ГЛАВА 1

Знакомство с RESTful Web API

> Используйте глобальную досягаемость, чтобы решать задачи, о которых даже не думали, для людей, с которыми никогда не встречались.
>
> *Принцип RESTful Web API*

Во введении я акцентировал внимание на специфическом названии книги. Здесь мы займемся изучением принципов, стоящих за *RESTful Web API*, а также поговорим о том, почему я считаю важным именно такое название и в чем его смысл.

Сначала я немного поясню, что значит фраза «RESTful Web API» и почему я решил использовать столь странное выражение. Затем мы поговорим о том, что я считаю основной движущей технологией, позволяющей создавать отказоустойчивые и надежные сервисы в открытом Интернете, а именно о *гипермедиа*. Завершит главу краткий раздел, посвященный изучению нескольких *общих принципов* реализации и применения интерфейсов сервисов на основе REST. На основе этих принципов выбирались и описывались паттерны и рецепты, собранные в книге.

Реализации на основе гипермедиа опираются на три основных элемента: сообщения, действия и словари (рис. 1.1). В гипермедиарешениях сообщения передаются с помощью распространенных форматов, таких как HTML, Collection + JSON и SIREN. Эти сообщения содержат информацию, основанную на общем словаре предметной области, например PSD2 для банковской сферы, ACORD для страхования или FIHR для здравоохранения. Эти же сообщения включают четко определенные действия вроде `save`, `share`, `approve` и т. д.

Оперируя этими тремя принципами, я постараюсь сформировать у вас представление о том, как мы создаем и используем сервисы сегодня. Я покажу, как, слегка изменяя точку взгляда и подход, можно обновлять дизайн и реализацию этих сервисов, чтобы повысить их юзабилити, снизить затраты на их создание и доступ к ним, а также расширить для производителей и потребителей сервисов возможности создавать и поддерживать жизнеспособные, использующие API бизнесы, даже когда некоторые сервисы, от которых мы зависим, оказываются ненадежными или недоступными.

Рис. 1.1. Элементы гипермедиа

Для начала разберем смысл названия книги.

Что такое RESTful Web API

Я уже несколько лет использую фразу «RESTful Web API» в статьях, презентациях и обучающих материалах. Мы с Леонардом Ричардсоном в 2013 году выпустили целую книгу (https://oreil.ly/a5Pwl). Иногда этот термин вызывает недопонимание и даже скептицизм, но почти всегда — любопытство. Что вообще эти три слова делают вместе? Что означает комбинация их сути в целом? Чтобы ответить на эти вопросы, для начала стоит прояснить значение каждого термина по отдельности.

В этом разделе мы разберем следующие определения.

REST по Филдингу

Архитектурный стиль, который подчеркивает масштабируемость взаимодействий компонентов, их независимое развертывание и универсальность интерфейсов.

Веб по Тиму Бернерсу-Ли

Всемирная сеть (World Wide Web) была представлена как универсальная связанная система обработки информации, в которой первостепенное значение имеют универсальность и портативность.

Позднее связывание по Алану Кею

Схема проектирования, позволяющая создавать системы, которые можно безопасно менять во время их работы.

REST по Филдингу

В далеком 1998 году Рой Т. Филдинг (Roy T. Fielding, https://oreil.ly/cOvy4) провел в Microsoft презентацию, на которой объяснил свой принцип *передачи состояния представления* (Representational State Transfer, сегодня известен как REST, https://oreil.ly/DqhFK). В этой презентации и докторской диссертации, которую он защитил двумя годами позже (*Architectural Styles and the Design of Network-based Software Architectures*, https://oreil.ly/78U8r), Филдинг продвигал идею, согласно которой для сетевых реализаций есть уникальный набор программных архитектур и один из шести выделенных им стилей, REST, особенно хорошо подходит для Всемирной паутины.

