Five Ways to Scale your API Without Touching Your Code

Steven Willmott
3scale Inc.
@njyx, @3scale

3SCALE

3scale is…

API
Infrastructure
Provider

Power 350+
APIs

110,000
Developers
writing Apps

(Selection)

API Tech Operations

API Business Operations

Developer Support

3scale.net

3SCALE

+ As of today

Also have a
new service for
Developers

Track
Transform
Analyze
API Traffic

http://www.apitools.com

Onward to **Scaling**…

What does **Scaling** Mean?

A: Handling more Transactions per second?

A: ~~Handling more Transactions per second?~~

A: Making more Customers Happy per Second

(whilst using the same resources)

3SCALE

Getting the Question Right

Scaling Transaction

- Decoupling Layers
- Caching
- Compression
- Database Changes
- Streaming / Real-time
- Smaller Payloads
- SSL Optimization

Very Important

Scaling Customers

- Do all of the transaction scaling plus more
- Change API Structure
- Change Client Design Params
- Think about Client needs to rethink transaction loads
- Share scaling burden with developers

Often Forgotten

Suppose you have…

- Fixed Hardware Resources
- (Mostly) Static Application Code Base

Five Scaling Tips That Don't Scale Transactions

3SCALE

1. API Design

Method / Structure Design

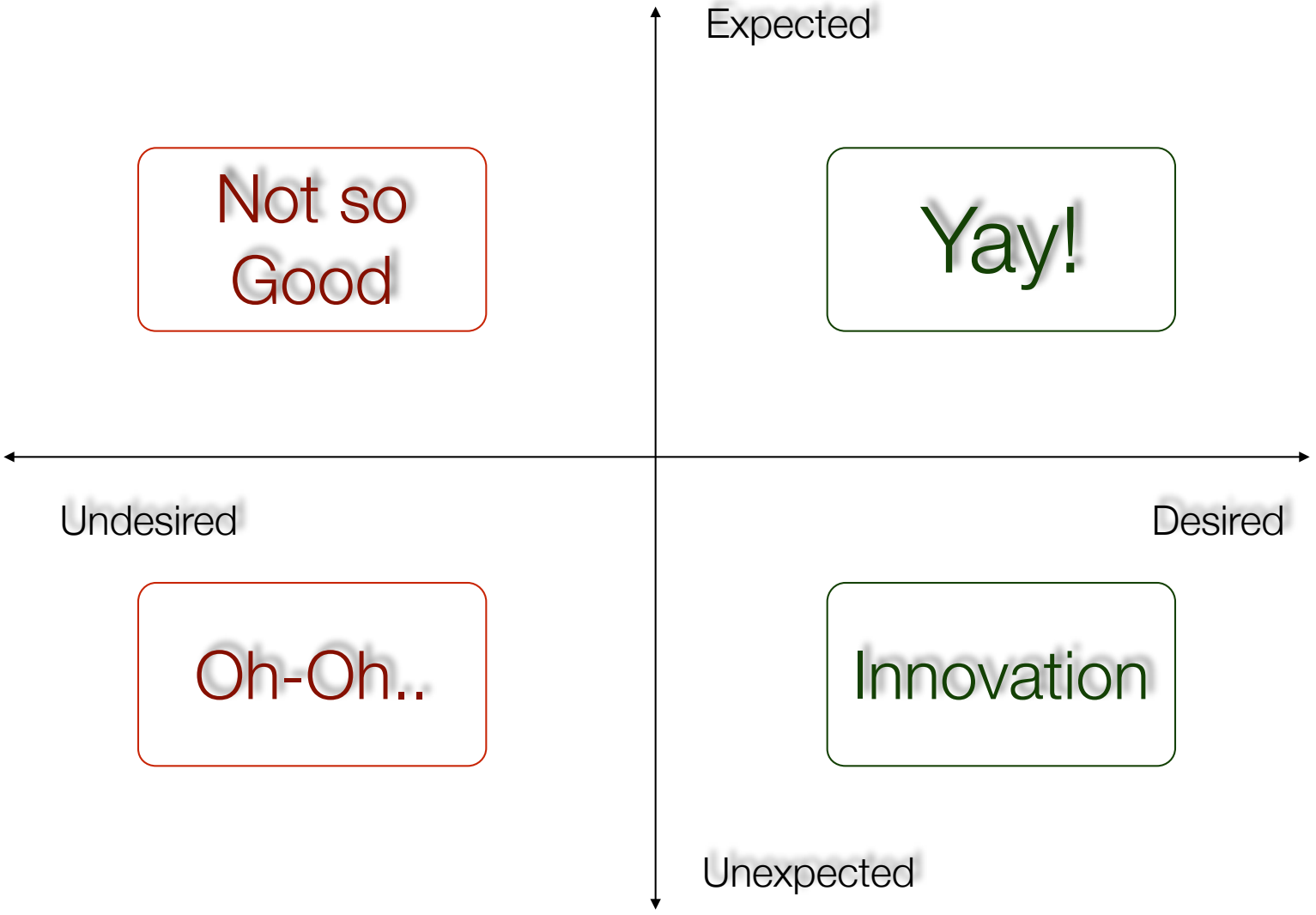
- Do methods / calls fit use cases?
- Are you passing a lot of internal IDs around?
- Are some methods more expensive to execute than others?
- Does the API Design mean heavy load calls are being made even if that data is not needed?
- Are clients making many calls when they should be making one?

