DevOps patterns to scale web applications using cloud services

Conference Paper · October 2013

DOI: 10.1145/2508075.2508432

CITATIONS
49

READS
2,247

1 author:

Daniel Cukier
University of São Paulo
15 PUBLICATIONS **163** CITATIONS

SEE PROFILE

DevOps Patterns to Scale Web Applications using Cloud Services

Daniel Cukier
Department of Computer Science - University of São Paulo / Elo7
danicuki@ime.usp.br

Abstract

Scaling a web applications can be easy for simple CRUD software running when you use Platform as a Service Clouds (PaaS). But if you need to deploy a complex software, with many components and a lot users, you will need have a mix of cloud services in PaaS, SaaS and IaaS layers. You will also need knowledge in architecture patterns to make all these software components communicate accordingly. In this article, we share our experience of using cloud services to scale a web application. We show usage examples of load balancing, session sharing, e-mail delivery, asynchronous processing, logs processing, monitoring, continuous deployment, realtime user monitoring (RUM). These are a mixture of development and system operations (DevOps) that improved our application availability, scalability and performance.

Categories and Subject Descriptors H.3.2 [*Information Systems*]: Information Storage and Retrieval—Information Storage; H.3.5 [*Information Systems*]: Information Storage and Retrieval—Online Information Services; D.2.11 [*Software*]: Programming Techniques—Software Architectures

Keywords APIs, AWS, cloud, Cloud computing, DevOps, Elo7, email, IaaS, Load balancing, PaaS, REST, S3, SaaS, scalability, scalable, Tomcat, Web services

1. Introduction

DevOps culture and practices are still in the early stages of adoption [24], but there are already some known DevOps principles that leads to successful websites [21]. While some of these principles are more related to the human side of software development, others are to the technical aspects of software solutions [22].

In this paper, we share the experience of applying DevOps in the context e-commerce startup Elo7. Sellers expose their products in our website. We are responsible for bringing buyers to these sellers. This category of e-commerce is known as Electronic Marketplace[20].

Today, in Elo7, there are more than 105 thousand sellers exposing more than 1.7 million products in the website. There are something around 8 million visits per month, 100 million requests per day in our web servers and more than 2 thousand product search requests every minute. The numbers are even more impressive when we look at the past and see that they doubled every year (and we expect this growth for the next years as well).

SPLASH '13 October 26 - 31 2013, Indianapolis, IN, USA

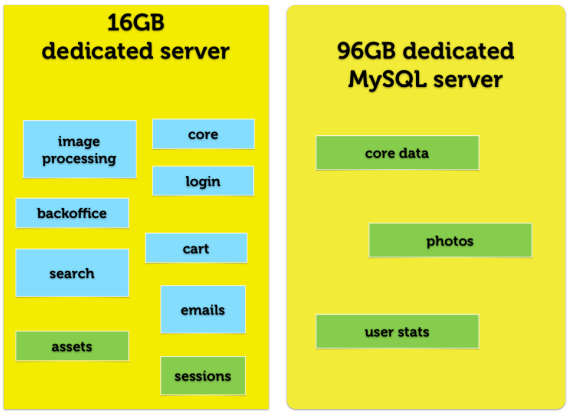

Figure 1. Our Old Architecture

At the beginning of 2011, we decided to move all our infrastructure to the cloud. In that time, we had a very simple and monolithic architecture. There were only two dedicated servers, one for the database and the other for the application. We did not have online backups, data redundancy, disaster recovery plan, hot deployment, or accurate monitoring. All our business data were basically stored in a single MySQL database (core, users statistics, product images, etc). User browser sessions were stored just in the application server session, as well as static application assets like images, javascripts, stylesheets, etc (Figure 1).