Спустя несколько лет я услышал распространенную нынче фразу: «Часто цитируют, никогда не читают». Этот острый комментарий хорошо подходит к диссертации Филдинга (2000). Я призываю всех, кто работает над созданием или обслуживанием ПО для веб-среды, уделить время и внимательно прочитать всю диссертацию, а не только ее пресловутую главу 5 под названием Representational State Transfer (https://oreil.ly/PdLrH). Описанное Филдингом более 20 лет назад разделение общих стилей на категории оказалось корректным, и многие из этих стилей позднее стали известны как gRPC, GraphQL, событийно-ориентированные, контейнеры и пр.

Использованный Филдингом метод определения желаемых свойств на уровне системы (доступность, производительность, простота, изменяемость и пр.), а также рекомендуемый им набор ограничений (клиент — сервер, отсутствие состояния, кэшируемость и т. д.) для *индуцирования* этих свойств и по прошествии более чем 20 лет оказывается полезным подходом к восприятию проектирования ПО, от которого требуется долгосрочная стабильность.

Хороший вариант обобщения предложенного Филдингом стиля REST содержится в самой диссертации (https://oreil.ly/tl6U6):

> *«REST предоставляет набор архитектурных ограничений, который в случае применения целиком выделяет масштабируемость взаимодействий компонентов, их независимое развертывание, универсальность интерфейсов и промежуточные компоненты для сокращения задержки взаимодействий, повышения безопасности и инкапсуляции устаревших систем».*

Для этой книги были отобраны такие рецепты, которые позволят проектировать и создавать сервисы, демонстрирующие многие из *ключевых архитектурных свойств* (https://oreil.ly/J0Fc2), выявленных Филдингом. Далее приведены список этих свойств и краткое описание их назначения и способов применения.

Производительность

Производительность основанного на сетевых взаимодействиях решения привязана к физическим ограничениям сети — объему проходящего трафика

(throughput), пропускной способности (bandwidth), издержкам и т. д., а также способам повышения производительности на стороне пользователя, таким как задержка запросов и возможность сократить время завершения с помощью параллельных запросов.

Масштабируемость

Возможность архитектуры поддерживать большое число компонентов или взаимодействий между компонентами внутри действующей конфигурации.

Простота

Для упрощения решения в первую очередь используют принцип разделения обязанностей на этапе распределения функциональности внутри компонентов, а также принцип универсальности интерфейсов.

Изменяемость

Простота, с которой можно вносить изменения в архитектуру в виде развития, расширения, персонализированной настройки и повторного использования.

Видимость

Возможность компонента мониторить взаимодействие между двумя другими компонентами или делать его опосредованным с помощью кэшей, прокси-серверов или прочих посредников.

Портативность

Возможность выполнять одно и то же ПО в разных средах, включая возможность безопасного перемещения кода (например, JavaScript), а также данных между системами среды выполнения (например, Linux, Windows, macOS и т. д.).

