First Proposal

First serious, second fun. Titles are as appropriate for the Nexmo Developer blog.

Tutorial Title

Preparing to Scale: How to Offload Your Nexmo Number Insights in PHP

Tutorial Abstract

> *Typos happen. In this tutorial we validate user-entered phone numbers using the Nexmo Number Insights API from our PHP web application. Once that's in place, we'll use CloudAMQP's RabbitMQ as a Service to move our requests offline. We'll learn how to use this powerful technique for scaling-out a web application. We'll see how Conway's Law informs our decisions as we separate user-facing web functionality from offline API use.*

About the App

The app is a small PHP/MySQL website allowing us to create user records that include a phone number. We use the Nexmo Insights API to format and validate the supplied phone number. During the tutorial we separate the Nexmo requests/responses from the website to run them offline. We use the Producer/Consumer programming pattern, passing messages via RabbitMQ.

Target Audience

- This tutorial informs solution architects of specific techniques and capabilities suitable for any programming language supported by the Nexmo and RabbitMQ APIs
- This tutorial teaches intermediate and advanced back end developers (in PHP) validation techniques using the Nexmo API, and specific methodologies for scaling-out a web application

Prerequisites

Readers should have

- Intermediate or advanced knowledge of PHP
- Basic knowledge of MySQL and MySQL table design

Familiarity with the CakePHP framework is helpful but not required.

Learning Objectives

Readers of the tutorial can learn:

- How to use the Nexmo Number Insights API in PHP
- That the Nexmo Number Insights API has three levels (basic, standard, advanced) and what the differences are
- How to validate user-entered phone numbers using the Nexmo Insights API (Phone number *verification*, as opposed to *validation*, is covered in the second tutorial)
- The Producer/Consumer programming pattern
- That typical PHP-centric Producer/Consumer use cases deal with web traffic increases (or spikes) and related database congestion
- This example of creating a new PHP-based microservice

Language and Framework

PHP 7.2 and CakePHP 3 using MySQL 5.6. The principles taught with this tutorial apply to any language supported by the Nexmo and RabbitMQ APIs.

Specific Nexmo/TokBox APIs

The Nexmo Number Insights API (each of the three levels).

Related Reading

- How-To Build a REST Implementation that Scales Better than SMPP
- Number Insight API Overview
- How to Use The Number Insight API with PHP
- The Official RabbitMQ Tutorials
- I'm British So I Know How to Queue by Stuart Grimshaw, solid technique and background theory building messaging systems with PHP and RabbitMQ
- Building Microservices: designing fine-grained systems by Sam Newman, outstanding book aimed at both practitioners and architects, teaching patterns for designing distributed systems

External Submissions

- PHP Architect Magazine - for any PHP-centric projects that teach concepts
- freeCodeCamp - any developer-centric articles/tutorials
- Dev.to - any developer-centric articles/tutorials
- codementor - any developer-centric tutorials
- scotch.io - any web development tutorials, thus plausibly anything PHP-centric
- Digital Ocean Community - any developer-centric tutorials
- LinkedIn - anything purporting to be informative
- CakePHP Blog - Anything built as a demo project with the CakePHP framework could be a topic for the FriendsOfCake group or the CakePHP Bakery blog. I can check with the core development team when the time comes.