After one year of work, our architecture became a lot more decoupled, distributed, complex and robust. Today, we are running using more than 20 virtual servers. We use memcached servers to store user browser session data, Amazon S3 to store products photos and static assets. We also developed a brand new application, which uses a REST API. We deployed a Single Sign On solution to let users to log in once into our multiple applications. Our search data and service is detached from the web application server. We have a test and staging environment with continuous integration and deployment software. We've developed a new back-office application, to empower our customer support team with tools necessary for the daily operations (Figure 2).

These are some of many changes we have been doing. Today, Elo7 has a team of 14 software engineers. The teams are not divided based on people's skills. We do not have operations team. There

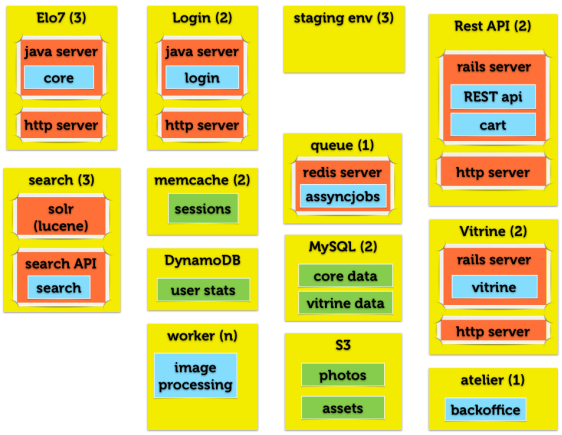

Figure 2. Elo7 new Architecture

are no *ops* guys. Actually, the *ops* **role** is performed by the same people who have the *dev* role, who also perform the *QA* role in this moment. Elo7 has a very flat hierarchy. Of course there are some specialists in each area, but normally everybody does a share of everything.

Given this context, now we would like to share some of the solutions we found trying to improve and scale our platform. Most of the solutions have some parts in the cloud, and they are profoundly based on the idea that running a successful online product is more than just writing lines of code. Furthermore, you must have knowledge of infrastructure, software architectures, product design and agile process management. In this paper we are focusing on infrastructure and software architecture patterns.

The following patterns are far from being a complete list of good DevOps solutions. They are part of something that can be transformed in a Pattern Language of DevOps practices to scale web applications using cloud services.

2. Store Big Files in Cloud Storages

2.1 Context

You have a web application that needs to deliver large files to users. These files can be documents, photos, videos, or other common binary formats.

2.2 Motivation

- Storing big quantities of large files in blob database columns can harm your database.
- Serve large data files from your own web servers consume a lot of resources.
- Large files consume a lot of disk space and you have to manage disk storage by yourself.
- Backing up your own file system can be very difficult and demand a lot of work.

2.3 Problem

Your users do not want to wait too long to download or upload files. They do not accept file corruption at all. When a file is confirmed to be uploaded, it must be safely stored and properly fetched whenever the user needs it. The user does not want to be bothered with "no disk space left" messages. Even if the number of users you serve grows exponentially, you still must be able to have the same good level of service.

2.4 Solution

Serve large static data files using a cloud web storage like Amazon S3 or Google Cloud Storage.

2.5 Resulting Context/Consequences/Side-effects

When you use a cloud based storage, you do not need to worry with data loss. The cloud provider gives you the SLA (Service Level Agreement[1]) for that, and most of the available services have more than 99.999999999% of data durability. You won't need to worry about storage service availability either because the cloud providers SLA is at least 99.9%. Disk space will never be a problem, since storage cloud providers guarantee that you will always have enough space to store whatever you want. Storing a file is as simple as an REST API call.

2.6 Rationale

The facilities that cloud services provide today for file storage are huge. The available solutions have the following advantages:

- Service costs are very low and accessible, even for small startups
- Integration is easy and very well documented sources for doing it
- Scalable and always available by its nature

Of course, if you have a very small amount of static files and they do not grow over time, you do not need a cloud storage. For all other cases, you will have to use one, sooner or later, in your architecture.