Надежность

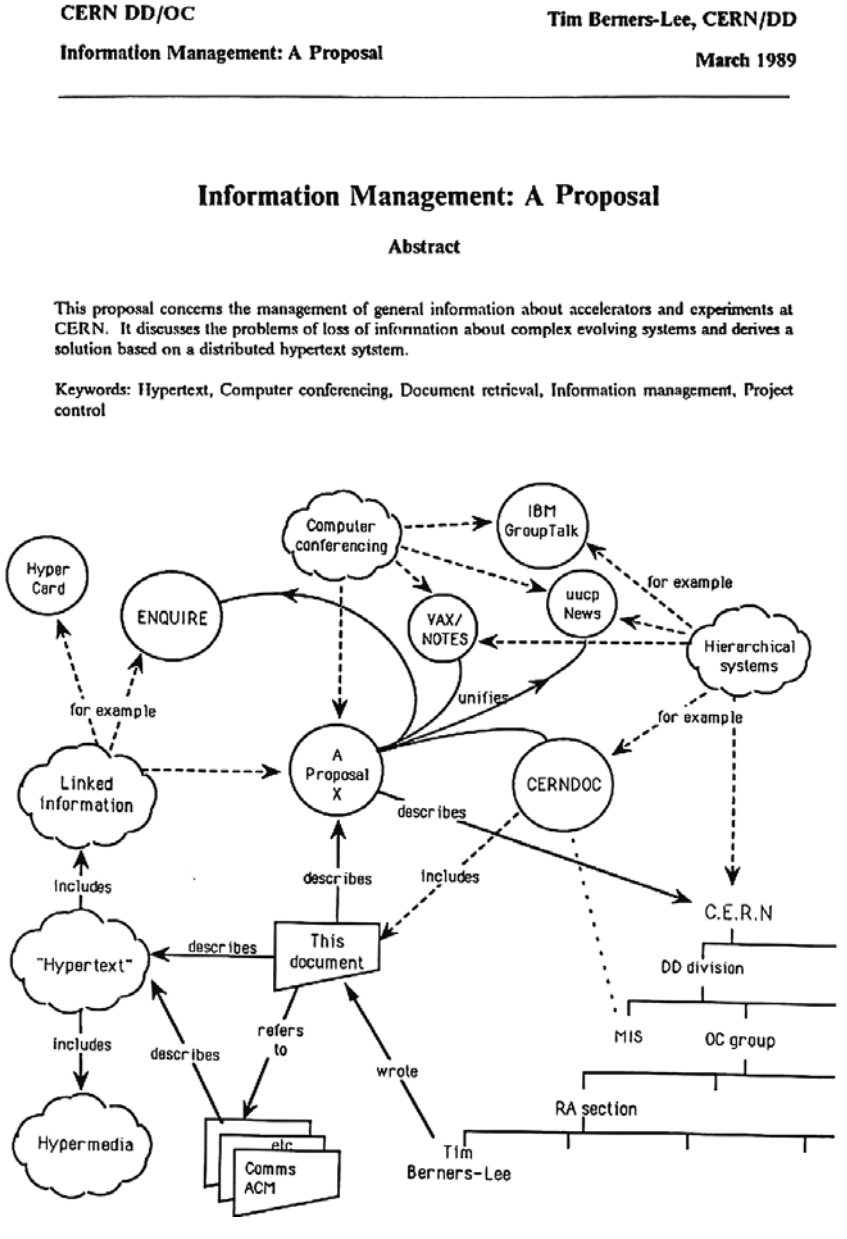
Степень уязвимости реализации к общесистемным сбоям в результате сбоя одного компонента (машины или сервиса) внутри сети.

Ключевая причина, по которой мы будем использовать в рецептах архитектурные принципы Филдинга, состоит в том, что они помогают получить реализации, которые можно масштабировать и безопасно изменять на больших дистанциях как в пространстве, так и во времени.

Веб по Тиму Бернерсу-Ли

Работа Филдинга опирается на труды другого пионера в онлайн-мире, сэра Тимоти Бернерса-Ли (https://oreil.ly/GpDkH). Более чем за десять лет до написания Филдингом диссертации Бернерс-Ли опубликовал 16-страничный документ *Information Management: A Proposal* (1989–1990 годы, https://oreil.ly/ZE5qk). В нем он предложил на тот момент уникальное решение для повышения эффективности

хранения и извлечения информации в лаборатории физики CERN, где работал. Эту свою идею Бернерс-Ли назвал Всемирной паутиной (World Wide Web) (рис. 1.2).



CERN DD/OC

Information Management: A Proposal

Tim Berners-Lee, CERN/DD

March 1989

Information Management: A Proposal

Abstract

This proposal concerns the management of general information about accelerators and experiments at CERN. It discusses the problems of loss of information about complex evolving systems and derives a solution based on a distributed hypertext sytstem.

