

web 2.0

Tim O'Reilly

Переклад
Валерія Семенюка



blogoreader

Що таке Веб 2.0

Автор: **Тім О'Рейлі**

Опубліковано 18 жовтня 2005 року

Переклад Валерія Семенюка

Крах доткомів восени 2001 року став для веба поворотним пунктом. Багато хто вирішив, що феномен веба був дуже роздутим, хоча, насправді, "мильні бульбашки" і, як результат цього, падіння акцій - неминуче супроводять всі [технологічні революції](#). Падіння акцій традиційно відбувається тоді, коли нова технологія готова зайняти центральне місце на сцені. На хвилі піднімаються і шахраї, і ті, хто дійсно добився успіху, - і в якийсь момент приходиться розуміння, чим перші відрізняються від других.

Концепція **Веба 2.0** народилася на спільному мозговому штурмі видавництва **O'Reilly Media** і компанії **MediaLive International**. Веб-піонер і віце-президент O'Reilly **Дейл Дагерті (Dale Dougherty)** відмітив, що сам-то веб далекий від краху і навіть більш важливий, ніж раніше, раз вже вражаючи нові застосування і сайти з'являються із завидною регулярністю. Більше того, у фірм, що пережили колапс, було щось загальне. Можливо, в результаті доткомовського краху має сенс говорити про **Веб 2.0**. Ми вирішили, що так і є. Так народилася [Web 2.0 Conference](#).

За півтора роки термін "Веб 2.0" прижився (більше 9,5 млн. посилань в **Google**). Але відносно того питання, що він означає, то [в товаристві згоди немає](#). Одні прийняли нову концепцію, інші допускають, що це безсенсовий маркетинговий термін.

Я постараюся пояснити, що ми маємо на увазі, говорячи про **Веб-2.0**.

На нашій першій зустрічі, ми визначали **Веб-2.0**, відштовхуючись від конкретних прикладів.

Веб 1.0	Веб 2.0
DoubleClick	AdSense
Ofoto	Flickr
Akamai	BitTorrent
mp3.com	Napster
Britannica Online	Wikipedia
Персональні сайти	Блоги
Evite	upcoming.org і EVDB
Спекуляція доменними іменами	Пошукова оптимізація
Оплата реклами по кількості показів	Оплата реклами по кількості переходів
Добування даних з HTML	Веб-сервіси
Публікація	Співавторство
Системи керування контентом (CMS)	Wiki
Каталоги (таксономія)	Теги (фолксономія)
Утримування користувачів	Синдикація контенту

Список все збільшувався і збільшувався. Але чому одне застосування ми віднесли до **Вебу 1.0**, а інше - до категорії **Веб-2.0**? (Це важливе питання, оскільки **Веб-2.0**

став настільки популярним, що багато компаній використовують сьогодні цей термін в своєму маркетингу, часто навіть не розуміючи, що він означає. З іншого боку, це питання не таке вже й просте, тому що багато ласих до красивих термінів стартапів не мають ніякого відношення до **Вебу 2.0**, тоді як окремі застосування, які ми вважаємо **Вебом 2.0**, навіть веб-додатками, - ні, наприклад, *Napster* і *Bittorrent*.) Аналізуючи найбільш успішні проекти **Веба 1.0** і самі цікаві нові додатки, ми спробували виділити основні принципи **Веба 2.0**.

Веб як платформа

Як і багато важливих концепцій, **Веб-2.0** не має чітких меж. Це, швидше, центр тяжіння. Ви можете уявити собі **Веб-2.0** як множину правил і практичних рішень. Вони об'єднанні в деяку подобу сонячної системи, що складається з вузлів, кожен з яких побудований з урахуванням деяких або всіх описаних правил та знаходиться на певній відстані від центру.



На [рисунок](#) показана карта **Веба 2.0**, створена нами під час мозкового штурму на конференції FOO Camp. Її не можна назвати завершеною, але вона ілюструє багато ключових ідей **Веба 2.0**.

Наприклад, на першій конференції **Веб-2.0** у жовтні 2004 ми з **Джоном Баттелем** (John Battelle) озвучили попередній список правил у спільному виступі, який відкривав конференцію. І перше правило свідчить: "Веб-сервер як платформа". Звичайно, ще до нас про це навзрід говорив **Netscape**, який "згорів" в жаркій битві з **Microsoft**. Більше того, дві компанії з нашого списку **Веб-1.0** - **Akamai** і **DoubleClick** - також були серед першопроходців, що розглядали веб-сервер як платформу. Люди нечасто сприймають ці компанії як постачальників веб-сервісів, але, насправді, демонстрація реклами - це перший широко поширений веб-сервіс, перший широко поширений "**mash-up**" (якщо використовувати термін, що завоював останнім часом популярність). Кожен банер доставлявся користувачеві в результаті непомітної кооперації двох сайтів, що спільно формують сторінку для показу. **Akamai** також розглядав мережу як платформу, і навіть на нижчому рівні: забезпечував непомітне кешування і щоб понизити навантаження на сайти своїх клієнтів, побудував мережу доставки контенту. Послідовники **DoubleClick** і **Akamai** не тільки використовували напрацювання цих компаній, але пішли далі, глибше відчуваючи дійсну природу нової платформи. Обидві компанії можна вважати за піонерів **Веб-2.0**, хоча нижче ми побачимо, як більш повно реалізувати можливості веба за рахунок використання нових підходів. Давайте уважно розглянемо три приклади, щоб зрозуміти, чим, по суті, старі компанії відрізняються від нових.

Netscape проти Google

Якщо **Netscape** був флагманом **Веба 1.0**, то **Google**, звичайно, - загальноновизнаний флагман **Веба 2.0**. Отже давайте порівняємо самі компанії та їх позиціонування. **Netscape** твердив про "веб як платформу" в термінах старої софтверної парадигми: головним продуктом компанії був веб-браузер (настільне застосування), і стратегія **Netscape** полягала у використанні свого домінуючого положення на ринку браузерів для просування дорогих серверних продуктів. Контроль над стандартами відображення контенту і браузерних застосувань міг, в теорії, забезпечити **Netscape** таке ж місце, яке Microsoft завоювала на ринку ПК. Автомобілі колись рекламували як "безкінні екіпажі". Так само, відштовхуючись від знайомих концепцій, **Netscape** просував "вебтоп" на місце "десктопу", припускаючи підживлювання вебтопів даними і додатками від провайдерів контенту (які куплять сервери у Netscape). У результаті і веб-браузери, і веб-сервери перетворилися на щось буденне, а акцент перемістився "до вершини стека", тобто до веб-сервісів.

Google, навпаки, був веб-додатком від народження. Це сервіс, за доступ до якого прямо або побічно платили користувачі. Жодна із звичних пасток старої софтверної індустрії йому була не страшна. Замість запланованих релізів - постійне поліпшення продукту. Замість ліцензування або продажів - просто використання. Немає потреби піклуватися про портування ПЗ на інші платформи - все, що потрібне для запуску **Google**, - це розширюваний масив із звичайних ПК із запущеною відкритою ОС та власні застосування і утиліти, яких ніхто за межами компанії не побачить. Фактично вартість ПЗ була пропорційна масштабу і динамічності даних, з якими воно допомагало управлятися.