In our company, we had a very good experience when we moved all product images from the MySQL database to Amazon S3. First of all, our database size shrunk from 100GB to 5GB. It was crashing every day and we needed a 96GB RAM server to keep the database running. After the migration, we could run our database server in a 8GB RAM server. Moreover, the server performance got a lot better. A lesson learned was: never use relational database to store large files.

We could put these files in a usual server disk, but we would certainly have problem with that since our image base grows very fast. Today we have more then 10TB of network transfer just for images, and there are more than 20 million image files. By using S3, it is also easy to process images (e.g. generate new thumbs) without impacting on website performance.

2.7 Known Uses

Today, we find more and more SaaS companies using cloud file storage to easily scale their service. It is well known that Dropbox uses S3 as it storage platform. Netflix also uses S3 to store and serve video streaming content to their users. A huge list of companies using cloud storage can be found in AWS website[5], which mentions also Airbnb (a community marketplace for people to list, discover, and book unique accommodations around the world), Mendeley (a free reference manager and academic social network).

2.8 Related Patterns / See also

If you are using your own filesystem to serve static photo files, you will need to migrate them to the cloud. If your image database is huge, this migration can take some time, and you will need to right

[1] The contract cloud providers have with customers, that guarantee their service will work properly. It is expected that the SLA is more than 99% in most services

software that guarantees that all photos were properly migrated. One easy way to do this is by enqueuing image copy jobs. For this, you can use the queue process system as described in Section 3.

2.9 Example

Amazon AWS provides libraries that make easy to use all of its services[18], including S3 storage system. Here is a simplified version of a Java class we created to facilitate photo transferring to S3:

```
import java.io.*;
import java.util.*;
import com.amazon.s3.*;

public class S3Utils {
  static AWSAuthConnection amazon =
    new AWSAuthConnection("<KEY_ID>", "<ACCESS_KEY>");

  public static void createImage(String folder,
      String name, byte[] imageBytes) throws IOException {

    Map<String, List<String>> headers =
        new TreeMap<String, List<String>>();
    headers.put("Content-Type",
        Arrays.asList(new String[] { "image/jpeg" }));
    headers.put("x-amz-acl",
        Arrays.asList(new String[] { "public-read" }));

    S3Object object = new S3Object(imageBytes, null);
    amazon.put(folder, name + ".jpg", object, headers)
      .connection.getResponseMessage();
  }
}
```

3. Queue based solution to process asynchronous jobs

3.1 Context

Your web application needs to respond quickly to users, but some user requests need to perform tasks that takes some time to be processed. You want to respond to the user that you are performing these tasks and process them in background. You will publish the task results somewhere the user can fetch later and, sometimes, you will notify the user when the task is done.

3.2 Motivation

- You could perform these tasks asynchronously using *threads* inside your application server, but it does not scale in many platforms, since you would need ten thousands of concurrent threads.
- If a task fail, you would want to re-run them later. You would need to implement a re-schedule police by yourself.
- You need to monitor the tasks progress, e.g. monitoring threads inside a JVM cannot be trivial.
- If tasks consume a lot of infrastructure resources (e.g. CPU, memory), it can harm your web server performance.
- You want easily provision new infrastructure resources when you have a lot of background jobs to process.

3.3 Problem

When you run time consuming jobs synchronously, the end user have to wait for it to continue using your website. Moreover, the web server will lock a thread and a network socket for this user,

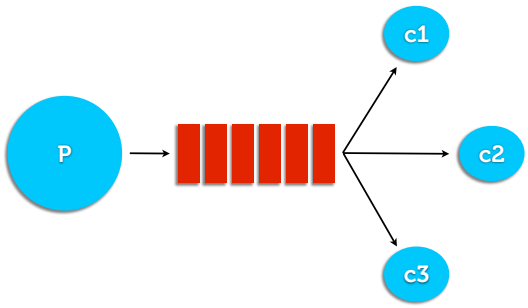

Figure 3. Producer-consumer queue

that could cause eventual unavailability problems. You could use native Java threads to run these jobs, but when they are many, it does not scale as well, since having many concurrent Java threads is limited. If one of these threads fails, you would have to manage this by re-creating it. This would overload your system, causing impact on end user response time.

