Joy of single purpose services in Go

GopherconIndia 2015
20th Feb 2015

Niket Patel

History

Study the past if you would define the future.
— Confucius

Beehively

- Communication, Calendar, Files.
- Attendance
- Gradebook
- Alerts
- Directory/People management

Ruby

Ruby is simple in appearance, but is very complex inside,
just like our human body.

- Matz, ruby-talk (2000)

But, We are not replacing Ruby.

This talk is about

- How Microservices helped?
- Why Go for Microservices?
- *"Use best tool for the job"* applies to programming language as well.

Major Problems with **Our** Ruby App

- **Memory consumption** - some parts of application traded memory in hope for speed.

- **Speed** - Ironically related to above

Our approach was...

- UX workarounds.
- Throw more money at ENOMEM.

General Ruby related problems

- **Ruby concurrency model** revolves around multiple processes — Our app is particularly nasty with memory consumption. So, multi process concurrency is costly.

- Ruby world moves quite faster then I would like. Things deprecates much faster.

A new problem we created

- **We started new version of Beehively** with goals for better UX and modern technology stack.

- After few months of work we realized that we can't ask our customers to move to new system which has less features. (Second system effect)

Triage

- Better UX options.
- Reduce memory problems.
- Reduce cost if possible.
- Bring new version of Beehively on par.
  (and keep that way)