Сервіс **Google** - це не сервер, хоча доставка сервісу забезпечується масивом інтернет-серверів, - і не браузер, хоча користувач дістає доступ до сервісу саме через нього. І це не прославлений пошукач, який зберігає контент та дозволяє користувачеві здійснювати пошук. Як і телефонний дзвінок, який трапляється не на кінцях телефонної лінії, а в мережі між ними, сервіс **Google** здійснюється в просторі між браузером, пошукачем і цільовим сервером, на якому міститься шукане. **Google** - це посередник між користувачем і його/її онлайнним досвідом.

І хоча **Google** і **Netscape** - софтверні компанії, очевидно, що **Netscape** належить до світу **Lotus, Microsoft, Oracle, SAP** та інших фірм, чий витоки - в софтверній

революції 80-х, тоді як **Google** та інші фірми з ним - це інтернет-додатки (як **ebay**, **Napster** і, чого вже там, **DoubleClick** і **Akamai**).

DoubleClick проти Overture і AdSense

Як і **Google**, **DoubleClick** - це дійсне дитя інтернет-ери. Компанія розглядає ПЗ як послугу, уміє управляти даними і, як зазначено вище, надавала доступ до веб-сервісам задовго до того, як цей термін був придуманий. Проте **DoubleClick** дуже жорстко обмежена своєю бізнес-моделью. У 90-х вважалося, що веб-сервер - це, перш за все, публікація контенту, а не взаємодія; що правлять бал не споживачі, а рекламодавці; що розмір має значення та Інтернет, за великим рахунком, складатиметься з розкручених веб-сайтів, чия популярність зміряна **Mediametrix** або іншим веб-аудитором.

В результаті **DoubleClick** з гордістю згадує на власному сайті про "більш ніж 2000 успішних установок" свого ПЗ. **Yahoo! Search Marketing** (раніше **Overture**) і **Google AdSense** в той же самий час обслуговують сотні тисяч рекламних майданчиків.

Overture і **Google** досягли успіху, тому що зрозуміли концепцію "довгого хвоста" (термінологія Кріса Андерсона) - "колективній потужності маленьких сайтів, які поставляють значущу частину контенту". Пропозиція **DoubleClick** має на увазі підписання офіційного контракту, який обмежує ринок до декількох тисяч великих сайтів. **Overture** і **Google** відшукали спосіб розміщення рекламного модуля практично на будь-якій сторінці. Більше того, вони вважали за краще рекламним форматам, орієнтованим на видавців і агентства (банери, поп-ап), менш нав'язливі, прив'язані до контексту і доброзичливіші до користувача текстові рекламні блоки.

Урок **Веба 2.0**: зробіть упор на призначені для користувача сервіси і алгоритмічну обробку даних, щоб дотягнутися до самих краєчків веба, звертайте увагу не тільки на голову, але й на хвіст.

Недивно, що інші історії успіху **Веба 2.0** демонструють нам приблизно ту ж поведінку. **Ebay** вирішив одиничні транзакції вартістю в декілька доларів між фізичними особами, виконуючи роль автоматичного посередника. **Napster** (хоча і був закритий із-за проблем із законом) побудував свою мережу, не намагаючись створити загальну централізовану базу даних, але спроектувавши всю систему так, що кожен клієнт ставав також і сервером, сприяючи тим самим зростанню мережі. [Взагалі кажучи, закрити **Napster** технічно стало можливо тільки тому, що повністю від централізованості творцям сервісу відмовитися не вдалося. - Прим. ред.]

Akamai проти BitTorrent

Як і **DoubleClick**, **Akamai** був оптимізований для роботи з головою, а не хвостом, орієнтований на центр, а не на околиці. Недивлячись на те що сервіс **Akamai** працював на благо тих, хто знаходився в кінці "довгого хвоста", полегшуючи їм доступ до популярних сайтів, гроші свої компанія одержувала власне від сайтів. **BitTorrent**, як інші піонери P2P-руху, зробив наступний крок до децентралізації Інтернету. Кожний клієнт є і сервером, файли розбиваються на фрагменти, які можуть бути завантажені з різних джерел, непомітно примушуючи користувачів надавати один одному канали і дані. Чим популярніший файл, тим швидше він може бути доставлений, так як більше користувачів забезпечує сумарну пропускну здатність і більше фрагментів цілого файла доступно в Мережі.

Таким чином, **BitTorrent** демонструє нам ключовий принцип **Веб 2.0**: чим більше людей використовує сервіс, тим автоматично він стає кращим. Якщо **Akamai** вимушений додавати сервери для покращення якості послуг, то кожен користувач **BitTorrent** приходиться на вечоринку із своїми ресурсами. Під цим розуміють "партнерську архітектуру", вмонтовану етику кооперації, згідно якої сервіс діє в першу чергу як розумного посередника, що з'єднує краї один з одним і використовує для цього ресурси самих користувачів.

Платформа завжди виграє у додатків

В кожному з попередніх епізодів конкурентної боротьби **Microsoft** успішно розігрівала карту платформи, перебивачи нею самі популярні додатки. За допомогою **Windows Microsoft** замінила **Lotus 1-2-3** на **Excel**, **WordPerfect** - на **Word**, а **Netscape Navigator** - на **Internet Explorer**.

Однак на цей раз конфлікт не між додаком і платформою. Це конфлікт двох платформ, кожна з яких пропонує радикально відмінну бізнес-модель. З одного боку, єдиний поставщик ПЗ з вражаючою базою інсталяцій, сильно інтегрованої **ОС** і **API**, що дає контроль над парадигмою програмування. З іншого боку - система, у якої немає власника, зібрана разом за допомогою багатьох протоколів, відкритих стандартів і угод про співпрацю.

Windows представляє собою апофеоз власного контролю за програмним **API**.

Netscape пробував перехопити ініціативу, використовуючи ті ж прийоми, що і сам **Microsoft** використовує проти своїх конкурентів, але програв. Однак **Apache**, що заснований на відкритих веб-стандартах, процвітає. Коли платформа змагається з платформою і стоїть питання про вибір платформи або, якщо копнути глибше, про вибір архітектури, про вибір бізнес-моделі, то битва ведеться на рівних.

Windows була відмінним рішенням проблем ранньої епохи ПК. Вона розрівняла ігрове поле для розробників додатків, вирішивши багато проблем, що терзали індустрію. Але єдиний ривок, здійснений силами єдиного постачальника, більше рішень бути не може. Він сам стає проблемою. Системи, орієнтовані на комунікації, якою є Інтернет як платформа, вимагають можливості взаємодії на рівні додатків. До тих пір, поки постачальник не контролює обидва кінці кожного з'єднання, його можливості по прив'язці користувача за допомогою **API** обмежені. Будь-який постачальник рішень для **Веб-2.0**, що вирішить заради вигоди замкнути власне застосування на себе шляхом контролю над платформою, за визначенням, не зможе скористатися його сильними сторонами.

Це я не до того, що можливостей для закриття ПЗ і створення конкурентних переваг більше немає, але ми віримо, що вони не лежать в контролі над програмним **API** і протоколами. Правила гри змінилися. І добитися успіху в епоху **Веба 2.0** зможуть ті, хто прийняв нові правила, а не намагається використовувати прийоми, що працювали в епоху програмного забезпечення для ПК.

