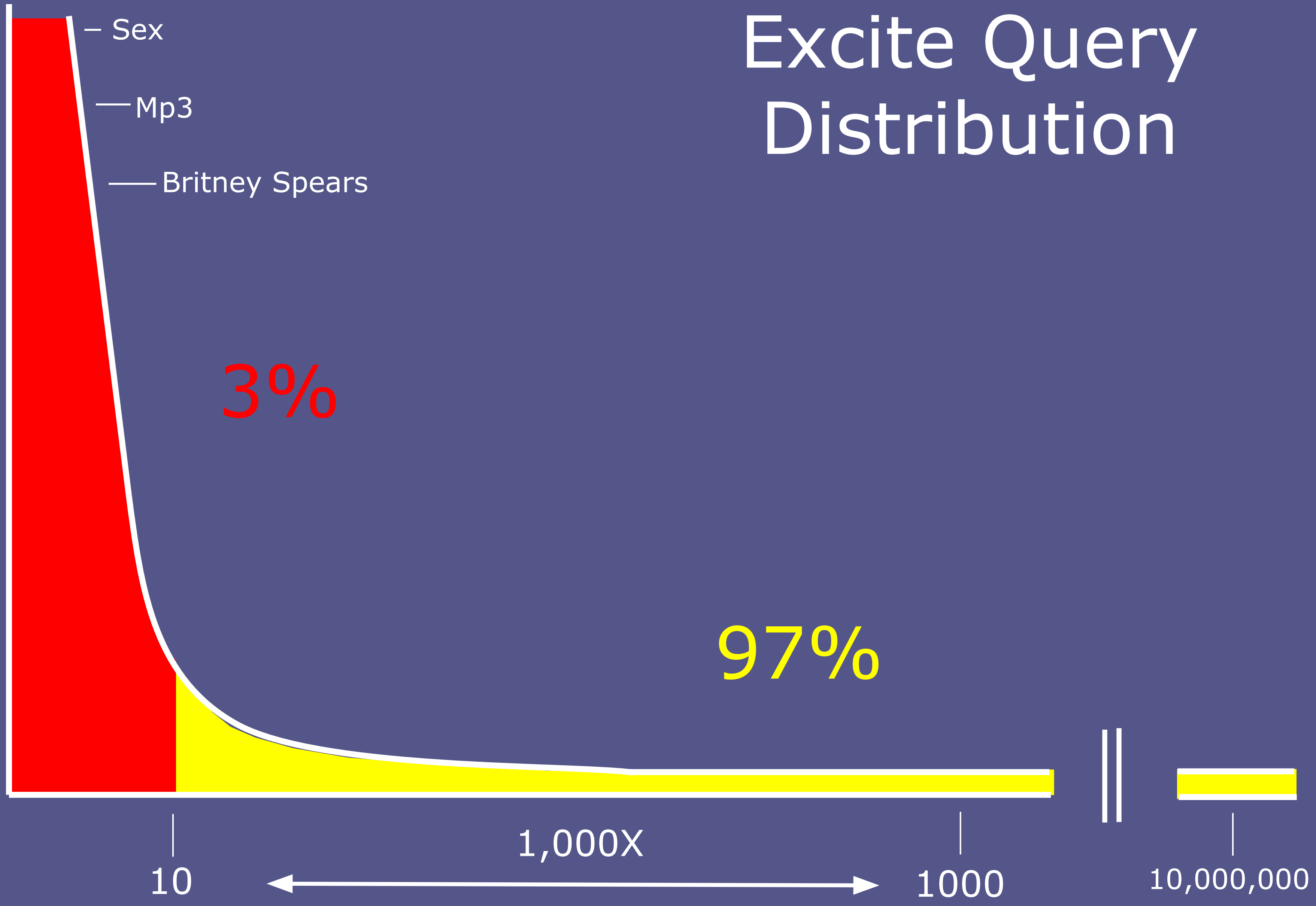
Thinking Small to Get Big

The long tail of software

Jotspot The Application Wiki

Excite Query Distribution

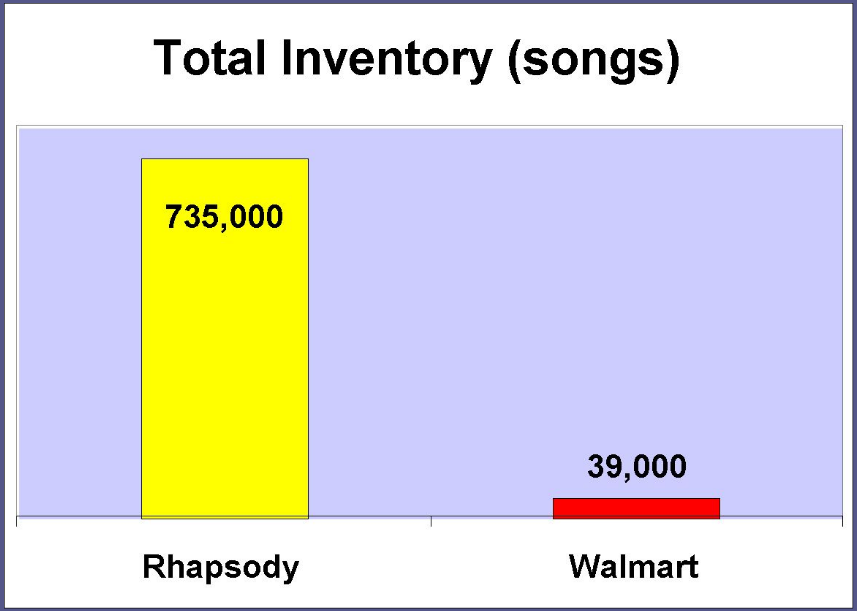