3.4 Solution

Use a queue based solution to process your jobs. From you web application, you just need to create a new instance of the job in the queue server. You will have a server with the list of jobs to process; and consumer clients that will pop tasks from the server and process them (Figure 3).

The consumer has all the software needed to perform the tasks. You can create as many consumer instances you want. You can create an image of the consumer instance and replicate them using auto-scaling policies.

3.5 Resulting Context/Consequences/Side-effects

With a queue based solution for asynchronous processing, your users will not need to wait for long running tasks to be completed. With a faster response time, their navigation experience will be improved.

On your side, you will have a better control of all background processes. If one of the asynchronous jobs fails, it will be automatically re-processed. You will be able to see reports about queue sizes.

Queue processing workers can be easily configured to scale out, using auto scaling policies based on queue size or CPU usage. You can configure your infrastructure to create new instances to process queues that have accumulated too many jobs. The number of jobs can grow or have spikes, but your response time to users will not be affected by this. If long running jobs have any kind of bugs, they will not affect the user experience either.

3.6 Rationale

Our company has a partnership with Brazilian postal service. Sellers in our platform can send their craft products for at lower rates. To make this solution available to our customers, we had to integrate our system with the Brazilian postal service's. The integration was to generate a posting code, so that our users could get the discount on shipping. The problem was that the postal service did not have any web service for this integration. It only had a web site,

where the user would need to fill manually many forms to get the posting code.

We created something we called HTML as a Service, using selenium-like software, that navigated the Post Office website and generated the posting code automatically. Obviously this process could take sometime to run (open Firefox browser, visit some page, fill some data, etc). In this case, a processing queue was the perfect solution. Our queue started with only this job. Today, we have more than 10 different asynchronous jobs running in the queue.

3.7 Known Uses

Well-known queue processing software include RabbitMQ, ActiveMQ, Resque and Sidekiq. The last two are based on Redis, a very light and stable key-value store database. They are both developed in Ruby but also support Java clients. RabbitMQ have support to Java, .NET and Erlang Client. ActiveMQ is an Apache project supporting integration with many programming languages besides Java such as PHP, Haskell, Ruby and Smalltalk.

3.8 Related Patterns/See also

When your platform is integrated with others through REST web services, sometimes you will not want that these third party services have influence over your end user experience. If you store customer pictures in cloud based storage (Section 2) or send notification emails using a cloud based email delivery service (Section 6), you want all these tasks to be processed by a queue, so your customer will not be affected. This pattern is very useful for all cases you need to integrate with a REST API and want to have a robust integration.

3.9 Example

In Elo7 website, when a user uploads a product photo, it must be processed to generate a optimized version of the image. Moreover, we also need to generate the image in different resolutions (e.g. thumbnails). Here is the code snipped in ruby using Sidekiq that we use to process images:

```
class ConvertImageWorker
  include Sidekiq::Worker
  sidekiq_options queue: :image_processing

  DIMENSIONS = %w(50x50 90x90 300x400 685x685)

  def perform uri_path
    @uri     = URI(URI.parse(URI.encode(uri_path)))
    download_to_temp
    DIMENSIONS.each{|dimension| process dimension}
  end
end
```

The "download_to_temp" method downloads the original image from the Internet to the local filesystem, then the "process" method call executes system calls to ImageMagick software, which converts the original file to a new one, with the new dimentions. The "process" method also uploads the new file to the cloud storage. The code example above uses the Sidekiq queue software. It is one of many open source queue solutions. You can also test Resque, RabbitMQ[27], Apache ActiveMQ[25].

4. Prefer PaaS over IaaS

4.1 Context

You have a complex application that has many architecture components. You need to deploy and maintain different pieces of software (e.g. Operational Systems, Relational Database Systems, NoSQL Databases, cache systems, firewalls, virtual machines, etc). Your companies core business is not technology, and you use infrastructure as a commodity, not as competitive differentiation.