Three Tweets

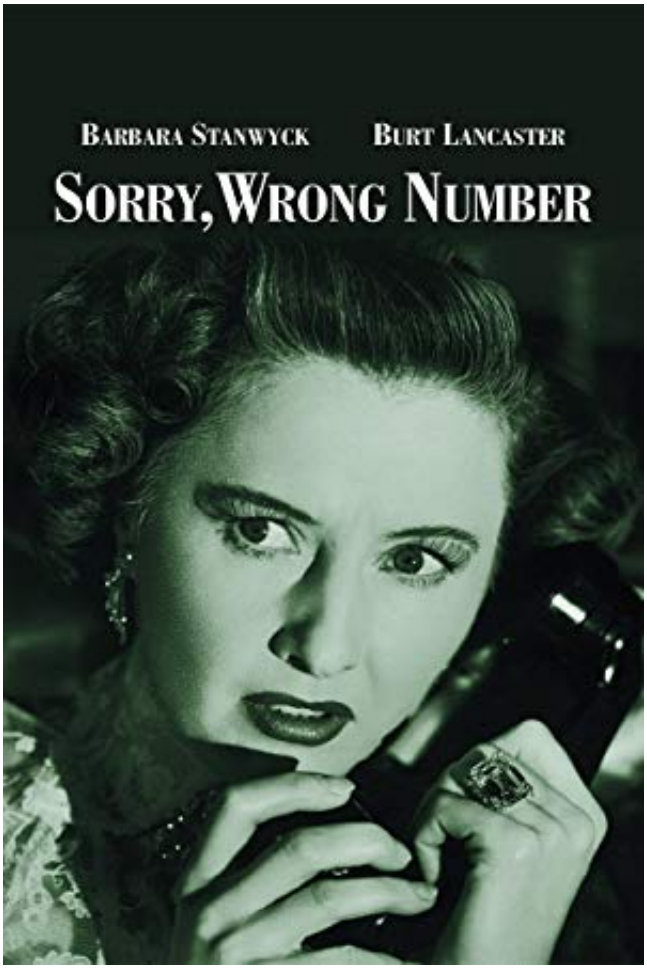
The following tweets should include an image; perhaps even just a screen shot of the code.

1. Typos happen. Validate user-entered phone numbers with the @Nexmo Number Insights API. This PHP tutorial shows you how, step by step.
2. Learn a powerful technique for scaling-out your PHP application using @CloudAMQP's RabbitMQ as a Service and the @Nexmo Number Insights API. Producer/Consumer Programming for the win!
3. Conway's Law helps us keep a separation of concerns. In this tutorial we'll see Conway's Law in action as we learn how to use a @Nexmo API asynchronously, separating it from the main web site.

I do long-form essays on Twitter, such as this 234-tweet epic: Big Iron and the World War II Codebreakers.

A 15-tweet thread might run as follows.

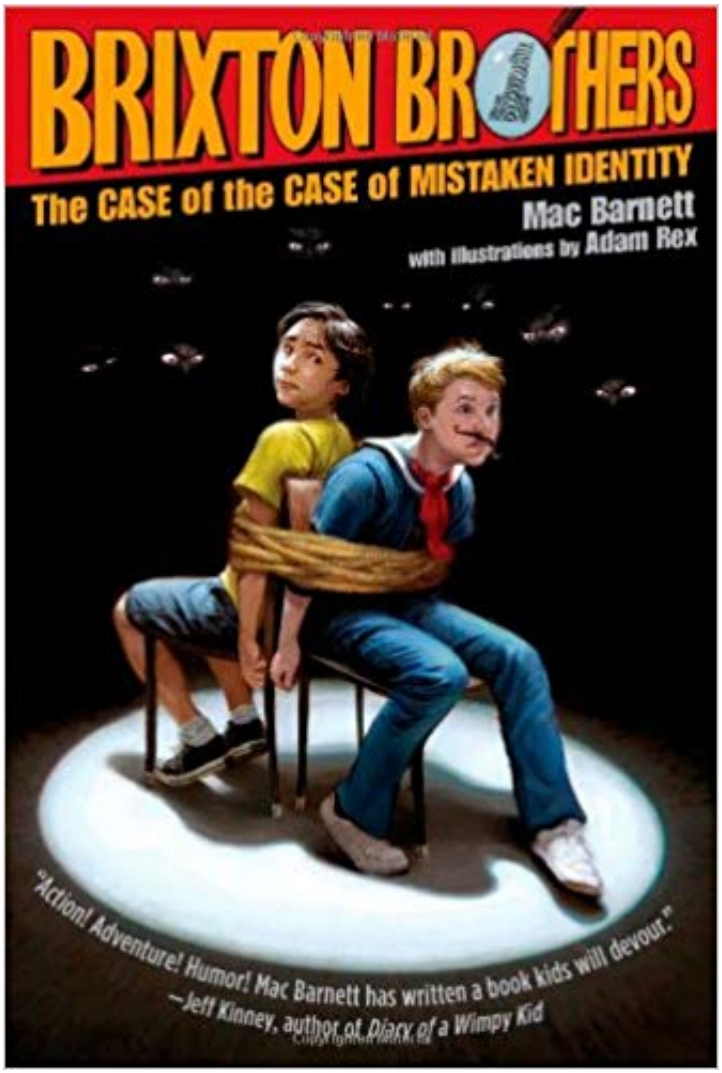
1. Typos happen. With phone numbers, that can get expensive. I've been thinking about this, and wrote a tutorial. THREAD