Стаття 2 © [Джерело](#)

Що таке Веб 2.0.

Використання колективного розуму

Автор: Тім О'Рейлі

Опубліковано 19 жовтня 2005 року

Переклад Валерія Семенюка

Головний принцип, що лежить за успіхом гігантів, народжених в епоху **Веба 1.0**, гігантів, які вижили і зробили **Веб-2.0** таким, яким він є, полягає в тому, що вони підсилили веб-технології за рахунок колективного розуму:

- в основі веба лежать посилання. Коли користувачі створюють новий контент і нові сайти, вони за допомогою користувачів, що виявили цей контент і що поставили на нього посилання, потрапляють в структуру веба. Багато в чому це нагадує формування синапсів в мозку, коли асоціації закріплюються за рахунок багаторазового повторення або яскравості переживань: так само павутина зв'язків розростається за рахунок колективної активності всіх веб-користувачів;

- [Yahoo!](#) - перша успішна інтернет-компанія. Вона народилася як каталог або як директорія посилань - результат старанної роботи тисяч, а потім і мільйонів користувачів. І хоча **Yahoo!** з тих пір диверсифікувала свій бізнес, створюючи самий різний контент, зібрана користувачами колекція до цих пір є її головним активом;
- прорив [Google](#) у пошуку, що миттю зробив компанію беззастережним лідером ринку, був заснований на **Pagerank**, методі, що використовує для забезпечення якнайкращих результатів перш за все посилальну структуру веба, а не характеристики проіндексованих документів.
- продукт [eBay](#) - колективна активність всіх користувачів. Як і сам веб-сервер, **ebay pic** разом із зростанням призначеної для користувача активності, і роль компанії - це роль відкривача контексту, в якому може реалізуватися призначена для користувача активність. Більше того, конкурентна перевага **ebay** полягає виключно в критичній масі продавців і покупців;
- [Amazon](#) торгує тими ж товарами, що і його конкуренти. У них ті ж описи продуктів, ті ж зображення обкладинок і той же редакторський контент від виробників. Але **Amazon** навчився привертати користувачів. У **Amazon** на порядки більше призначених для користувача оглядів; запрошення прийняти участь в роботі сервісу розміщені буквально на кожній сторінці - і що ще важливіше, компанія використовує призначену для користувача активність для забезпечення якісніших результатів пошуку. Якщо пошук на **Barnesandnoble.com**, зазвичай, веде на власні продукти компанії або вже проплачені, то результати пошуку на **Amazon** - це самі продукти, що замоляються і популярність яких обчислюється в реальному часі не тільки на підставі продажів, але й з урахуванням інших чинників, які працівники **Amazon** називають *flow (потік)*. Недивно, що **Amazon**, на порядки випереджає конкурентів по активності користувачів, випереджає їх і у фінансовому відношенні.

Перерахуємо значущі інноваційні компанії, що зробили ставку на взаємодію з користувачами:

- [Wikipedia](#), онлайн-енциклопедія, побудована навколо неправдоподібної ідеї, що енциклопедична стаття може бути додана будь-яким користувачем і відредагована іншим. Радикальний експеримент в області довіри, що на практиці застосував афоризм Еріка Реймонда "у семи тисяч нянюк - дитя в шоколаді" для створення контенту. **Wikipedia** вже зараз знаходиться в першій сотні вебсайтів, і багато хто думає, що незабаром вона опиниться і в першій десятці. Надзвичайна зміна в динаміці створення контенту!
- такі сайти, як [del.icio.us](#) і [Flickr](#). Обидві компанії, що опинилися останнім часом в центрі уваги, першими представили концепцію, яку деякі називають "фолксономією" (в протилежність таксономії), тобто сумісною категоризацією сайтів з використанням вільних вибраних ключових слів (тегів). Розстановка тегів дозволяє вийти за жорсткі рамки категорій і використовувати множинні асоціації, що перекриваються, на зразок тих, що створює наш власний мозок. У канонічному прикладі розміщений на **Flickr** знімок щеняти може бути помічений і як "щеня", і як "милий" - що полегшує подальший пошук інформації;
- створювані спільно спам-фільтри (такі як [Cloudmark](#)), які збирають думки користувачів електронної пошти стосовно того, що є спамом, а що не є ним, і працюють краще, ніж системи, що покладаються на аналіз самих повідомлень;
- загальновідомо, що найуспішніші інтернет-проекти не рекламувалися. Своєю популярністю вони зобов'язані "вірусному маркетингу", "сарафановому радіо". І якщо сайт або продукт залежить від звичайної рекламної кампанії, то швидше за все це не Веб-2.0;
- навіть велика частина самої інфраструктури веба - включаючи Linux, Apache, MySQL і Perl, PHP або Python - зобов'язана P2p-методам відкритих вихідників, які самі по собі є приклади результатів колективної, можливої завдяки мережі інтелектуальної діяльності. На [SourceForge.net](#) більше ста тисяч проектів у

відкритих вихідниках. Кожен може додати проект, будь-хто здатний завантажити і використовувати код, і нові проекти мігрують від країв до центру, якщо користувачі починають їх використовувати.

Природний процес розповсюдження програмного забезпечення повністю забезпечується вірусним маркетингом.

Урок: мережеві ефекти від взаємодії з користувачами - це ключ до ринкового домінування в епоху Веб-2.0.

Блоги і мудрість мас

Одна з самих розтиражованих особливостей епохи **Веба 2.0** - блог. Персональні домашні сторінки стояли у самих витоків веба, особистий щоденник і щоденна колонка - недалеко, загалом, знаходилися. Так з якого приводу шум?

В своїй основі блог - це просто персональна домашня сторінка у форматі щоденника. Але як [зауважив Рік Скрента](#) (Rich Skrenta) - хронологічна організація блога хоч і "здається дрібницею, приводить до абсолютно нового ланцюжка *поширення-просування-вартість*".

Багато в чому блоги зобов'язані [RSS](#) - найзначнішому нововведенню у фундаментальній архітектурі веба з тих пір, як перші хакери зрозуміли, що **CGI** можна використовувати для створення веб-інтерфейсів до БД. **RSS** дозволяє не просто посилатися на сторінку, але й підписуватися на неї, отримуючи повідомлення кожного разу, коли сторінка змінюється. **Рік Скрента** називає це "приростаючим вебом", інші - "живим вебом".

Динамічні сайти прийшли на зміну статичним сторінкам ще десять років тому. Але у разі живого веба динамічними стали не сторінки, а посилання на них. Посилаючись на веблог, ви посилаєтеся на сторінку з постійно змінним контентом, яка містить *пермалінки* (постійні посилання) для кожного індивідуального запису і нагадує про кожну зміну. **Rss**-фід - це набагато чіпкіша прив'язка до сайту, ніж, скажімо, закладка або посилання на конкретну сторінку.

RSS також означає, що браузер перестав бути єдиним засобом для перегляду сторінки. Хоча деякі Rss-агрегатори (такі як **Bloglines**) є веб-додатками, є і настільні клієнти, і мобільні.