Keywords: Hypertext, Computer conferencing, Document retrieval, Information management, Project control

Рис. 1.2. Предложенная Бернерсом-Ли схема Всемирной паутины (1989)

Всемирная паутина (WWW) стала развитием идеи Теда Нельсона (Ted Nelson, https://oreil.ly/mkylW), который придумал термин *«гипертекст»*, соединяя связанные документы *ссылками*, а потом и *формами*, в которые пользователям предлагалось вводить данные, отправляемые на серверы по всему миру. Эти серверы легко и быстро настраивались с помощью бесплатного ПО, установленного на совместно используемых настольных компьютерах. Структура WWW соответствовала правилу максимальной простоты, которое гласит, что для решения задачи нужно задействовать самую простую из подходящих технологий. Иными словами, целью было сделать общее решение максимально простым (но не проще). Позже эта концепция была зафиксирована как документ W3C (https://oreil.ly/h0B2Q) с тем же именем.

Такой подход установил низкий порог входа для всех, кто хотел присоединиться к сообществу WWW, и помог вызвать его взрывную популярность в 1990-х и начале 2000-х годов.

ЦЕЛЬ ВСЕМИРНОЙ ПАУТИНЫ

В документе, описывавшем систему, которая позднее стала веб-средой (https://oreil.ly/UIHJ7), Бернерс-Ли написал: «Нам нужно работать над созданием универсальной связанной информационной системы, в которой приоритетом будут универсальность и портативность».

В WWW любой документ можно было отредактировать, чтобы он ссылался (указывал) на другой документ в Сети. При этом никаких дополнительных процессов проделывать ни на одной из сторон этой связи не приходилось. По сути, люди могли свободно создавать собственные соединения, собирать свои избранные документы и создавать личный контент, не получая разрешения от кого бы то ни было. Создание всего этого контента становилось возможным за счет использования ссылок и форм внутри страниц, которые позволяли выстраивать уникальные пути и взаимодействия — такие, о которых авторы исходного документа (того, который подключался) ничего не знали.

Два аспекта WWW — правило максимальной простоты и свободу создавать собственные соединения — мы будем применять во всех приведенных в книге рецептах.

Позднее связывание по Алану Кею

Еще один важный принцип создания надежных и отказоустойчивых сервисов, способных *существовать в веб-среде*, был придуман Аланом Кеем (Alan Kay, https://oreil.ly/CbNWf) — ученым из США, работавшим в компьютерной сфере. Именно ему приписывают популяризацию термина «объектно-ориентированное программирование» в 1990-е годы.

АЛАН КЕЙ ПРО ООП

Объясняя свое видение *объектно-ориентированного программирования* (ООП) в сообщении для списка рассылки в 2003 году (https://oreil.ly/vN8e4), Кей написал: «Для меня ООП означает только: 1) обмен сообщениями, 2) локальное сохранение, защиту и сокрытие состояния объекта, 3) позднее связывание всего».

В 2019 году Кертис По (Curtis Poe, https://oreil.ly/pfi4t) написал статью, в которой разобрал данную Аланом Кеем трактовку ООП и, помимо прочего, указал: «Позднее связывание важно, так как Кей утверждает, что оно позволяет не прибегать слишком рано к *одному истинному способу* решения проблемы, тем самым упрощая изменение этих решений, но при этом дает возможность создавать системы, которые вы сможете изменять *в процессе их работы*!» (выделение в тексте соответствует оригиналу).

Более непосредственный анализ связей между REST по Филдингу и ООП по Алану Кею изложен в моей статье 2015 года *The Vision of Kay and Fielding: Growable Systems that Last for Decades* (https://oreil.ly/TLvzq).

Аналогично тому, как Кей представлял программирование с помощью ООП, веб-среда, то есть сам Интернет, *работает непрерывно*. И сервисы, которые мы устанавливаем на подключенную к Интернету машину, по факту изменяют систему в процессе ее работы. Это нужно иметь в виду при создании сервисов для веб-среды.

Именно тот факт, что позднее связывание позволяет изменять системы во время их работы, мы будем использовать в качестве опорного принципа в рецептах книги.