4.2 Motivation

- It is very expensive to maintain your own low level infrastructure. Probably you will need to have staff dedicated to this task. Manage infrastructure will take you precious time. You'd rather to dedicate your engineers time to business related issues.
- You need to find professional software engineers with different skills to cover all the requirements of your systems architecture.
- Developing a highly available and scalable architecture is not trivial.

4.3 Problem

Maintaining operational system installations demands knowledge and time of your engineering team. If your engineers spend much time on these tasks, they will not give the proper attention to business related issues, that are higher priority for your customer.

4.4 Solution

Run your application using Platform as a Service (PaaS) instead of Infrastructure as a Service (IaaS)

4.5 Resulting Context/Consequences/Side-effects

When you choose to use PaaS over IaaS, you will not have to worry about a lot of aspects of the architecture, especially scalability and availability. Moreover, most of the time you won't need to monitor any low level infrastructure, focusing on monitoring just application related metrics. This is good to keep you focused on you business core.

On the other hand, when your PaaS provider has any infrastructure problem, your application will be affected out of your control. Besides this, PaaS platforms are usually more restrictive regarding to technologies or libraries that you can use. You will not be completely free to choose any software to run under PaaS. Because of security or performance issues, PaaS providers standardize what you can and what you cannot run in their platforms.

Another important aspect to consider when choosing PaaS over IaaS is cost. Obviously, the infrastructure cost still exists for PaaS providers. In the end, you will have to pay for the infrastructure plus the PaaS provider profit. Even when they have lower infrastructure costs compared to yours (because of their scale), sometimes using PaaS will be more expensive than having your own infrastructure. It is not easy to calculate the final cost of the whole solution, so you will have to spend some time to conclude what choice is better for your environment.

In conclusion, if you do not want to spend your time on infrastructure, nor your technical team has the sufficient knowledge about scalability, security, availability and other non-functional requirements of your products, you'd better choose PaaS (Table 1).

4.6 Rationale

In our company, we wrote a very small ruby application that serves to forward incoming phone calls to developers when they are working on weekends and the support team detects some anomalies in the website. We use the Twilio service as a Telecom provider. Twilio integrates very well with web services. We created a web service that has a small database with our developers phone numbers and the schedule of what developer is working on each weekend. We did not wanted to deploy this application in one of our servers and maintain it, so we deployed it on Heroku. Since the service has a very low usage, we do not even have to pay for it.

PaaS	IaaS
• Low knowledge about infrastructure • No knowledge about scalability or secutiry • Do not want to monitor infrastructure • You application does not have have any specific technology	• Want to have more control of the infrastructure • Infrastructure is core to your business • You have software components that are do not conform to PaaS providers standards

Table 1. When to use PaaS or IaaS

4.7 Known Uses

Web hosting providers are a good example of PaaS used by many companies, but examples are not restricted to them. Actually, most of providers do not offer an automatic scalable solution. But some of them do and are supposed to be very scalable and easy to use, like Heroku, CloudFoundry and Google App Engine to cite some. Examples of companies who use these services are Art.sy, ASICS, CloudApp, CareerBuilder, Gigya, Best Buy, Boo-box, etc [12][14].

4.8 Example

There are tens of examples of PaaS providers. To cite some: Amazon Web Services Elastic Beanstalk, AppFog, Cloud Foundry, Engine Yard, Force.com, Google App Engine, Heroku, Windows Azure, etc. Most of web hosting companies like Kings Host, Dreamhost, Locaweb, Linode, etc are also examples of PaaS providers. In some of them, deploying code to production is as easy as executing two or three command lines. In Heroku, you deploy an application with these commands:

```
$ heroku create
$ git push heroku master
```

Creating a Play Framework 2.0 Application and deploying to Cloud Foundry is as easy as:

```
$ play new hello-world
$ play dist
$ cf push --path=dist/hello-world-1.0-SNAPSHOT.zip
```

5. Load Balancing Application Server with memcached user sessions

5.1 Context

You run a web application over Tomcat or a similar Java web application server that needs to serve thousands of users. There is no single server that is capable to hold all your user requests. Moreover, you need to be fail proof for server outages. This means that if one of your server fails, you still need to deliver the service to your end user.