RSS зараз починають використовувати не тільки для сповіщень про нові записи в блозі, але і для всіх видів інформаційних апдейтів, включаючи зміни курсів акцій і прогнозу погоди. Таке використання - в деякій мірі повернення до кореня. **RSS** народилася у 1997 році в результаті перетину технології **Really Simple Syndication** **Дейва Вайнера** (Dave Winer), що використовується для сповіщення про зміни в блогах, і нетьскейпівської **Rich Site Summary**, яка дозволяла користувачам створювати довільні нетьскейпівські сторінки з регулярно оновлюваним потоком даних. **Netscape** втратив цікавість до цієї технології, і вона дісталася піонерам блогінгу - компанії Вайнера **Userland**. У нинішніх застосуваннях ми бачимо спадщину від обох "батьків".

Але не тільки **RSS** відрізняє блог від звичайної сторінки. **Том Коутс** (Tom Coates) [відмічає](#) важливість постійних посилань, пермалінків:

"сьогодні це може виглядати очевидним, але пермалінки - ефективний засіб, що перетворив веблоги з механізму простих публікації в говірку безлічі частково пересічних співтовариств. Завдяки пермалінкам стало легко посилатися на конкретні записи в інших журналах і обговорювати їх. Дискусії розширювалися. Розмов ставало все більше. В результаті дружні зв'язки міцніли і ставали надійнішими. Пермалінк був першим - і найуспішнішою - спробою побудувати між блогами містки".

У багатьох випадках комбінація **RSS** і пермалінків додає в **HTTP** функціональність, властиву **NNTP**. Блогосферу можна розглядати як новий P2p-еквівалент **Usenet** і форумам, таким "півним" раннього інтернету. Тепер користувачі могли не тільки з

точністю до коментаря посилається на чужі сайти, але - через механізм трекбеків - могли бачити, хто посилається на них і реагувати: або через зворотні посилання, або за допомогою коментарів.

Цікаво, що двосторонні посилання були метою ранніх гіпертекстових систем (Xanadu). Пуристи вітали появу трекбеків як крок вперед до двосторонніх посилань. Але відзначимо, що трекбеки не були по-справжньому двосторонніми - швидше, вони (потенційно) симетричні односторонні посилання, що створюють ефект двосторонніх посилань. Різниця може здатися незначною, але на практиці системи соціальних мереж (**Friendster, Orkut, LinkedIn**), що вимагають підтвердження одержувача для створення з'єднання, відчувають нестачу масштабування. Як говорить співзасновник **Flickr Катерина Фейк** (Caterina Fake), увага рідко буває взаємною (**Flickr** дозволяє користувачам створювати списки перегляду - кожен користувач може відстежувати фотопотік іншого користувача через RSS. Об'єкт уваги повідомляється, але його дозвіл для створення потоку не потрібний). Якщо ключова частина **Веба 2.0** - використання колективного розуму - перетворює веб-сервер на якусь подібність глобального мозку, то блогосфера - це його внутрішній голос. Може, він і не пов'язаний з глибинними структурами мозку (підсвідомістю), але є аналогом мислення. Потужний сплеск блогосфери як віддзеркалення того, про що люди думають і чому приділяють увагу, викликаний наступним.

По-перше, через те, що пошукачі використовують структуру посилань для відшукування потрібних сторінок, блогери, як найродючіші творці актуальних посилань, почали грати диспропорційну роль у формуванні результатів пошуку. *По-друге*, оскільки співтовариство блогерів володіє високими внутрішніми посиланнями, помітність сторінок ще більше зростає. І навіть нещадно критикований ефект замкнутих співтовариств (їх учасники часто зациклені на одних і тих же темах і не дуже звертають увагу на зовнішній світ) - теж зіграв блогам на руку.

Але якби мова йшла тільки про посилення впливу, феномен блогів був би не цікавий. Але як і **Wikipedia** блоги використовували колективний розум як фільтр. На сцені з'явилася те, що **Джейм Суріовескі** (James Surioweski) назвав "[мудрістю мас](#)". Майже за принципом **Pagerank**, який дає кращі результати, ніж аналіз вмісту документа, колективна увага блогосфери сама по собі стала оцінкою якості контенту.

Медіасайти старого формату розглядають індивідуальні блоги як конкурентів, але суперництво ведеться не з конкретним блогом, а з блогосферою в цілому. Це зіткнення не сайтів, а бізнес-моделей. Світ **Веба 2.0** - це також світ, який **Ден Гілмор** назвав "[ми, медіа](#)". Світ, в якому аудиторія вирішує, що дійсно важливе.

Архітектура взаємодії

Деякі системи спроектовані для посилення взаємодії. Існує три способи створення великої БД. *Перший* - платити людям за її складання (**Yahoo!**). *Другий* - набрати для того ж завдання добровольців (**open-source**-проекти). *Третій* шлях відкрив **Napster**. У клієнті **Napster** за замовчуванням завантажена пісня, яка була доступна для скачування іншими користувачами мережі. Таким чином, кожен користувач **Napster** збільшував цінність розподіленої БД. Потім ця ж схема була повторена в інших P2P-сервісах.

Користувачі можуть підвищити цінність додатку, але мало хто робитиме це добровільно. Тому додатки слід проектувати так, щоб збагачення проекту призначеного для користувача інформацією відбувалося автоматично. Цей момент має бути частиною архітектури додатку.

Вдала архітектура, можливо, навіть більше вплинула на успіх відкритого софтвера, ніж згадані добровольці. Архітектура Інтернету і веба (як і архітектура відкритих проектів) така, що примушує нас автоматично підвищувати їх цінність під час використання. У кожного з таких проектів - невелике технологічне ядро, чіткі механізми розширення і підхід, що дозволяє будь-якій людині додавати нові компоненти, нарощуючи нові шари "цибулини".

Іншими словами, ці технології демонструють мережеві ефекти, просто тому, що вони так спроектовані.

Таку архітектуру взаємодії можна назвати природною. Але як показав приклад **Amazon**, послідовні зусилля (а рівно і економічні стимули - наприклад, партнерська програма) можуть створити подібну архітектуру і в системі, якій за звичайних умов це не властиво.

Стаття 3 ©[Джерело](#)

Що таке Веб 2.0.

Дані - це наступний Intel Inside

Автор: Тім О'Рейлі

Опубліковано 20 жовтня 2005 року

Переклад Валерія Семенюка

Всі сучасні інтернет-застосування зав'язані на бази даних: пошукач від **Google**, каталог (і пошукач) від **Yahoo!**, склад товарів на **Amazon**, картотека товарів і продавців на **ebay**, карти **Mapquest**, каталоги **Napster**. **Хел Веріан** торік навіть сказав, що *"SQL - це новий HTML"*. Компаніям епохи **Веба 2.0** важливо уміти працювати з БД. Так важливо, що деколи ми називаємо нові застосування не **software**, а **po ware**.

Все це підводить нас до головного питання: хто володіє даними?

Очевидно - і тому є безліч прикладів, - що в епоху інтернету той, хто володіє БД, володіє і ринком, а значить, отримує левову частку прибутку. Монополія на реєстрацію доменних імен, надана американським урядом компанії **Network Solutions** (пізніше куплена **Verisign**), була однією з перших по-справжньому грошових операцій в Інтернеті. І якщо зберегти ринкову перевагу, контролюючи **API**, все важче, контроль над важливими джерелами даних забезпечити куди простіше. Особливо якщо ці джерела дорого відтворити (або вони були наповнені за допомогою користувачів сервісу).