Sorry, Wrong Number

Photo from https://www.amazon.com/Sorry-Wrong-Number-Barbara-Stanwyck/dp/B00K03Z6TM/

1. Web sites are getting more careful about personally identifiable information (PII), and rightly so. There's an important side issue, which is accidentally gathering information about the wrong person.



Mistaken Identity

Photo from https://www.amazon.com/dp/B002QJZ9W2/

1. @Nexmo has a Number Insights API to help us with this situation. The API has three levels - basic, standard, advanced. https://developer.nexmo.com/number-insight/overview#basic-standard-and-advanced-apis
2. The basic level provides locale and formatting information. I found that useful when I fumbled the first digit of a phone number - I was suddenly routing to a different country! That would be expensive.

Temple Wat Xieng Thong, Laos

Public domain photo from https://imagespublicdomain.wordpress.com/2009/11/25/cia-world-factbook-photos-6-laos-thailand-cambodia-vietnam-indonesia/

1. The standard level tells us whether the number belongs to a land line, mobile device, or is a virtual number. Depending on your situation this might be exactly what you need.

Telephone

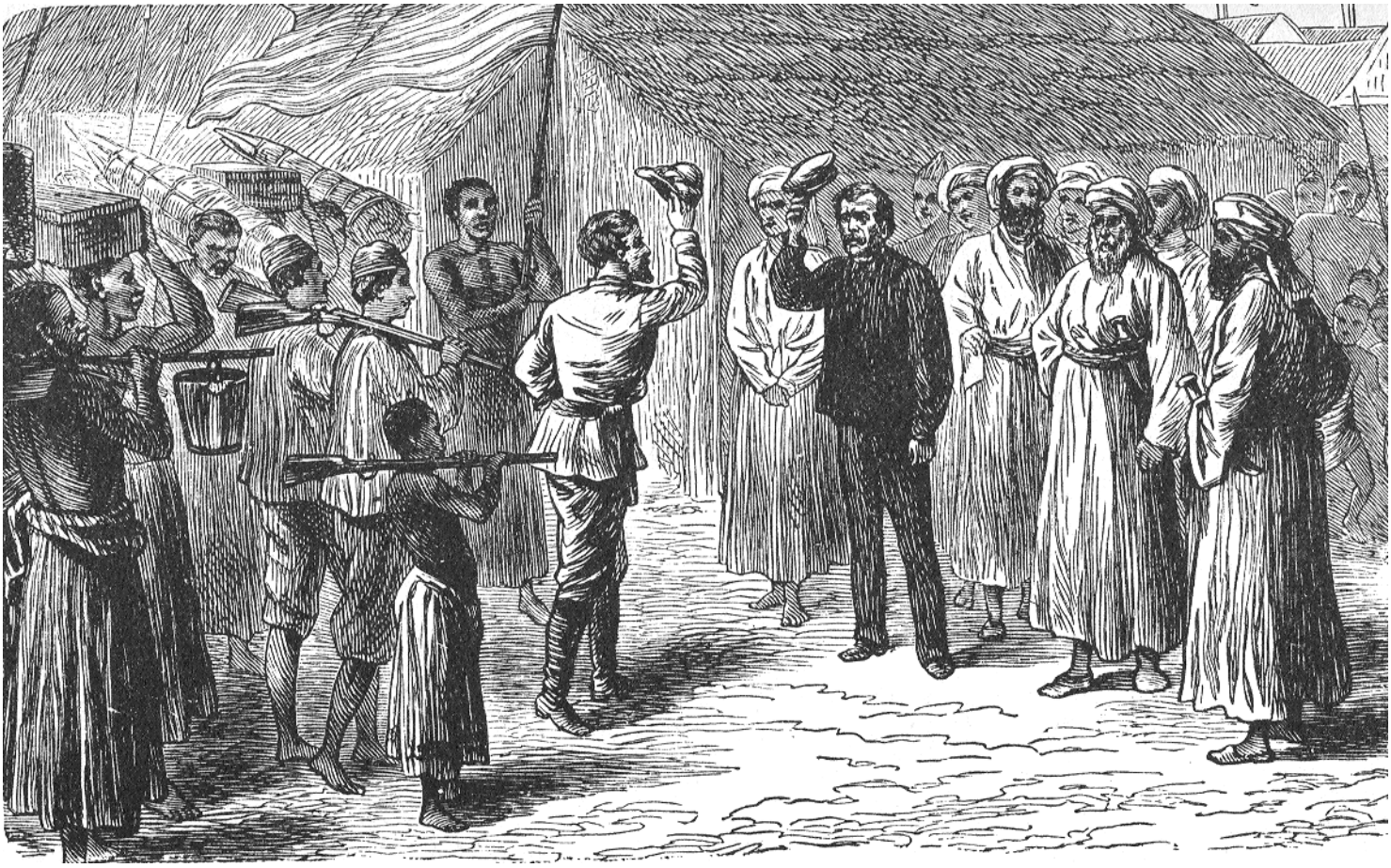
Free image from http://absfreepic.com/free-photos/download/black-vintage-telephone-3872x2592_97851.html