Your application is constantly changing. You need to deploy new features almost every day. You cannot stop your service while you deploy new code to your servers. There are a lot of user session information stored in the web servers and you do not want to lose them after server restarts. You could have all session information stored in cookies, but your application has a lot of legacy code, that could be difficult to change and refactor.

5.2 Motivation

- Tomcat hot deploy does not work perfectly. After a couple of hot deploys, the server usually crashes or goes to a memory leak state.
- Your will have an outage if you restart Tomcat and have only a single server.
- If you have multiple independent servers behind a Load Balancing with sticky sessions, users will lose their session data when their respective servers are out.
- Having a complete stateless application is not always possible, specially when there is a lot of legacy code.
- The cluster provided by Tomcat does an all-to-all replication which limits scalability.
- Deploying

5.3 Problem

If you do not use any persistence strategy for user session data, when your application server restarts, users lose their session information. Logins, cart items and other useful information are lost. You could store this information in cookies, but you have the downside of sending these cookies in all transactions degrades your application performance. Moreover, cookies are limited in size.

Even if you have a session storage strategy, if it is not memory based, you will fall into performance and availability issues as well. By using the Tomcat native cluster solution, you cannot scale out well, because all sessions are replicated to all servers. This means that the more users you have, the more memory you will need in each server. This obviously does not scale for millions of users.

More than the session data, you need to balance network traffic between Tomcat servers. You could do this by deploying a HAProxy or Nginx reverse proxy software over Linux instances in the cloud. But you would have to manage this server. Moreover, it would be a single point of failure in your architecture.

5.4 Solution

Use memcached session manager (MSM) software[13] to store user session data and replicate it between many tomcat servers instances. Put all servers behind a Load Balancing, preferring cloud based load balancing.

5.5 Resulting Context/Consequences/Side-effects

With many application servers running behind a Load Balancing and sharing sessions with memcached, your users will not lose their session after Tomcat restarts or fails. You will not have a single point of failure anymore. This solution handles also memcached nodes failures as well. You will be able to deploy new code in the middle of the day, with no website outage. If one of your servers fail, users will automatically be migrated to a new one, preserving all their session data.

The number of users can grow infinitely, you just need to create new Tomcat7 servers, put them behind a cloud based Load Balancing and configure the session manager to use the memcached session replication strategy.

5.6 Rationale

The Tomcat provided cluster solution DeltaManager does an all-to-all replication which limits scalability. The other Tomcat approach, BackupManager, does replication to another tomcat, that requires special configuration in the load balancer to support it. The memcached-session-manager supports non-sticky sessions (not supported by Delta/BackupManager) and provides session locking to handle concurrent requests served by different tomcats.

Moreover, DeltaManager/BackupManager use java serialization, memcached-session-manager comes with pluggable serialization strategies is based on kryo, one of the fastest serialization libs as of today (2013).

When using cloud based load balancing, you will not have to worry about availability and single point of failures. Your provider guarantees that the Load Balancer will always be there. Moreover, configuring these cloud based load balancers is as easy as navigating in the Internet. Providers offer easy-to-use user interfaces to configure load balancers.

5.7 Known Uses

Amazon Elastic Load Balancing is an example of cloud based Load Balancing. Rackspace also has its own solution for that cloud based load balancing. Some examples of companies that use these services are R7, Ci&T, Grupo El Comercio, Kununu, 36Boutiques, etc.

Tchibo, one of the biggest mail order companies and e-commerce shops in Germany, uses memcached session manager. So do GMX, one of the biggest mail service providers in germany.