Microservices

3 Problems

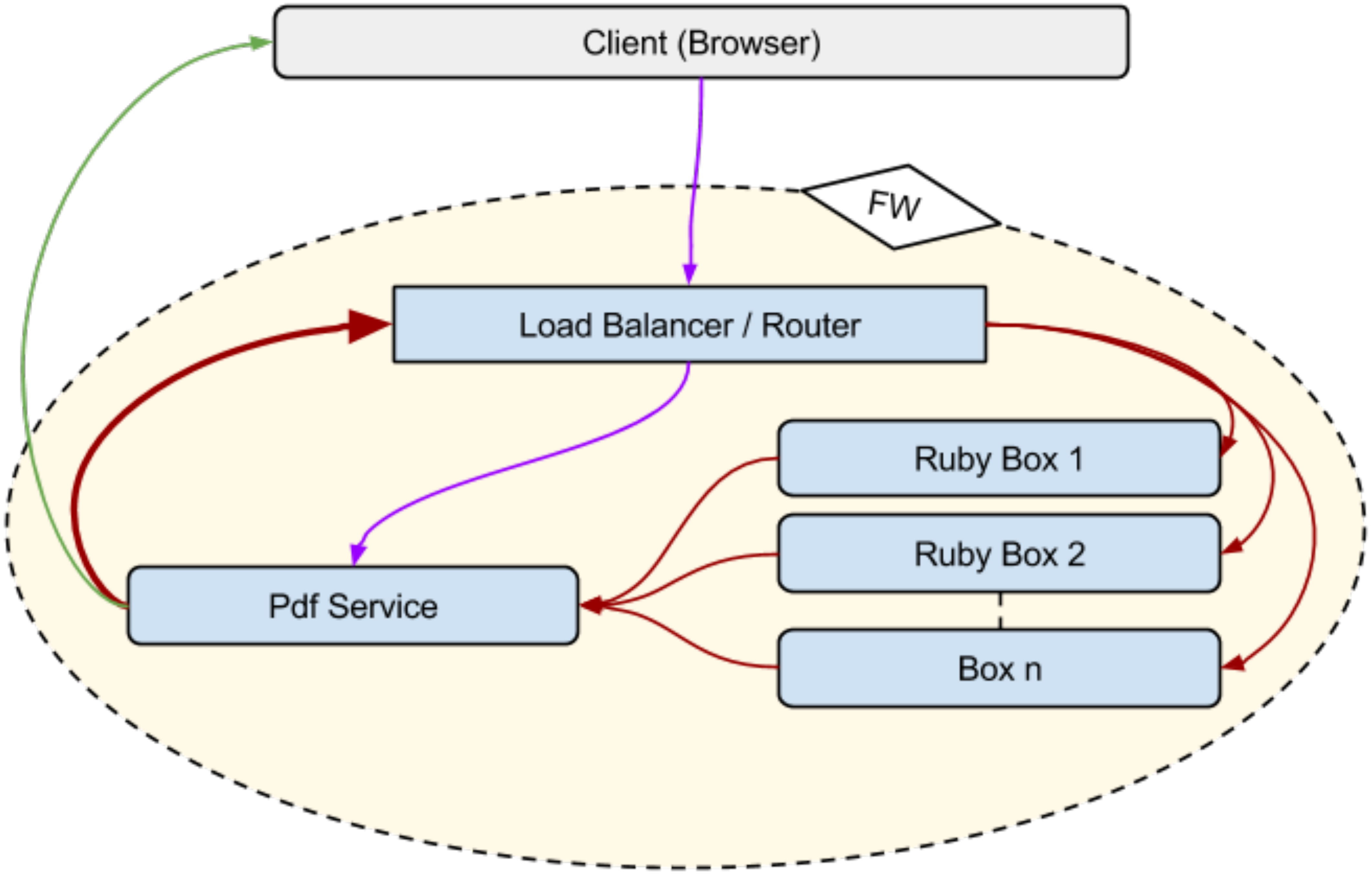
3 Services

Pdf Reports Feature

- Consumed lots of memory.
- Speed

Pdf Service

What is a pdf service?



Request (Inbound)

Method	Path	Request Content	Response
POST/GET	/	application/x-www-form-urlencoded	redirect/message

Request Params

Parameter	Default	Description
path[]	(required)	Url path(s) to fetch and render as pdf
format	Letter	Page format A4, Legal, Letter etc
orientation	portrait	Use portrait or landscape
out	(uuid).pdf	Final pdf name
email		If given deliver final pdf to this email
callback		Callback to upload final file

Except path[] all parameters are optional

Benefits of Pdf Service

- Group reports generating at **12x** speed
- Reduced memory consumption by **30%**
- Accurate status tracking.

Urgent alerts

- This feature send emails, text(sms), voice calls as fast as possible.
- Used rarely and by definition at some unpredictable time.
- Concurrency cost (# of worker * ~400MB)
- Tracking real time progress was nearly impossible (hint, more Memory)

Alert service

What is Alert Service?

- Thin wrapper around our service providers. (twilio and mailgun)
- 10, 30 or 60 workers costs just 5MB
- Real time progress, accurate status as it happens.
- We just need to ensure service remains up — No worries about # of alerts goes out same time.

Method	Path	Request Content	Response
POST	/	application/json	application/json

Request Header

X-Auth-Token : Mandatory header, value should be auth token.

Request Body

```
{
    "send_voice": true,
    "audio_url": "https://url-to-mp3-file",
    "email_html": "%recipient.first%,<p><h2>%recipient.msg%</h2></p>",
    "email_text": "%recipient.first%,\n%recipient.msg%",
    "send_sms": false,
    "send_email": true,
    "from_numbers": ["+19990009999"],
    "originator": "Your school",
    "message": "Hello, welcome to the exciting world of connected apps. -Your
school",
    "people": [
        {
            "first_name": "John",
            "last_name": "Doe",
            "emails": ["john@example.com"],
            "phones": ["+91 999 000 2222"],
            "mobile_phones": ["(888)999-0000"]
        }
    ]
}
```

Gradebook

- Speed
- Memory issues
- UI needed overhaul

Gradebook Service

What is a Gradebook service?

- It manages cache and calculation concerns of Gradebook
- Simple LRU Cache, we got control over memory consumption.
- By managing cache at correct level, we reduced invalidations.
- Calculations were very fast, so I avoided caching results but just source data.

Benefits of Gradebook service

- We achieved 50x speed in some cases.
- In most cases 10x speed up.
- Controlled use of Memory for caching.

Bonus Tip: React.js

- We built Next gen Gradebook UI using React framework.
- React is like Go, very simple to understand so you can concentrate on problem rather then pleasing framework/language.

Fine print

- We used nginx as router.
- Redis as glue
- Go service treat ruby application either as API client or proxy upstream.

Deployments

- Go deployments: build + rsync + upstart
- Recently we started using amazon CodeDeploy.
  - Took couple of hours to ready Go App.
  - Ruby App it took 4 days (with several failed attempts)

Conclusion

Conclusion

- Main Application + Go Microservices is working beautifully.

Conclusion

- Main Application + Go Microservices is working beautifully.
- We are moving to Go from Ruby

Conclusion

- Main Application + Go Microservices is working beautifully.
- ~~We are not moving to Go from Ruby~~
  - We are adding Go

Conclusion

- Main Application + Go Microservices is working beautifully.
- ~~We are not moving to Go from Ruby~~
  - We are adding Go
- Simple Deployments

Caveats*

Use “Go” Today™

Thank You

Niket Patel (niket@niketpatel.com)

Beehively (beehively.com)

@nexneo