Итак, в текущем разделе мы рассмотрели:

- использование представлений Филдинга о проектировании систем, допускающих безопасное масштабирование и изменение с течением времени;
- применение предложенного Бернерсом-Ли правила максимальной простоты и принципа снижения барьера входа, чтобы позволить любому подключенному к сети легко соединяться с любым другим ее участником;
- использование позднего связывания по Кею для упрощения изменения частей системы в процессе ее работы.

Важной техникой, которая поможет нам все это реализовать, является *гипермедиа*.

Почему гипермедиа

По моему опыту, концепция гипермедиа находится на пересечении нескольких важных инструментов и технологий, которые положительно повлияли на формирование нашего информационного общества. И я думаю, что она поможет повысить доступность и юзабилити сервисов в Интернете в целом.

В этом разделе мы поговорим:

- об эпохе гипермедиа;
- о ценности сообщений;
- о важности словарей;
- о магических строках Ричардсона.

История гипермедиа насчитывает почти 100 лет и прослеживается в работах XX века по психологии, взаимодействию человека и компьютера, а также в теории информации. Она лежит в основе схемы Всемирной паутины Бернерса-Ли (см. раздел «Веб по Тиму Бернерсу-Ли» ранее в этой главе) и может стать основой нашей сети API. Именно поэтому ее следует изучить подробнее. Для начала давайте определим, что такое гипермедиа и приложения на основе гипермедиа.

Гипермедиа

Термины «гипертекст» и «гипермедиа» придумал Тед Нельсон в начале 1950-х. Он использовал их в своей работе 1965 года, посвященной средствам ACM, *Complex Information Processing: A File Structure for the Complex, the Changing and the Indeterminate* (https://oreil.ly/bqY3B). Как писал в 2008 году Томас Исаковиц (Tomas Isakowitz), изначальная структура гипертекстовой системы (https://oreil.ly/1Ggxv) «состоит из *узлов*, которые содержат информацию, и *ссылок*, которые представляют связи между этими узлами». Гипермедиасистемы акцентируются на *связях* между их элементами.

По сути, гипермедиа дает возможность связывать между собой отдельные узлы, также называемые *ресурсами*, например документы, изображения, сервисы и даже фрагменты текста в документе. В сети эта связь реализуется с помощью универсальных идентификаторов ресурсов (universal resource identifiers, URI, https://oreil.ly/ehJpL). Когда такое соединение подразумевает возможность передачи данных, ссылки выражаются в виде *форм*, которые также можно предлагать пользователям или запрограммированным машинам для ввода информации. К примеру, HTML поддерживает создание ссылок и форм с помощью тегов `<a>`,

`<img>`, `<form>` и др. Существует несколько форматов, которые поддерживают ссылки и формы гипермедиа.

Эти элементы гипермедиа также можно возвращать вместе с результатами запроса. Возможность передавать ссылки и формы в ответах позволяет клиентским приложениям выбирать и активировать эти гипермедиаэлементы, чтобы обеспечить прогресс выполнения. Это дает возможность создавать сетевые решения, состоящие исключительно из серии ссылок и форм (вместе с возвращаемыми данными), которые при переходе по ним обеспечивают решение предусмотренной задачи (например, вычисление результатов, получение обновления и сохранение данных в удаленном расположении и т. д.).

Ссылки и формы обеспечивают универсальность интерфейсов (например, использование гипермедиадокументов по HTTP), которая лежит в основе гипермедиаприложений. Клиентские приложения на основе этой технологии, такие как HTML-браузеры, могут задействовать эту универсальность для поддержания широкого спектра новых приложений даже без изменения или обновления своего исходного кода. Мы просто движемся от одного решения к другому, проходя по ссылкам или вводя их вручную, и применяем одно и то же установленное клиентское приложение для чтения новостей, обновления списков дел, онлайн-игр и т. д.