5.8 Related Patterns

There are many different solutions for load balancing web servers [7]. The one cited in this paper is simple to implement, but very powerful to scale. Even if you have many servers load balanced, you still need to monitor their performance and constantly check log files in these servers. For that, we recommend taking a look at Logging (Section 7) and Real User Monitoring (Section 8)

5.9 Example

Configuring the Tomcat server to use memcached is simple. You just need to follow the setup and configuration instructions provided in the memcached session manager website[13]. Basically, you will need to have a memcached node created in your infrastructure and point your Tomcat to it with the following configuration in your context.xml file:

```
<Context>
...
 <Manager
 className=
 "de.javakaffee.web.msm.MemcachedBackupSessionManager"
 memcachedNodes="n1:host1:11211,n2:host2:11211"
 failoverNodes="n1"
 requestUriIgnorePattern=".*\.(ico|png|gif|jpg)$"
  />
</Context>
```

6. Email delivery

6.1 Context

Your application sends a lot of emails to users when they perform some specific tasks, specially transactional operations (e.g. new user registry, order confirm, payment received, etc). You need a guarantee that these emails are properly delivered, since they are an important part of your product user experience.

6.2 Motivation

- Having your own SMTP server for email delivery can cost a lot.
- If you want to have your own SMTP server, you will need a system administration with great knowledge about mail servers and the SMTP protocol.
- Programming SMTP client software is not as trivial as programming a REST Web Service client.
- Normally, when using your provider's SMTP server, you have a limit of how many emails you can send.
- You can't easily track wether the user received or clicked on a received message.
- It is not easy to guarantee that the sent email will not fall into customers SPAM box.

6.3 Problem

Email is still very important communication tool[19]. If you want to send relevant notifications to your users, it is expected that you send them emails with such notifications. If your users do not receive these emails properly, they will have their business impacted negatively, and so do your business. Moreover, if these emails do not have a clear and well presented message, users will not read them or will not be able to perform the actions you need them to do.

If you cannot track whether the users opened a given email, you cannot be sure they received the message, gave proper attention to it or received the message in their spam box.

6.4 Solution

Use cloud mail delivery services that provide easy-to-use REST API.

6.5 Resulting Context/Consequences/Side-effects

When using a cloud based email provider, you have more guarantees that your emails will be properly delivered. Furthermore, with the provider delivery reports, you will be able to understand your users behavior when receiving emails. You could, for example, reschedule a message send if you notice that the user did not read the previous one.

Cloud based email services usually give you the possibility to reserve IP addresses for the message servers, so you can gain credibility of spam checker tools over time, avoiding your messages falling into spam boxes.

These services also provide email template tools to customize your messages to have the same user interface. This is a good way for your users to develop identification with your brand.

You will not need to maintain SMTP servers and have a highly available solution for them. Moreover, you don't need to worry about scalability either. If your demand on message sending grows, the only thing you will need to do is send more requests to the cloud email providers. They will deal with scalability for you.

Besides these advantages, you will certainly have cost reductions, because most of these services have very economic prices for email sending, varying from US$0.10/thousand emails (e.g. Amazon SES) to US$0.80/thousand emails (e.g. most expensive package in SendGrid)

6.6 Rationale

REST APIs are a very common way to integrate systems. Developers are very used to provide and consume REST APIs. There are a lot of libraries in most of programming languages that makes easy to integrate with REST Web Services with a few lines of code.

6.7 Known Uses

The known cloud services providing email delivery are:

- Amazon SES
- SendGrid
- Mandrill

Some examples of companies that use these services are Pinterest, FourSquare, Spotify, Pandora, Eyejot, Greplin, Talkbox, MediaNet/Prise, etc.