Подивіться на копірайти на картах від [MapQuest](#), [maps.yahoo.com](#), [maps.msn.com](#) або [maps.google.com](#). Скрізь буде позначка **"Maps copyright Navteq, Teleatlas"** або **"Images copyright Digital Globe"** (це новий постачальник супутникових зображень). Обидві компанії неабияк вклалися в свої БД. (Тільки **Navteq**, як то кажуть, витратила на створення БД з назвами вулиць і маршрутами 750 млн. доларів. **Digital Globe** довелося розлучитися з 500 млн. доларів, щоб запустити власний супутник, який робить знімки з вищою роздільною здатністю, ніж урядові сателіти.) **Navteq** дійшла до того, що почала ліпити свій логотип на автомобілі, які оснащені системами навігації, - майже як колись **Intel** зі своїм **Intel Inside**.

Дані, поза сумнівом, і є єдиний важливий компонент подібних застосувань, тоді як сам софтвер здебільшого поставляється у відкритому вигляді, а навіть якщо і ні - все одно цілком доступний.

Давайте на прикладі високо конкурентного ринку веб-картографії подивимося, як нерозуміння важливості володіння ключовими даними може погіршити конкурентоспроможність. Першою на ринку веб-карт була **Mapquest** в 1995 році, за нею прийшла **Yahoo!**, потім - **Microsoft**, а недавно до них приєднався і **Google**, - при цьому всі компанії, ліцензують у постачальників інформації, по суті, одні і ті ж дані.

Візьмемо протилежний приклад: **Amazon**. Спочатку його БД була побудована на реєстрі коду **ISBN** від **R.r.bowker**. Бази конкурентів, відповідно, не мали істотних відмінностей. Але на відміну від **Mapquest**, **Amazon** без втоми доповнював дані, додаючи інформацію, надану видавцем, - обкладинки, зміст і навіть фрагменти з книг. Що важливіше, **Amazon** привернув користувачів для написання анотацій, і тепер саме **Amazon** - а зовсім не **Bowker** - є головним джерелом бібліографічної

інформації для філологів і бібліотекарів, не кажучи вже про простих смертних. Також в **Amazon** був розроблений унікальний ідентифікатор [ASIN](#), покриття якого ширше, ніж у **ISBN**.

В загальному, **Amazon** наздогнав і перегнав своїх постачальників інформації.

Уявіть, що так само поступила б **Mapquest**: привернула б користувачів до анотування карт та маршрутів і навіть до створення нових інформаційних шарів.

Боротися з такою компанією конкурентам, у яких в наявності тільки оригінальні дані, що ліцензують, було б куди важче.

Власне цим зараз займається **Google**. **Google Maps** - це експеримент по створенню конкуренції між постачальниками даних і розробниками додатків. Спрощена модель програмування від **Google** привела до появи безлічі додаткових сервісів, які побудовані на поєднанні функціональності **Google Maps** з іншими даними, доступними в інтернеті. Так, наприклад, [housingmaps.com](#) дозволяє накладати на карти від **Google** ріелторські оголошення від [Craigslist](#). На виході у нас виходить нове інтерактивне застосування, чудовий приклад змішування технологій.

У даний момент подібні гібриди в основному є іноваційними експериментами, долею хакерів. Але і підприємницька активність не за горами. Та вже можна бачити як мінімум один клас таких розробників - адже сам **Google** "відвів" роль джерела даних від **Navteq**, перетворивши себе на популярного посередника. У найближчі декілька років ми станемо свідками справжнісіньких битв між постачальниками даних і постачальниками додатків - коли обидві сторони усвідомлюють, що певна інформація може бути ключовою для побудови блоків додатків **Веба 2.0**.

За певні класи ключових даних - місцеположення, особисту інформацію про користувачів, календарі суспільно-значущих подій, ідентифікатори товарів і простору імен - битва вже розпочалася. Якщо відтворити набір інформації - задоволення не з дешевих, то компанія, у якої ці дані вже є, може спробувати скористатися своїм положенням і розіграти карту **Intel Inside**. У інших випадках переможе та фірма, чия база даних першою набере критичну масу за допомогою користувачів, - якщо, звичайно, компанія зможе обернути ці агреговані дані в системний сервіс.

До прикладу, якщо ми говоримо про мережеву ідентифікацію користувачів, то **Paypal**, **Amazon**, **1-click** і мільйони користувачів систем зв'язку цілком можуть вважатися за суперників. (У цьому сенсі остання ініціатива **Google**, що дозволив підтверджувати акаунти на **Google** з телефону, виглядає як спроба розширити свою базу за рахунок телефонних систем.) З іншого боку, є такі стартапи, як [Sxip](#), що зробили ставку на інтегровану особу і намагаються створити розподілене і просте рішення, на основі якого можна буде побудувати єдину підсистему для всього **Веба 2.0**. На ринку календарних довідників є [EVDB](#), що намагається на базі wiki-подібної архітектури побудувати найбільший спільно наповнювальний календар. І хоча сьогодні ще рано робити прогнози, очевидно, що до появи додатків нового покоління приведуть ті стандарти і рішення, які дозволять ефективно обернути певні класи даних в надійні підсистеми "операційної системи інтернету".

Перш ніж йти далі, скажемо декілька слів про користувачів, які оберігають своє *privacy* і право на володіння інформацією як зіницю ока. У багатьох ранніх веб-додатках копірайт враховувався лише номінально. Так, права на всі перегляди, опубліковані на **Amazon**, належать **Amazon**, але компанія нікого ще не переслідувала за їх републікацію. Проте як тільки компанії зрозуміють, що контроль над даними і є їх головна конкурентна перевага, то почнуть стерегти свої дані куди відповідальніше.

Як успіх приватного софтвера привів до народження руху [Free Software](#), так і посилення ролі приватних **БД** вже в наступному десятилітті приведе до народження руху за **Вільну Інформацію**. Ранні прояви цієї тенденції можна побачити вже зараз, в таких проектах, як **Wikipedia**, ліцензії **Creative Commons**, або в проектах програмістів типу [Greasemo](#) (дає користувачам [можливість](#) визначати, як саме відобразатимуться дані на їх комп'ютерах).

Кінець циклу розробки ПЗ

Одною з головних характеристик сучасних інтернет-застосувань є те, що вони розповсюджуються у вигляді сервісу, а не товару. Це, у свою чергу, веде до фундаментальних змін в бізнес-моделях компаній-розробників.

Компанія повинна уміти управляти процесами. Мистецтву розробки додатків повинне супроводити вміння організувати щоденні операції для підтримки роботи цих застосувань. Розрив між софтвером-артефактом і софтвером-сервісом такий великий, що вже зараз не можна написати хороший продукт і забути про нього - його потрібно підтримувати щодня. **Google** щодня прочісує веб-сервер, щоб оновлювати свої індекси, відсікаючи пошуковий спам. **Google** повинен щодня обслуговувати сотні мільйонів запитів, поставляючи користувачеві не тільки якісні результати пошуку, але і контекстну рекламу. І не випадково інформація про системне адміністрування, обслуговування мереж, балансуванню навантаження і тому подібне охороняється **Google**, мабуть, навіть краще, ніж самі пошукові алгоритми. **Google** навчився автоматизувати згадані процеси, а це - ключова частина його цінової переваги перед конкурентами.

