработоспособный код окажется и на их локальных компьютерах, а поскольку для подготовки к предстоящей демонстрации команде предстоит интегрировать много кода, это становится серьезным препятствием. По сути дела, Джон поставил команде подножку – ведь интеграция станет возможна только после того, как кто-то потратит свое время на отмену его изменений.

Такая ситуация возникает до обидного часто. Сделать наспех и сбежать – преступление, поскольку в результате нарушается нормальный ход работы. Это печально распространенный среди разработчиков способ сэкономить немного времени лично для себя, в итоге потратив впустую чужое время, что служит проявлением прямого неуважения к другим людям. И все же это происходит повсеместно. Почему? Обычно потому, что полноценная сборка системы или проведение тестов занимает слишком много времени.

Здесь в игру вступаете вы – архитектор. Вы тратите массу усилий на создание гибкой архитектуры, обучаете разработчиков гибким методам разра-

ботки (таким как разработка через тестирование) и обеспечиваете наличие сервера для непрерывной интеграции. Кроме того, вам необходимо сформировать культуру, правила которой запрещают понапрасну расходовать чужое рабочее время. Для этого помимо прочего нужно позаботиться о создании качественной инфраструктуры автоматизированного тестирования, поскольку она способна изменить поведение разработчиков. Если тесты выполняются быстро, разработчики будут проводить их чаще, что уже само по себе хорошо, но кроме этого они не будут оставлять своим коллегам нерабочий код. Если тесты зависят от внешних систем или для их выполнения необходимы обращения к базе данных, измените их так, чтобы они могли выполняться локально с заглушками (или хотя бы с базой данных, хранящейся в памяти), и пусть на сборочном сервере эти тесты выполняются медленно. Не заставляйте людей дожидаться, пока компьютер выполнит свою работу, иначе они начинают искать лазейки, которые в результате создают проблемы для всех остальных.

Не жалейте времени на то, чтобы обеспечить быструю работу с системой. Это повышает эффективность труда, устраняет поводы для уклонения сотрудников от работы и в конечном итоге способствует ускорению процесса разработки. Создавайте суррогаты и симуляторы, устраняйте зависимости, делите систему на меньшие модули – делайте что угодно, чтобы у ваших коллег не было даже тени искушения последовать принципу «сделать наспех и сбежать».

Никлас Нильссон (Niclas Nilsson) – консультант и преподаватель в области разработки программного обеспечения, а также писатель, глубоко увлеченный искусством программирования, ценитель хорошей архитектуры и дизайна. Является одним из соучредителей factor10 и редактором материалов в сообществе архитектуров ПО на сайте InfoQ.

Архитектор должен быть практиком

Джон Дэвис

Хороший архитектор должен подавать личный пример другим. Он должен быть способен заменить любого члена своей команды и выполнить любую работу – от прокладки сети и настройки процесса сборки до написания модульных тестов и выполнения тестов производительности. Без хорошего понимания всего диапазона технологий архитектор мало чем отличается от обычного руководителя проекта. Члены команды могут обладать более глубокими знаниями в своих узких областях – это совершенно нормально, – но вряд ли они смогут доверять своему архитектору, если тот не разбирается в используемых технологиях. Как уже было сказано, архитектор – это интерфейс между технической командой и бизнесом, а значит, он должен понимать все технические аспекты, чтобы играть роль представителя команды перед бизнес-руководством, не обращаясь постоянно за помощью. Из тех же соображений архитектор должен понимать деловые аспекты организации, чтобы успешно привести разработчиков к цели – удовлетворению коммерческих интересов компании.

Работа архитектора сродни работе пилота самолета: он может не выглядеть занятым, но в действительности использует десятилетия накопленного опыта для постоянного наблюдения за ситуацией и немедленно вмешивается при возникновении нештатной ситуации. Руководитель проекта (второй пилот) обеспечивает повседневное управление, избавляя архитектора от рутины и управления персоналом. В конечном итоге архитектор должен отвечать за качество продукта и его своевременную сдачу. Эти задачи трудно решить без личного авторитета, который играет чрезвычайно важную роль в успехе любого проекта.

Лучший способ обучаться – наблюдать за другими; именно так мы учимся в детстве. Хороший архитектор должен уметь выявить проблему, собрать

команду и, не занимаясь поисками виновных, объяснить суть проблемы, а затем предложить элегантное решение или обходной путь. При этом архитектор может попросить у команды помощи, нисколько не теряя авторитета. Разработчики должны ощущать свой вклад в решение задачи, но архитектор при этом направляет ход обсуждения и определяет правильный подход.