Рецепты в этой книге основываются на структуре гипермедиа, чтобы выступать решением не только для приложений, управляемых людьми, вроде HTML-браузеров, но и для тех, которыми управляют машины. Это особенно актуально для клиентов, которые для доступа к сервисам сети опираются на API. В главе 4 я познакомлю вас с приложением командной строки, которое позволяет быстро создавать скрипты для клиентских приложений на основе гипермедиа, не меняя кода этих приложений (см. приложение Г).

Эпоха гипермедиа

Идея информационного соединения людей существовала давно. В 1930-х бельгиец Поль Отле (Paul Otlet, https://oreil.ly/sxkh5) представлял себе машину, которая позволит людям искать и составлять индивидуальные комбинации аудио-, видео- и текстового контента, а также получать к ним доступ из любой точки мира. Прошло почти 100 лет, и стриминговая революция наконец произошла.

Поль Отле

Схема подключения домашних машин к различным источникам новостей, развлечений и информации была представлена Отле в 1940 году (рис. 1.3). Эта схема, которую он назвал Всемирной сетью (World Wide Network), очень похожа на схемы Теда Нельсона (будет показана чуть позже) и Тима Бернерса-Ли (см. раздел «Веб по Тиму Бернерсу-Ли» ранее).

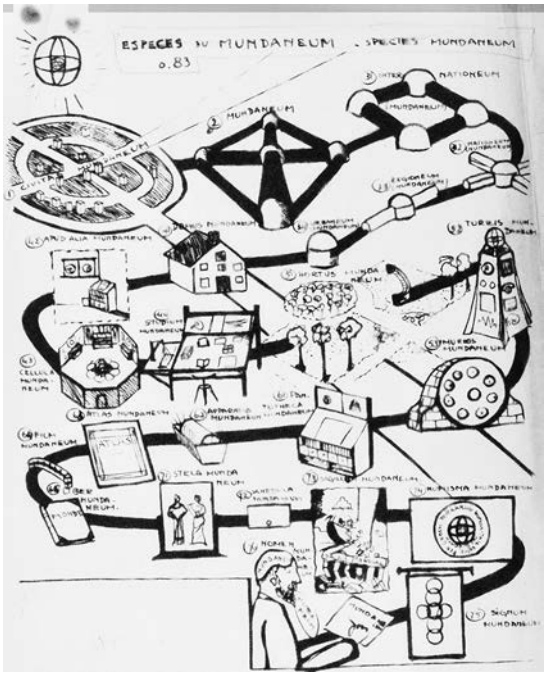


Рис. 1.3. Схема Всемирной сети Отле (1940 год)

Вэнивар Буш

Работая менеджером в проекте «Манхэттен», Вэнивар Буш (Vannevar Bush, https://oreil.ly/q3xPC) заметил, что, когда отдельные специалисты объединялись для работы над задачами в творческой обстановке, они зачастую обменивались идеями, переключаясь с одной на другую и создавая новые связи между научными трудами. В июле 1945 года он опубликовал свои наблюдения в статье *As We May Think* (https://oreil.ly/V5U0l), где описал похожую на предложенную Отле рабочую станцию, в которой использовались микрофиша и указывающее устройство, крепившееся на голове читателя.

Дуглас Энгельбарт

Прочитав статью Буша, молодой офицер, служивший тогда в Восточной Азии, задумался, как воплотить описанное им устройство в жизнь. На это ушло почти 20 лет, но в 1968 году тот самый офицер, Дуглас Энгельбарт (Douglas Engelbart, https://oreil.ly/D5eaV), провел презентацию, которая сегодня известна как «Мать всех демонстраций» (*The Mother of All Demos*, https://oreil.ly/a38AX), представив ряд идей, над которыми работал со своей командой. В этой презентации был показан первый на тот момент «интерактивный компьютер», который позволял оператору

с помощью указывающего устройства выделять текст и щелкать по ссылкам для перехода. Для этой демонстрации Энгельбарту потребовалось изобрести мышь.