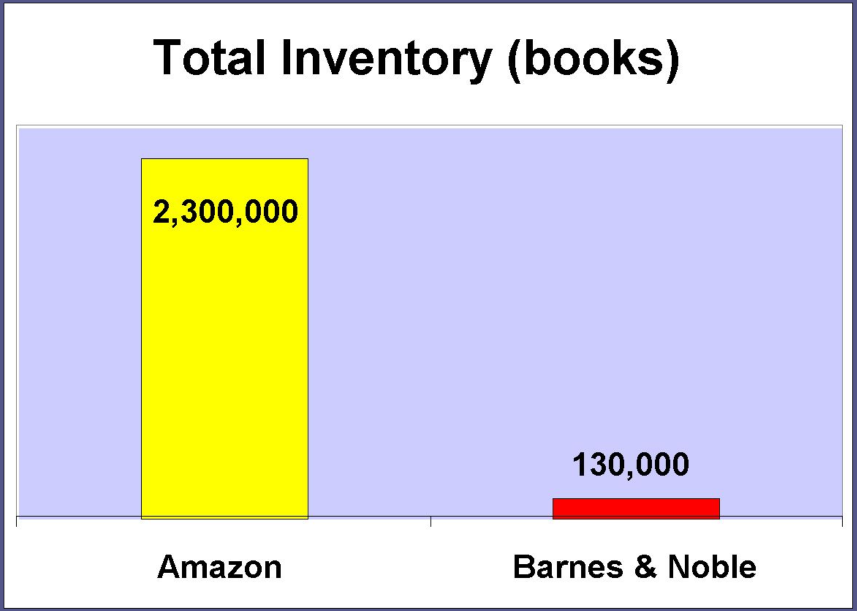
JotSpot The Application Wiki

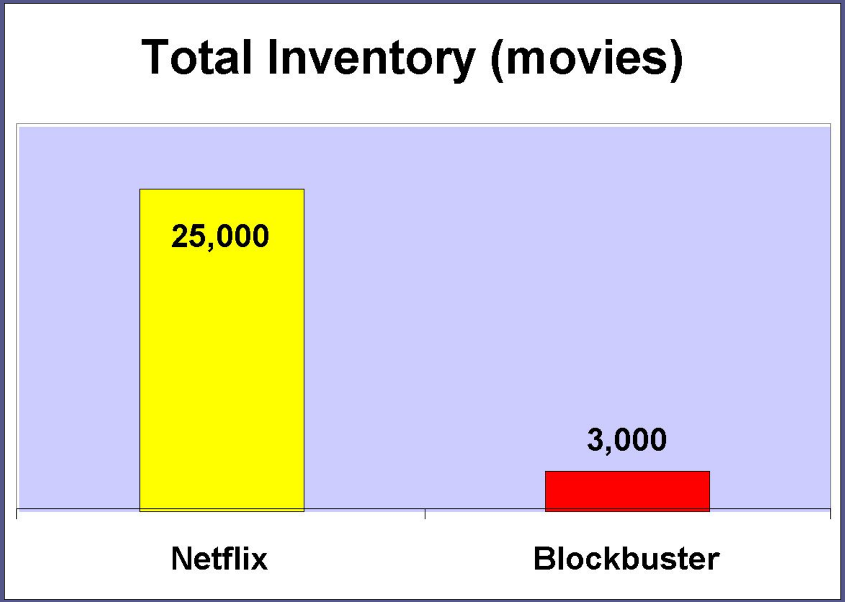
Excite didn’t figure out how to make a business out of 97% of our traffic volume