Add Specific Methods

Remove Methods

Unbundle & Rebundle

Expected Use of Your API



Concept: Thor Mitchel / Google

3SCALE

2. API Call Aggregation

Post-Hoc API Design

- Creating Aggregate Methods which compose base functionalities
- Cut Access to Individual Methods
- Taylor to Clients or Use Cases
- Adjust the set of compound methods over time
- Works best when you own all the clients

e.g

Great talk by Daniel Jacobsen APIStrat New York:

http://bit.ly/1gAN6mp

3. Rate Limiting

Restrict Outlier Usage

- Rate Limits and Clear communication change Developer behavior
- Rate limits generally mean more sensible code is being written and your servers are taxed less
- Rate Limits stop fringe high volume users from negatively impacting everybody else
- Make sure the limits are published - Developers hate undocumented limits

e.g

Own build…

Make sure key use cases are still feasible, and that rate limits are published.

4. Off Boarding Calls

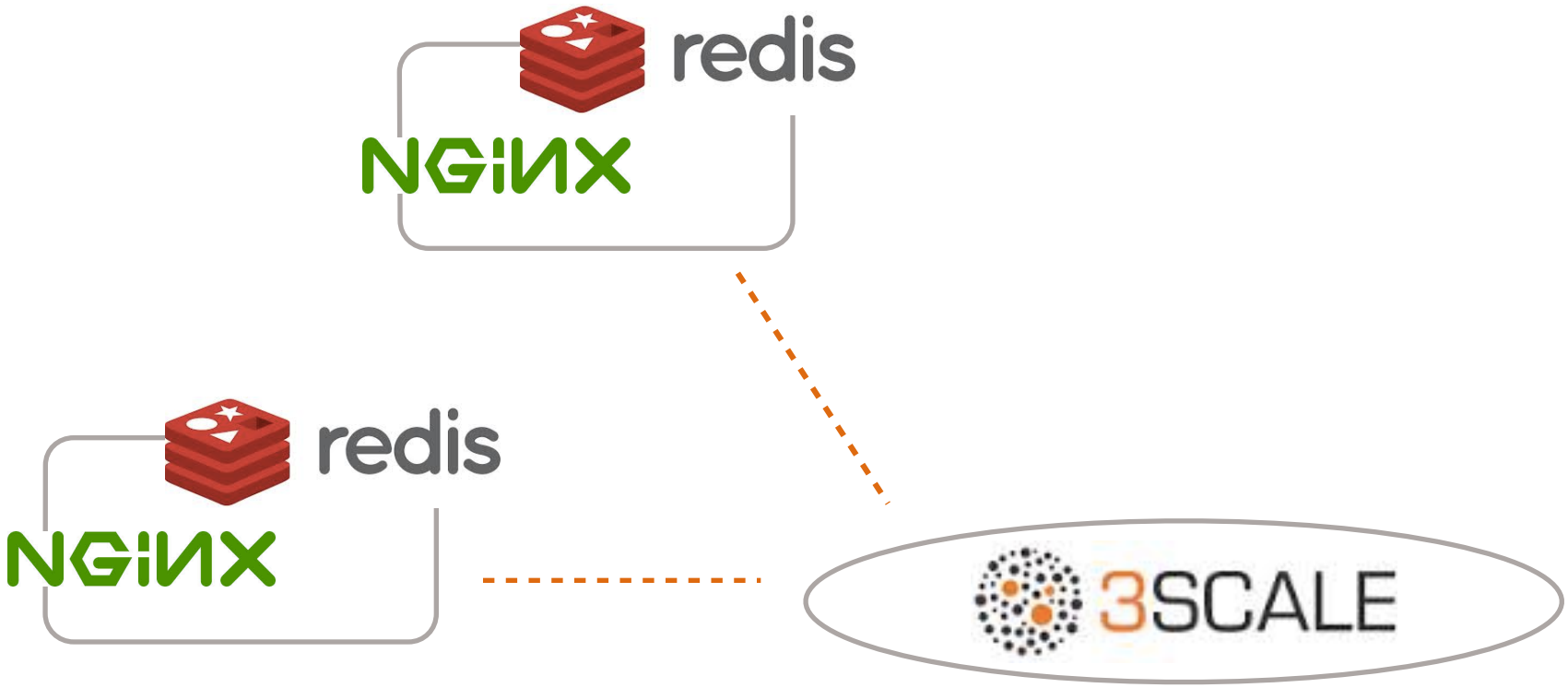
Make the Client Work!

- Calls that are never made you don't have to handle!
- Client Side Caching
- Allow it in your Terms of Service
- Build it into your SDKs, suggest it in your code samples
- Calculate your rate limits by assuming caching
- Use Caching headers
- Consider caching both reads and writes

e.g

Again you need to ensure the core API Use cases are still supported if you go this route

3scale XtremeCapacity



3scale XC Nodes offboard Lookup & Processing to Local Resources

5. SDK Provision

Optimize your Clients

- It can be tough for third developers to write good code
- Writing SDKs can make it easier
- Encodes best practice in terms of API Calls
- Makes Call Patterns more predictable
- You can encode counterpart behavior to your rate limits
- Local check for errors in calls (so your backend never receives them).

e.g

SDKs can create maintenance issues - but critical for large developer bases

Five Techniques

1. API Design
2. API Call Aggregation
3. Rate Limiting
4. Off Boarding Calls
5. SDK Provision

Almost Certainly you have some wins here….

3SCALE

Conclusions

Make sure you are asking the right question!

Your API structure and policies massively affect how it scales

Out of the Box thinking can save a lot of dollars (and the planet)

3scale: http://www.3scale.net

APItools: http://www.apitools.com

APICodex: http://apicodex.3scale.net

Thank You!

Contact:

http://www.3scale.net

@njyx - steve@3scale.net