МАТЬ ВСЕХ ДЕМОНСТРАЦИЙ

Презентация Энгельбарта, проведенная им более 50 лет назад, в декабре 1968 года, на конференции по мейнфреймам в Сан-Франциско, заложила стандарт для демонстраций из Кремниевой долины, которые мы видим сегодня. Энгельбарт выступал в течение 90 минут, сидя в специально разработанном кресле Имса (https://oreil.ly/yvkYl, прототип современных кресел Аэрона), работая с настроенной под себя клавиатурой, мышью и кнопочным пультом. Все это время он спокойным голосом описывал свои действия через гарнитуру. Энгельбарт показал первый интерактивный компьютерный экран, продемонстрировал такие возможности, как «вырезать — копировать — вставить», гиперссылки и мультикурсорное редактирование, связь с коллегами, находящимися за сотни миль, по видеосвязи в формате «картинка в картинке», контроль версий и несколько других концепций, которые вошли в обиход только спустя более десяти лет. Если вы не смотрели запись его выступления (https://oreil.ly/EycWe), настоятельно рекомендую.

Тед Нельсон

Современник Энгельбарта, Тед Нельсон, писал о потенциале персональных вычислительных устройств еще в 1965 году (https://oreil.ly/YlocI), используя им же придуманные термины «гиперссылки», «гипертекст», «гиперданные» и «гипермедиа». К 1974 году его книга *Computer Lib/Dream Machines* (Tempus Books) сформировала представление о мире, заполненном личными электронными устройствами, связанными друг с другом через Интернет. В это же время Алан Кей (см. раздел «Позднее связывание по Алану Кею» ранее в этой главе) описал устройство *Dynabook* (https://oreil.ly/wiNEO), очень похожее на современные компактные ноутбуки и планшеты.

Все эти первые изыскания способов связи и обмена информацией несли в себе одну основную идею, заключавшуюся в том, что глобальное соединение расширит возможности людей и повысит уровень креативности и инноваций. К концу 1980-х Тим Бернерс-Ли собрал успешную систему, которая включала в себя все идеи его предшественников. Реализованная им Всемирная паутина сделала связывание страниц документов безопасным, простым и масштабируемым.

В этом и заключается суть использования API сервисов — определение связей между разными элементами сети для нахождения новых решений.

Джеймс Дж. Гибсон

Примерно в то же время, когда Тед Нельсон знакомил мир с термином *«гипертекст»*, еще один человек, психолог Джеймс Дж. Гибсон (James J. Gibson, https://oreil.ly/iQuJR), придумал другое, ставшее актуальным для этой сферы понятие.

В книге *The Senses Considered as Perceptual Systems* (https://oreil.ly/C5msH, Houghton-Mifflin), вышедшей в 1966 году, где он писал о том, как люди и другие животные воспринимают окружающий мир и взаимодействуют с ним, Гибсон ввел термин *«аффорданс»*. Вот его определение (https://oreil.ly/DFTSe):

> *«Аффордансы окружающей среды — это то, что она предлагает животному, что она для него обеспечивает и дает, вне зависимости от того, хорошо это или плохо».*

Аффордансы поддерживают взаимодействие между животными и средой в том же смысле, в каком гиперссылки Нельсона позволяют людям взаимодействовать с документами в сети. Современник Гибсона Дональд Норман (Donald Norman, https://oreil.ly/dFmp3) популяризовал термин *«аффорданс»* в своей книге *The Design of Everyday Things* (https://oreil.ly/KqmiE), вышедшей в 1988 году (Doubleday). Норман, считающийся дедушкой концепции человеко-компьютерного взаимодействия (Human-Computer Interaction, HCI, https://oreil.ly/jEdZx), использовал этот термин для определения способов, с помощью которых проектировщики ПО могут понимать такое взаимодействие и способствовать его достижению. Бóльшая часть того, что мы знаем о юзабилити программного обеспечения, зиждется на работах Нормана и других специалистов в этой области.