Google Did. $50bn anyone?

An efficient marketplace for advertisers to reach SMALL audiences.

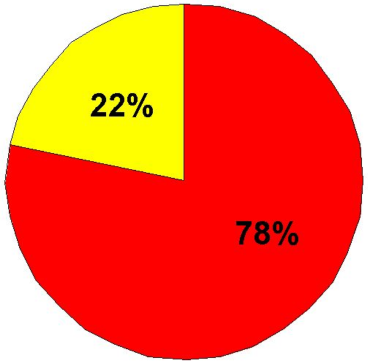




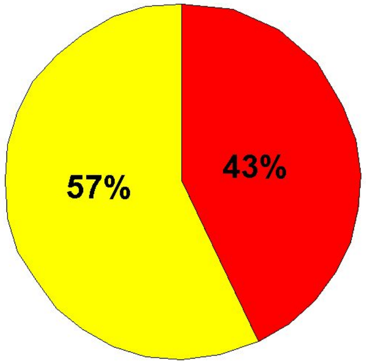


Jotspot The Application Wiki

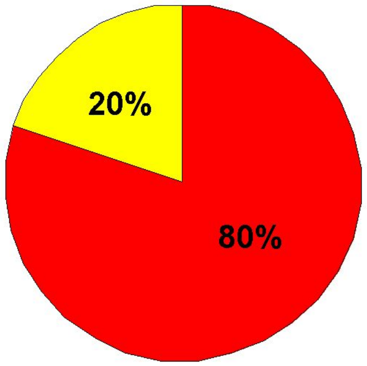
Rhapsody
% Sales from Long Tail



Amazon
% Sales from Long Tail



Netflix
% Sales from Long Tail



Every single iTunes song has been bought at least once

The transformative businesses are going to operate in and monetize the long tail

There's a long tail for software

In the past, the software tail has been inaccessible.

Too hard to write.
Too expensive to write.
Too brittle once deployed.
Too expensive to market and distribute.

The focus has been on dozens of markets of millions

Instead of millions of markets of dozens.

Business software in the long tail.

Jotspot The Application Wiki

Software's Long Tail

<u>Head</u>

<u>Tail</u>

Software's Long Tail

Software's Long Tail

Head

- Fixed, stable feature set

Tail

- Evolvable, changes with requirements

Software's Long Tail

Software's Long Tail

Head	Tail
• Fixed, stable feature set	• Evolvable, changes with requirements
• Architected	• Evolved

Jotspot The Application Wiki

Software's Long Tail

Software's Long Tail

Head	Tail
• Fixed, stable feature set	• Evolvable, changes with requirements
• Architected	• Evolved
• Permanent	• Disposable

Software's Long Tail

Software's Long Tail

Head

- Fixed, stable feature set
- Architected
- Permanent
- 100k-1M users

Tail

- Evolvable, changes with requirements
- Evolved
- Disposable
- 1-1000 users

Software's Long Tail

Software's Long Tail

Head	Tail
Fixed, stable feature set	Evolvable, changes with requirements
Architected	Evolved
Permanent	Disposable
100k-1M users	1-1000 users
Big pieces	Small pieces

Jotspot The Application Wiki

Software's Long Tail

Software's Long Tail

Head	Tail
Fixed, stable feature set	Evolvable, changes with requirements
Architected	Evolved
Permanent	Disposable
100k-1M users	1-1000 users
Big pieces	Small pieces
Monolithic	Loosely joined