Також не випадково, що скриптові мови - [Perl, Python, PHP, а зараз ще і Ruby](#) - відіграють в житті компаній **Веба 2.0** таку ж важливу роль. Перший вебмастер **Sun Microsystems Хасан Шредер** (Hassan Schroeder) якось назвав **Perl** "скотчем інтернету".

Скриптові мови (ери софтверних артефактів, що зневажаються програмістами) - це природний вибір для системних і мережевих адміністраторів, оскільки розробники створюють динамічні системи, які вимагають постійної зміни.

Користувачів потрібно сприймати як співрозробників - як, наприклад, прийнято при розробці відкритого софтвера (навіть якщо само ПЗ навряд чи буде випущено під відкритою ліцензією). Максима відкритого софтвера - "випускай релізи раніше і частіше" - тепер формулюється ще жорсткішим: "нескінченна бета-версія". Програми оновлюються щомісячно, щонеділі і навіть щодня.

Не випадково на логотипах таких проєктів, як **Gmail, Google Maps, Flickr, del.icio.us** і т. п., слівце "beta" може висіти роками.

Відслідковування поведінки користувачів в реальному часі дозволяє бачити, які нові властивості використовуються і як вони використовуються - і це ще одна ключова складова успіху технології. Веб-розробник одного з розкритих мережевих сервісів відзначає: "ми додаємо дві-три нові властивості в різні частини сайту щодня, і якщо користувачам вони не подобаються - ми відмовляємося від цих нововведень. Якщо подобаються - упроваджуємо на всьому сайті".

Кел Хендерсон (Cal Henderson), головний розробник **Flickr**, недавно розповів, що новий [бїлд Flickr з'являється кожні півгодини](#). Це абсолютно інша модель розробки! І хоча поки що не всі веб-додатки розробляються з такою екстремальною швидкістю, майже у всіх цикл розробки радикально відрізняється від всього, що було в епоху ПК або клієнт-серверів. З цієї причини редактори **Zdnet** навіть прийшли до висновку, що **Microsoft** не вдасться перемогти **Google**: "бізнес-модель **Microsoft** побудована на припущенні, що користувач оновлює своє комп'ютерне оточення разів в два або три роки. **Google** же залежить від того, що новенького виявить користувач в своєму комп'ютерному оточенні сьогодні".

Не дивлячись на те що **Microsoft** вже продемонструвала неймовірну здатність вчитися і врешті-решт перевершувати своїх конкурентів, немає сумнівів, що конкуренція змусить **Microsoft** (і - ширше - будь-яку сучасну софтверну компанію) перетворитися на компанію абсолютно іншого типу. Дійсним компаніям Веба 2.0 буде простіше, оскільки їх не тягнуть назад старі підходи (а також супутні бізнес-моделі і джерела прибутків).

Спрощення моделі програмування

Як тільки ідея веб-сервісів стала *au courant*, в боротьбу вступили великі компанії, що виставили складні набори веб-сервісів, які дозволяють розробляти надійні середовища програмування для розподілених застосувань.

Успіх веба багато в чому зобов'язаний тому, що велика частина теоретичних побудов, присвячених гіпертексту, була відкинута на користь простих прагматичних рішень, які і послужили основою ідеальної конструкції. **RSS** став, можливо, єдиним широко поширеним веб-сервісом саме тому, що він простий. А складні корпоративні набори все ще чекають свого часу.

Amazon надає два типи веб-сервісів. Перший не відступає від формалізму **SOAP (Simple Object Access Protocol)**, тоді як другий просто здійснює передачу **XML** через **HTTP** за допомогою спрощеного підходу, відомого як **REST (Representational State Transfer)**. Веб-сервіси першого типу використовуються для B2b-транзакцій (наприклад, між **Amazon** і роздрібними партнерами), але 95 відсотків всіх операцій проводиться за допомогою **REST**.

То ж прагнення до простоти спостерігається і у іншого "сьогодення" веб-компаній. Візьмемо **Google Maps**. Простий AJAX-інтерфейс був швидко "розібраний" хакерами, які потім зуміли використовувати дані, що поставлялися, для організації нових сервісів.

Картографічні веб-сервіси були доступні і раніше: від **Gis**-вендорів (**ESRI**, наприклад) і таких компаній, як **Mapquest** і **Microsoft Mappoint**. Проте **Google Maps** завоював світ, завдяки своїй простоті. І якщо експериментування з даними веб-сервісів від "сьогодення" вендорів вимагало укладання контракту, то **Google Maps** був спроектований так, що дані можна було відразу використовувати в своїх цілях - і хакери дуже скоро навчилися це робити.

Звітси можна винести декілька важливих уроків:

- підтримуйте спрощені моделі програмування і ви отримаєте вільно-зв'язаних партнерів. Проблема корпоративних веб-сервісів в тому, що вони припускають жорстко обумовлене партнерство. У багатьох випадках це виправдано, але часто найцікавіші застосування можуть бути побудовані на вельми крихкій основі.
- думайте про синдикацію, а не про координацію. Прості веб-сервіси - як **RSS** або сервіси на базі **REST** - займаються синдикацією даних, не намагаючись контролювати, що відбувається з інформацією на іншому кінці ланцюжка. Ідея [наскрізної передачі даних](#) є однієї з базових ідей самого інтернету.
- проектуйте з урахуванням можливих переробок і поліпшень. Системи, подібні вебу, **RSS** і **AJAX**, схожі тим, що особливих перешкод для їх повторного використання не існує. Велика частина корисного софтвера знаходиться у відкритих вихідниках, а якщо і немає, то є не так вже багато способів захистити свою інтелектуальну власність. Стандартна браузерна функція "подивитися вихідник" дозволяє будь-якій людині скопіювати будь-яку веб-сторінку. **RSS** був спроектований для того, щоб користувач міг читати контент тоді, коли це зручно йому, а не постачальникові інформації. Найуспішніші веб-сервіси - це, як правило, такі служби, які можуть бути змінені несподіваним для їх творців чином (some rights reserved).

Іновації у зборах

Спрощення бізнесових моделей - це природня супутня обставина спрощеного програмування і спрощених зв'язків. У **Web 2.0** ця властивість добре використовується. Нова служба, подібна **housingmaps.com**, була побудована простим миттєвим злиттям двох існуючих служб. **Housingmaps.com** не має бізнесової моделі (поки що) - але для багатьох невеликих послуг, **Google Adsense** (або, можливо, **Amazon**, яка пов'язана з грошовими зборами, чи обидві) забезпечує успішний проект, що включає еквівалент митної моделі.

Ці приклади інтуїтивно забезпечують другий ключовий принцип **Web 2.0**, до якого ми звертаємося в "іноваціях у зборах." Коли товарних компонент дуже багато, то ви можете новим і ефективним способом створити звичайною їх збіркою нове значення. Так як революція ПК сприяла багатьом можливостям для іновацій у зборі технічних пристроїв, з компаніями подібних Dell, які роблять науку з таких збірок, і таким чином, наносять поразку компаніям, чиї бізнесові моделі потребують іновацій у розробці пристроїв, ми допускаємо, що **Web 2.0** забезпечить можливість компаніям, які конкурують, одержувати кращі результати при використанні і об'єднанні послуг, які забезпечуються іншими.