1. The advanced level is for people or web sites who need to learn the risk associated with that number. We can determine, for example, whether the number is both valid and reachable (as of the time of the API call).

Field Telephone

Image from https://www.123rf.com/photo_15766183_old-military-field-telephone-isolated.html

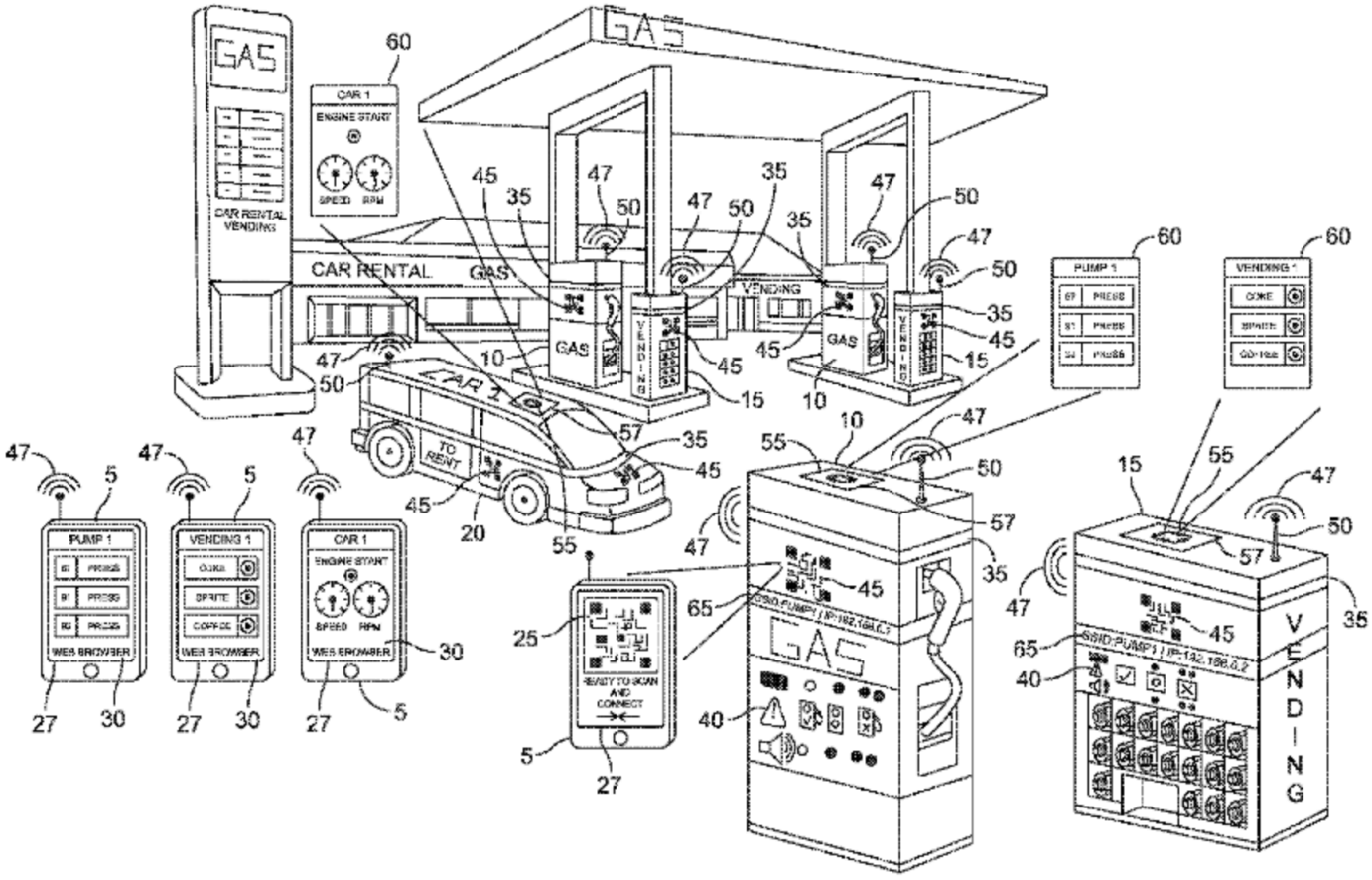
1. Now is a good time to distinguish between number VALIDATION and number VERIFICATION or AUTHENTICATION. (Image, “Dr. Livingstone, I presume?” from Stanley’s 1872 book “How I Found Livingstone”)



Dr. Livingstone, I presume?

Public domain https://en.wikipedia.org/wiki/File:Rencontre_de_Livingstone_-_How_I_found_Livingstone_(fr).png

1. We've been talking about VALIDATION: Is this a real phone number? Is it for a cell phone? Is it reachable? That's what my tutorial covers.