Архитекторам следует присоединяться к команде уже на самых ранних стадиях проекта; они должны не сидеть в башне из слоновой кости, указывая оттуда путь вперед, а работать «в поле» вместе со всеми остальными. Вопросы выбора стратегического направления или технологии не следует превращать в самостоятельные исследования или новые проекты – к ним надо подходить прагматически, разыскивая ответ в ходе практической работы или обращаясь за советом к коллегам-архитекторам (все хорошие архитекторы знают друг друга).

Хороший архитектор обязан на уровне эксперта владеть как минимум одним из инструментов своей профессии, например интегрированной средой разработки (IDE); помните, что архитектор должен быть практиком. Вполне логично, что архитектору ПО следует хорошо знать IDE, архитектору баз данных – инструментарий построения диаграмм «сущность–связь» (ER-диаграмм), а информационному архитектору – инструменты XML-моделирования. Однако ведущий архитектор обязан уметь применять инструменты всех уровней, от контроля сетевого трафика с использованием Wireshark до моделирования сложных финансовых сообщений в XMLSpy, – для него не существует слишком низких или слишком высоких уровней.

Как правило, архитектор приходит с хорошим резюме и впечатляющим прошлым. Этим обычно несложно произвести впечатление на руководство и технический персонал, но без демонстрации своих умений на практике он вряд ли сумеет завоевать уважение команды. В такой ситуации команда будет испытывать трудности с обучением, а ее члены вряд ли смогут справиться с той задачей, для решения которой их наняли.

Джон Дэвис (John Davies) в настоящее время является ведущим архитектором в фирме Revolution Money (США). Недавно основал новую компанию Incept5.

Учитесь у архитекторов зданий

Кейт Брайтуэйт

> *Архитектура – социальный акт и материальный театр человеческой активности.*
>
> *Спиро Костоф (Spiro Kostof)*

Сколько найдется архитекторов программного обеспечения, считающих свою роль исключительно (или в первую очередь) технической? Разве не должны они в действительности быть посредниками и арбитрами для воюющих фракций среди заинтересованных в проекте сторон? Сколькие из них рассматривают свою работу с чисто интеллектуальной точки зрения, не уделяя должного внимания человеческому фактору?

> *Архитектор становится великим благодаря не столько уму, сколько чистому, чуткому сердцу.*
>
> *Фрэнк Ллойд Райт (Frank Lloyd Wright)*

Что в большей степени присуще архитекторам вашей организации: необузданная интеллектуальная мощь и способность помнить мельчайшие технические детали или хороший вкус, изысканность и душевная щедрость? В какой атмосфере предпочли бы работать вы сами?

> *Врач может похоронить свою ошибку, архитектор – разве что обсадить стены плющом.*
>
> *Фрэнк Ллойд Райт*

Не является ли «сопровождение унаследованных систем» лишь подрезкой этого плюща? Хватит ли вам как архитектору силы духа уничтожить собственное неудачное творение? Или вы предпочтете просто прикрыть его? Райт сказал также, что лучшим другом архитектора является кувалда. А что вы разбили ею за последнее время?

> *Архитекторы не только верят, что сидят по правую руку от Бога, но и считают, что если Бог встанет, то они займут его место.*
>
> *Карен Мойер (Karen Moyer)*

Замените «Бог» на «заказчик».

> *В архитектуре, как и во всех остальных прикладных видах искусства, конечная цель управляет действием. Конечная цель – хорошо построенное здание. Такое здание обладает тремя свойствами: Удобство, Прочность и Эстетичность.*
>
> *Генри Уоттон (Henry Watton)*

Когда вам в последний раз попадался на глаза программный продукт, архитектура которого доставила вам эстетическое удовольствие? А вы сами стремитесь к тому, чтобы ваши работы были эстетичными?

> *Человек, не являющийся великим скульптором или художником, не может быть архитектором. Если он не скульптор, не художник, то сможет быть только строителем.*
>
> *Джон Раскин (John Ruskin)*

Занимает ли эстетика достойное место в вашей архитектуре? Движет ли вами в ходе объединения компонентов при построении систем художественный интерес к формам и текстурам, скульптурное чувство баланса и стремление к передаче движения либо ощущение важности отрицательного пространства?

И наконец, высказывание, не требующее комментариев, – верное средство от самого разрушительного синдрома в работе архитектора:

> *Это выглядит фантастическим парадоксом, но тем не менее является важнейшей истиной: абсолютно совершенная архитектура не может быть действительно великой.*
>
> *Джон Раскин*

Биография автора приведена на стр. 35.