Інвестиційний тезис Web 2.0

Ризикований капіталіст **Поль Кедроський** (Paul Kedrosky) писав, що "ключем є знаходження дієздатних капіталовкладень, де ви не згодні з консенсусом". Досить цікаво спостерігати, як кожний аспект **Web 2.0** залучає незгоду до консенсусу: кожен, хто робить наголос на триманні даних в таємниці, Flickr/Napster/ і т.д. сприяє розголосу цього. Це не тільки розходиться в думках, щоб бути неприємним (pet food! он-лайн!), це розходиться в думках, де ви можете побудувати що-небудь поза відмінностями. Flickr будує спільноту, Napster - широту збірок. Іншим способом, щоб подивитися на це, успішні компанії цілком відмовляються від чогось дорогого, але вважається критичним, щоб одержати що-небудь цінне безкоштовно це було одного разу дорого. Наприклад, Wikipedia відмовляється від центрального редагування контролю у відповідь на швидкість і широту. Napster відступив на ідеї "каталога" (всі пісні, які продавець продавав) і одержав широту. Amazon відступив на ідеї наявності фізичного фронтона магазину, але повинна була обслуговувати цілий світ. Google відступив на великих клієнтах (початково) і добрав 80%, чиї потреби не зустрічалися. Є щось подібне до айкідо (застосування сили вашого противника проти нього ж самого) у вислові, "ви знаєте, ви правильні -- абсолютно хто-небудь у всьому світі МОЖЕ оновити цю статтю. І догадайся, що, це погана новина для вас." -- Nat Torkington

Стаття 4 ©[Джерело](#)

Що таке Веб 2.0. Софт працює поверх пристроїв

Автор: Тім О'Рейлі (Tim O'Reilly)
Опублікована 21 жовтня 2005 року
Переклад Валерія Семенюка

Збираємо по-новому

Спрощення бізнес-моделі - це природний супутник спрощеного програмування і вільного партнерства. У **Вебі 2.0** повторне використання не засуджується. Нові сервіси, такі як housingmaps.com, є простим поєднанням двох існуючих служб. У **Housingmaps.com** немає бізнес-моделі (поки), але безліч невеликих сервісів живе за рахунок **Google AdSense** (або, можливо, амазонівських програм, або - і тих, і інших).

Ці приклади ілюструють ще один ключовий принцип **Веба 2.0** - те, що ми називаємо "збірка по-новому". Коли навколо стільки дешевих компонентів, ви можете створювати щось цінне, просто збираючи з них несподівані або ефективні комбінації. Точно так, як і ПК-революція дала "путівку в життя" компаніям що збирає

комп'ютери із звичайних комплектуючих, **Веб-2.0** надає можливість компаніям, що збирають свої застосування з чужих компонент.

Софт працює поверх пристроїв

Ще одна особливість **Веба 2.0**, яка заслуговує на увагу, це те, що тепер веб не прив'язаний до платформи ПК. Перед відходом з **Microsoft** розробник **Дейв Стац** (Dave Stutz) дав своєму колишньому працедавцеві [пораду](#): "забезпечити високий прибуток здатне програмне забезпечення, що працює поверх пристроїв".

Звичайно, так можна охарактеризувати практично всі веб-додатки. В кінці кінців, просте застосування вимагає для своєї роботи принаймні два комп'ютера: один - для хостингу сервера, другий - для браузера. І як ми вже обговорювали, розвиток веба як платформи розширює цю ідею до синтетичних додатків, складених з сервісів, які надаються множиною комп'ютерів.

Але - з Вебом 2.0 таке трапляється частенько - "2.0" означає не щось абсолютно нове, а розвиток і поглиблення існуючих концепцій. І фраза Стаца пояснює, як потрібно проектувати додатки для нової платформи.

На даний момент часу кращим прикладом нового підходу є [iTunes](#). Цей додаток без проблем сполучає кишеньковий пристрій з грандіозною веб-базою, залишаючи ПК роль локального кеш-сервера і контролюючої станції. Спроби донести веб-контент мобільних пристроїв, зрозуміло, робилися і раніше, але зв'язка *ipod/itunes* є одною з перших стосовно застосувань, які сполучають в єдиний ланцюжок відразу декілька пристроїв. Інший хороший приклад подібного підходу - цифровий відеомагнітофон **TiVo**.

itunes і **Tivo** також демонструють інші ключові принципи **Веба 2.0**. Вони не є веб-додатками самі по собі, проте використовують потужність веб-платформи, перетворюючи веб на непомітну, практично невидиму частину своєї інфраструктури. **Tivo** і **iTunes** - сервіси, а не коробкові застосування (хоча у випадку з **iTunes** це не зовсім вірно - програма цілком може використовуватися і як коробковий софтвер для управління призначеними для користувача даними). Більш того, і **Tivo**, і **iTunes** намагаються використовувати колективний розум, хоча в обох випадках ці експерименти натикаються на опір з боку лобі власників інтелектуальної власності. У **iTunes** архітектура взаємодії користувачів досить обмежена, хоча останні нововведення в області підтримки підкастинга дещо змінили положення справ у кращий бік.

Все це одна з областей **Веба 2.0**, в якій ми чекаємо тим більших змін, чим більше пристроїв підключено. Які застосування з'являться, коли всі наші телефони і автомобілі будуть не тільки споживачами інформації, але й її постачальниками? Моніторинг транспортних корків в реальному часі, флеш-мобі, любительська журналістика - ось тільки декілька перших ластівок, що натякають нам на можливості нової платформи.

Багаті інтерфейси призначені для користувача

Ще з часів браузера **Viola** (1992) у всіх на вустах слово "аплети" та інші способи доставки активного контенту через браузер. Поява в 1995 році **Java** теж супроводилося згадкою аплетів. **Javascript** - а потім і **DHTML** - були представлені публіці як прості шляхи для виконання додатків на боці клієнта і збагачення призначених для користувача інтерфейсів. Декілька років опісля **Macromedia** використовувала термін **Rich Internet Applications** (втім, ним не гребувала і **Laszlo Systems** - розробник флеш-рішень у відкритих вихідниках), щоб підкреслити можливості **Flash** не тільки в області доставки мультимедіа-контенту, але і як основи для побудови GUI-інтерфейсів.

Проте потенціал веба для побудови повномасштабних додатків не приймали серйозно до появи **Gmail**, за якою послідував **Google Maps**. В обох випадках веб-додатку пропонували багаті призначені для користувача інтерфейси і майже

невідрізниму від ПК-додатків інтерактивність. В одному зі своїх есе **Джес Джеймс Гаррет** з веб-студії **Adaptive Path** "охрестив" використані для цієї мети [технології Ajax](#). Він писав:

Сам AJAX не є технологією. Це декілька цілком самостійних технологій, що працюють разом. AJAX включає в себе:

- обробку стандартів XHTML і CSS;
- динамічне відображення та інтерактивне використання DOM (Document Object Model);
- взаємний обмін і управління даними за допомогою XML і XSLT;
- асинхронне витягання даних за допомогою XMLHttpRequest;
- і JavaScript, що об'єднує все це разом.