Telephone System

U.S. Patent Office, publication US 2019/0110193 A1

1. A related issue is VERIFICATION, that is, AUTHENTICATION that the phone number is available to the person we think it does. In other words, 2-Factor Authentication.

Two Factor Authentication

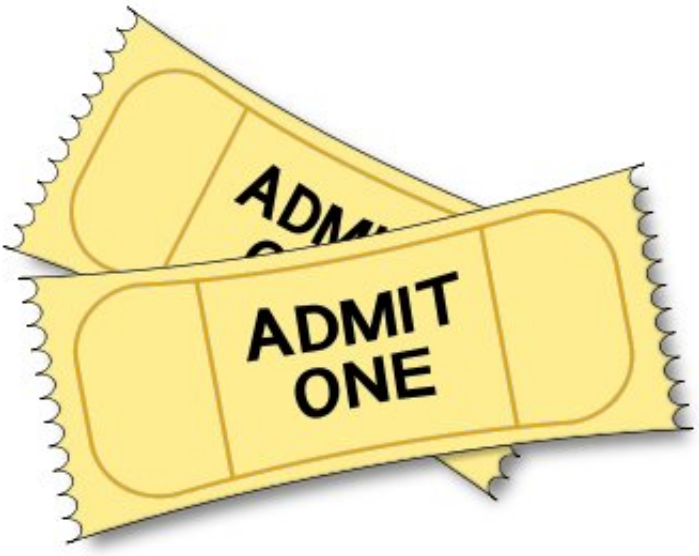
Wikimedia Commons
https://en.wikipedia.org/wiki/File:Diff%C3%A9rents_mod%C3%A8les_de_lecteurs_de_cartes_bancaires.jpg

1. @NexmoDev has the Verify API and workflow for this very situation, but that's a separate tutorial! https://developer.nexmo.com/verify/overview
2. I'm also reminded that software development doesn't happen in a vacuum. When we're collecting information such as phone numbers, there's a reason. We might be preparing for event or tournament registration, for example.