Гипермедиа опирается на аффордансы. Элементы гипермедиа (ссылки и формы) — это те составляющие веб-среды, которые делают *возможными*[1] дополнительные действия, такие как поиск существующих документов, отправка данных на сервер для хранения и пр. Гибсон и Норман представили психологический и социальный аспекты взаимодействия с компьютером, на которые мы будем опираться в приводимых далее рецептах. По этой причине вы заметите, что многие рецепты включают использование ссылок и форм, которые позволяют подстраивать состояние приложения для разных сервисов.

Ценность сообщений

Как мы видели ранее, Алан Кей представлял объектно-ориентированное программирование как концепцию, основанную на *передаче сообщений* (см. раздел «Позднее связывание по Алану Кею» ранее в этой главе). Тим Бернерс-Ли поддержал эту точку зрения, когда в 1992 году сформировал ориентированный на обмен сообщениями протокол гипертекстовой передачи (Hypertext Transfer Protocol, HTTP, https://oreil.ly/JEVu6), а годом позже помог определить формат сообщений в виде языка гипертекстовой разметки (Hypertext Markup Language, HTML, https://oreil.ly/gQGS4).

После создания протокола и формата для передачи обобщенных сообщений (вместо передачи локализованных *объектов* или *функций*) будущее Интернета было определено. Этот ориентированный на сообщения подход проще ограничивать, изменять со временем, и он предлагает более надежную платформу для

[1] От англ. afford, то есть выступают аффордансами. — *Примеч. пер.*

будущих доработок, таких как появление совершенно новых форматов (XML, JSON и пр.), и изменения использования самого протокола (документы, сайты, веб-приложения и т. д.).

НЕКОТОРЫЕ НЕ ОСОБО УДАЧНЫЕ ПРИМЕРЫ

Применяемый в HTTP подход с инкапсулированными сообщениями сделал возможными и не особо удачные инновации вроде Java Applets, Flash и XHTML. И хотя протокол HTTP создавался с учетом поддержки технологий наподобие названных провальных альтернатив ориентированному на сообщения HTML, эти форматы прожили недолго и их удаление из экосистемы не нанесло в долгосрочной перспективе вреда самому протоколу. Это лишний раз доказывает отказоустойчивость и гибкость подхода HTTP к взаимодействию на уровне приложений.

Ориентированные на сообщения онлайн-решения имеют параллели и с физическим миром. Колонии насекомых, например термитов и муравьев, известные отсутствием иерархии или руководящих особей, используют для взаимодействия систему сообщений на основе феромонов. Примерно в то же время, когда Нельсон говорил о гипермедиа, а Гибсон — об аффордансах, американский биолог и натуралист Э. О. Уилсон (E. O. Wilson, https://oreil.ly/0E5AF) и американский математик Уильям Боссерт (William Bossert) писали работу (https://oreil.ly/xVHFn), посвященную изучению колоний муравьев и использованию ими феромонов как инструмента управления крупными сложными сообществами.

С учетом всего этого вы наверняка не будете удивлены тем, что в основе всех рецептов, приведенных в книге, лежит обмен сообщениями с целью передачи информации между машинами.

Важность словарей

Подход на основе сообщений — это отличное решение в качестве платформы. Но даже универсальные форматы вроде HTML должны передавать информацию понятным способом. В 1998 году, примерно в то же время, когда Рой Филдинг разрабатывал свой подход REST для применения в сети (см. раздел «REST по Филдингу» ранее в этой главе), Питер Морвилл (Peter Morville, https://oreil.ly/G2Ekz) и Луи Розенфельд (Louis Rosenfeld, https://oreil.ly/G2LMX) выпустили книгу *Information Architecture for the World Wide Web* (O'Reilly). Считается, что именно она послужила началом движения по развитию *информационной архитектуры*. Профессор Дэн Клин (Dan Klyn, https://oreil.ly/UP6si) из Мичиганского университета объясняет это понятие, используя три ключевых элемента: *онтология* (конкретное значение), *таксономия* (упорядоченность частей) и *хореография* (правила взаимодействия частей).