Jotspot The Application Wiki

Software's Long Tail

Software's Long Tail

Head	Tail
• Fixed, stable feature set	• Evolvable, changes with requirements
• Architected	• Evolved
• Permanent	• Disposable
• 100k-1M users	• 1-1000 users
• Big pieces	• Small pieces
• Monolithic	• Loosely joined
• Generic	• Situated

Jotspot The Application Wiki

Software's Long Tail

Software's Long Tail

Head	Tail
Fixed, stable feature set	Evolvable, changes with requirements
Architected	Evolved
Permanent	Disposable
100k-1M users	1-1000 users
Big pieces	Small pieces
Monolithic	Loosely joined
Generic	Situated
Lock-in	Open

JotSpot The Application Wiki

Software's Long Tail

Software's Long Tail

Head	Tail
Fixed, stable feature set, Architected, Permanent, 100k+ users, Big pieces	Changes with requirements, Evolved, Disposable, 1-1000 users, Small pieces
Monolithic	Loosely joined
Generic	Situated
Lock-in	Open
Most users not builders	Most users are builders

Jotspot The Application Wiki

Software's Long Tail

Software's Long Tail

Head	Tail
Fixed, stable feature set, Architected, Permanent, 100k+ users, Big pieces	Changes with requirements, Evolved, Disposable, 1-1000 users, Small pieces
Monolithic	Loosely joined
Generic	Situated
Lock-in	Open
Most users not builders	Most users are builders
Low-level tools	High-level tools

Jotspot The Application Wiki

Software's Long Tail

Software's Long Tail

Head	Tail
Fixed, stable feature set, Architected, Permanent, 100k+ users, Big pieces	Changes with requirements, Evolved, Disposable, 1-1000 users, Small pieces
Monolithic	Loosely joined
Generic	Situated
Lock-in	Open
Most users not builders	Most users are builders
Low-level tools	High-level tools
Complex, feature bloat	Simple, few features (but right ones)

Jotspot The Application Wiki

These characteristics imply a set of features…

Flexibility

- Handles structured and unstructured data
- Easy to modify and migrate schemas

Evolvability

- Tolerant development process
- Amenable to easy changes as project progresses

Users are builders

- Short distance between using the app and modifying the app
- Integrated view and edit

Loosely joined, Open

- Easy to get data in and out

Simplicity, Small pieces

- Minimal object model, small number of component types (RESTful)

High-level tools

- Rich environment (environment is useful without even programming)
- Ala Excel

Antecedents

HyperCard

- Rich environment (stacks were useful without scripting).
- Tolerant (data model was flexible).
- Integrated view & edit.
- Not so easy to get data in and out, and app was not networked for group use.

Excel

- Rich environment (spreadsheet is useful without even calculating, much less macros).
- Tolerant (easy to refactor).
- Integrated view & edit
- Simple data model (everything is a cell).
- Easy to get data in and out (CSV, HTTP).
- Not networked, and cell model makes it primarily useful for numerical modeling.

Microsoft Access

- Not a rich environment (you have to build tables before the app is useful).
- Not tolerant (hard to migrate schemas).
- Integrated view & edit for only some users.
- Complex data model.

Lotus Notes

- Rich environment.
- Moderately tolerant.
- Some users can be builders but building is generally complex.
- Not easy to get data in and out.
- More complex object model.
- Not loosely joined.

Traditional Wikis

- Tolerant.
- Integrated view & edit.
- All users can be editors and creators.
- Not loosely joined.
- Inflexible data model: no structured data.
- No programming allowed.

How do we meet the set of required features?

Tolerant Development

- Revision control, flexible data model

Short Distance b/w Using/Editing

- Just one click!

Easy to get data in and out

- Rich standards-based methods to access and publish data

Minimal Object Model

- Everything is a wiki page

Rich Environment

- “Degenerate" application is a wiki, which is highly useful on its own

Backup, Notes, etc.

What enables access to the long tail?

With iTunes, Netflix, Google, eBay, it’s lowering cost to address very small markets.

In Software

That trend has been happening over time. But, it’s still not feasible to address a market of ten.

Software is:
Brittle
Expensive
Lowest Common Denominator

Doesn’t have to be this way

Transformations have happened in the past

Microsoft Excel

Excel

Before

- Highly specified
- Highly technical
- 6 months or more
- Out of step with business pace

After

- Rapidly created
- Disposable
- Single use
- Evolvable
- Far less technical