Sporting Event

Photo
https://en.wikipedia.org/wiki/File:Diff%C3%A9rents_mod%C3%A8les_de_lecteurs_de_cartes_bancaires.jpg

1. Registration periods can mean sudden traffic spikes. In fact you could even be dealing with bots sweeping up thousands of concert tickets for scalping on the secondary market.



Ticket

http://clipart-library.com/tickets-cliparts.html

1. The problem, though, is not just the user-facing traffic surge. There could be major workflows behind this registration activity. We could have lots of database work, email verifications, opt in or out, reports to generate, and so on.

Workflows

http://clipart-library.com/tickets-cliparts.html

1. So, I my tutorial explains and demonstrates a pattern allowing us to scale out and ride through the surge. The sample code is open source on GitHub.

Server Room

Public domain https://commons.wikimedia.org/wiki/File:Inside_Suite.jpg

1. Funny thing, that… my tutorial has also just shown you how to build your first microservice. END https://github.com/ewbarnard/NexmoInsight/tree/master/Articles/OfflineLocale

Call for Papers

I generally turn my magazine articles into conference talks (and not the other way around). This proposed tutorial is suitable as both a regular talk and a hands-on workshop. In the workshop (3 hours) we can take on both validate (Number Insights API) and verify (Verify API and workflow) phone numbers.

Talk Titles

- Typos Happen: Validate User-Entered Phone Numbers (regular, 30 minutes, intermediate)
- Producer/Consumer Programming: Offload 3rd-party API use (regular, 60 minutes, intermediate)
- Full Throttle: Producer/Consumer Programming (regular, 60 minutes, intermediate)
- Preparing to Scale: Offload to Microservices (regular, 60 minutes, intermediate)
- Validate and Verify User-Entered Phone Numbers (workshop, 3 hours, intermediate)

Talk Descriptions

I generally pick from one of several potential descriptions when submitting to a CFP. Since most conferences that I've submitted to request multiple submissions, I try to make each *set* of submissions as much of a variety as possible. I can never predict what gets chosen by the organizers, and feedback indicates that it's a matter of filling in pieces of the puzzle in relation to other speakers' talks.

The following descriptions assume submission is to a PHP-centric conference and thus there is no need to mention that the application is in PHP.

Potential descriptions:

- Learn how to validate user-entered phone numbers using the Nexmo Number Insights API. This step-by-step tutorial walks through signup and obtaining API keys, the API request and response, and what features are available in the basic, standard, and advanced API service levels. (regular talk, 30 minutes, intermediate)
- Producer/Consumer programming is a great technique for offloading work from your main application. You can scale resources to meet increased demand. You can "smooth out" traffic spikes by placing your backlog in a queue. You can set aside long-running tasks such as thumbnail generation. We'll develop a bare-bones CakePHP application which uses Nexmo's Number Insights API for validating user-entered phone numbers. We then create a high-performance upstream/downstream workflow that produces and consumes via a free CloudAMQP (RabbitMQ) account. We'll see a pattern for developing microservices as we offload work.
- Typos happen. Mistyped phone numbers can cost money and potentially even trigger regulatory issues when contacting the wrong number by mistake. In this hands-on workshop we'll implement both validation and two-factor verification of a phone number using the Nexmo API and best-practice workflows. We'll learn to use ngrok and webhooks for API callbacks.
  - Prerequisites: ngrok installed; local development environment (LAMP stack or the equivalent)
  - Workshop is hands-on (3 hours); you will be writing PHP code for your local environment