AJAX також є ключовим компонентом таких застосувань, як [Flickr](#), додатків від [37signals](#), [Gmail](#) і [Orkut](#) (Google). Ми вступаємо в безпрецедентний період нововведень в призначених для користувача інтерфейсах, раз вже розробники, нарешті, навчилися створювати веб-додатки, які нічим не поступаються в цьому аспекті ПК-додаткам.

Цікаво, що багато з нових можливостей, насправді, немає ніякої новизни. Ще в кінці 90-х р. і **Microsoft**, і **Netscape** розуміли, що це є досяжним, але їх битва привела до несумісності стандартів. І лише після того, як **Microsoft** виграла "браузерні війни" і у нас де-факто залишився один-єдиний стандарт браузера, створення таких застосувань стало реальним. **Firefox**, звичайно, відродив конкуренцію на ринку браузерів, але поки що ми не бачимо руйнівної війни стандартів, яка затримала наш розвиток в кінці XX століття.

У найближчі декілька років з'явиться безліч нових веб-додатків - як абсолютно нових, так і переробок старих ПК-додатків під веб-сервер. Будь-яка зміна платформи створює нові можливості для захоплення лідируючого положення на ринку навіть якщо цей ринок здавався повністю стабільним.

Gmail вже продемонстрував декілька цікавих нововведень в області електронної пошти, комбінуючи сильні сторони веба (доступність з будь-якої точки нетривіальні можливості роботи з даними, можливість пошуку) з призначеними для користувача інтерфейсами, порівнянними по зручності з ПК-додатками. Між тим, поштові клієнти на платформі ПК удосконаляться в іншому напрямі, додаючи до свого інтерфейса, наприклад, можливості інстант-месенджера і датчика онлайн-присутності. Як далеко зайдуть інтегровані клієнти, об'єднуючи найкраще, що є в e-mail, IM і мобільній телефонії (використовуючи **VoIP**, щоб додати голосові можливості)? Гонка почалася.

Легко побачити, як **Веб-2.0** змінює звичні "адресні книги". Адресна книга для **Веба 2.0** використовує локальну базу контактів на ПК або телефонів для запам'ятовування тих контактів, які ви спеціально відзначили для збереження. Між тим **Gmail**-подібний веб-агент пам'ятає всі повідомлення, отримані або відіслані всі адреси, всі телефони і на основі евристики соціальних мереж намагається вирішити, які саме альтернативи вам запропонувати, коли в локальній базі потрібного контакту немає. За відсутності потрібної відповіді система може задіювати соціальну мережу більшого масштабу.

Текстовий процесор для **Веба 2.0** підтримуватиме спільне редагування [в стилі Wiki](#). Але окрім цього він же надає широкі можливості форматування, яке ми звикли чекати від ПК-додатків. [Writely](#) - хороший приклад подібного застосування, хоча мейнстримом такий підхід поки не назвеш.

Але революція **Веб-2.0** не обмежується ПК-додатками. **Salesforce.com** демонструє, як використовувати веб-сервер для розповсюдження ПЗ в якості корпоративного сервісу (CRM).

Завдяки новим гравцям потенціал **Веба 2.0** буде розкритий повністю. Але успіху доб'ються ті компанії, які не тільки навчаються створювати нові інтерфейси, але і (завдяки архітектурі взаємодії) отримають в своє розпорядження спільно підготовлені дані.

Що повинні уміти компанії у Вебі 2.0?

Ми відзначили деякі принципові особливості **Веба 2.0**, але кожен наведений приклад ілюстрував лише одні положення, тоді як інші при цьому упускалися. Давайте спробуємо підсумовувати найважливіші моменти для компаній **Веб-2.0**:

- недорогі масштабовані сервіси, а не коробкове ПЗ;
- контроль над унікальними, складними для відтворення джерелами даних, які можуть бути збагачені за рахунок користувачів;
- відношення до користувачів як до співрозробників;
- залучення колективного розуму;
- охоплення "довгого хвоста" за рахунок самообслуговування користувачів;
- софт повинен працювати поверх пристроїв;
- спрощення моделі розробки інтерфейсів, що призначені для користувача, та спрощення бізнес-моделі.

І коли ви наступного разу почуєте про **Веб 2.0** - звертєся з цим списком. Чим більше пунктів виконано, тим більше компанія відповідає концепції **Веба 2.0**. Втім, не варто забувати, що дійсна майстерність в одній з областей може стати вигіднішою, ніж невеликі уміння в кожній із семи.

Підходи до проектування Веба 2.0

В книзі "[Зразкова мова](#)" **Христофор Олександр** (Christopher Alexander) описує формат для стислого опису рішень архітектурних проблем. Він пише: *"Кожний зразок описує проблему, яка повторюється знову і знову в нашому оточенні, а пізніше пропонує суть рішення цієї проблеми, таким чином, що ви можете використовувати це рішення мільйон разів або без повторного його виконання."*

Довгий хвіст. Маленькі сайти створюють більшу частину контенту інтернету; вузькі ніші створюють велику частину інтернет можливих додатків.

Порада: заохочуйте самостійність користувачів і забезпечуйте алгоритмічне управління даними зі свого боку - це дозволить охопити веб цілком, не тільки центр, але й краї, не тільки голову, але й хвіст.

Дані - це наступний Intel Inside. Додатки все більше залежать від даних.

Порада: для отримання конкурентної переваги відшукайте унікальне, важке для відтворення джерело даних.

Цінність, принесена користувачами. Ключ до успішної конкуренції на ринку інтернет-додатків - збагачення власних даних силами користувачів.

Порада: не обмежуйте свою "архітектуру взаємодії" розробкою софтвера. Явно і неявно залучайте користувачів до процесу поліпшення вашого додатку.

Мережеві ефекти за замовчуванням. Не багато знайдеться користувачів, які за власною ініціативою почнуть вам допомагати.

Порада: зробіть так, щоб зростання ваших даних було побічним ефектом використання додатку.

Деякі права збереження. Захист інтелектуальної власності обмежує повторне використання і перешкоджає експериментам.

Порада: якщо користь забезпечуються сумісним використанням, то забезпечте, наскільки це можливо, слабкий захист вашій ІС. Проекуйте їх з урахуванням "покращеності" і "переробленості".

Безконечна бета. Коли пристрої і програми підключені до Інтернету, додатки перестають бути артефактами і перетворюються на сервіси.

Порада: не намагайтеся упакувати нові властивості в реліз, замість цього додавайте їх по мірі готовності в поточну версію. Зробіть зі своїх користувачів тестерів, здатних відгукнутися в реальному часі, і стежте за їх реакцією.

Кооперація замість контролю. Додатки **Веба 2.0** побудовані як мережа сервісів, що працюють спільно.

Порада: відкривайте інтерфейси вебу, забезпечте синдикацію контенту і використовуйте чужі веб-сервіси, якщо це потрібно. Використовуйте спрощені моделі для програмування та побудови вільно-зв'язаних систем.

Програми працюють поверх пристроїв. ПК не є єдиним пристроєм, на якому можуть виконуватися інтернет-додатки, а додаток, обмежений одним пристроєм, менш цінний, ніж його універсальний конкурент.

Порада: проектуйте додаток так, щоб він міг працювати на кишенькових пристроях, ПК та інтернет-серверах.