6.8 Related Patterns/See also

Even if you use a third party API for email message delivery, it is expected that these services have a variable response time. In the worst case, these services can be unavailable for some time during the day. You do not want to lose messages, nor want your end user to wait so long for a third party service. One way to avoid this problem is to use asynchronous processing, instead of calling the mail service API directly from your application. Enqueue the API call using a queue based service (Section 3). The queue will deal very well with availability, performance and stability issues presented by the third party service.

6.9 Example

SendGrid is one of the biggest email delivery service and provides both REST and SMTP. Here is the code example in Ruby of how to integrate your application with SendGrid mail delivery [23]:

```
require 'mail'
Mail.defaults do
  delivery_method :smtp,
   { :address   => \smtp.sendgrid.net",
     :port      => 587,
    :domain    => \yourdomain.com",
    :user_name => \yourusername@domain.com",
    :password  => \yourPassword",
    :authentication => 'plain',
    :enable_starttls_auto => true }
end

mail = Mail.deliver do
  to 'yourRecipient@domain.com'
  from 'Your Name <name@domain.com>'
  subject 'This is the subject of your email'
  text_part do
    body 'Hello world in text'
  end
  html_part do
    content_type 'text/html; charset=UTF-8'
    body '<b>Hello world in HTML</b>'
  end
end
```

Besides mail delivery, SendGrid provides and whole set of reports about the API calls and your users behavior (email opening rates, bounce rates, delivery amount, etc).

Another easy to use service is Amazon SES. Here is a Scala code example using Amazon Simple Email Service (SES) module for Play 2.0 [8]:

```
Ses.sendEmail(Email(
  subject = "Test mail" ,
  from = EmailAddress("John", "john@john.com"),
  replyTo = None,
  recipients =
    List(Recipient(Message.RecipientType.TO,
         EmailAddress("Daniel",
                      "danicuki@ime.usp.br"))),
  text = "text",
  htmlText = "htmlText",
  attachments = Seq.empty))
```

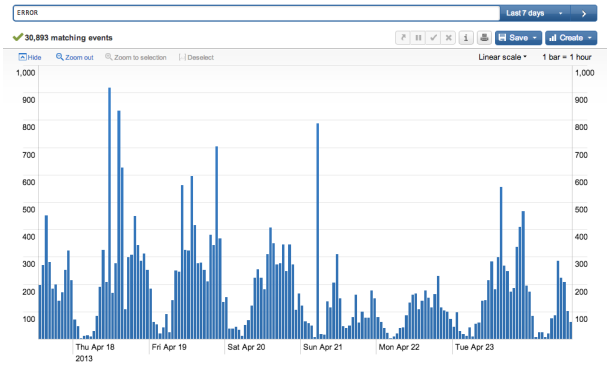

Figure 4. Errors/hours graph based on logs

7. Logging

7.1 Context

Your complex application is deployed among many servers. The application generates a lot of log files, each of them with thousands of lines. These logs have information that would be very useful for you. You don't want all members of your team logging into production servers to read logs. Moreover, you want all the logs to be consolidated in a single point. You want to find patterns in the logs, make custom searches on it and have graphs of the logs evolution along time.

7.2 Motivation

- Manually copying log files demands a lot of work
- If you will maintain all your logs, you need to take care of disk usage
- If your logs are huge, you will need to index them to make efficient searches

7.3 Problem

Your developers need to quickly check logs without losing time searching for what they need. If you have any anomaly in your system and log files indicate this, you do not want to take hours to identify the problem. You want to immediately be notified. Searching and indexing huge amount of big text files is not trivial. You do not want to develop a system for searching logs or aggregate logs from many servers into a single repository. Moreover, logs require a lot of infrastructure and you do not want to spend your time managing log files infrastructure.

7.4 Solution

Use a cloud based log service to consolidate your application logs.

7.5 Resulting Context/Consequences/Side-effects

By using a cloud based log service, you will have easy access to all your logs in a centralized system. You can make fast searches or filter your logs for specific terms. You can easily build graphs that shows trends about your system and problem hot spots (Figure 4). Moreover, you can create alerts based on thresholds (e.g. if errors log is greater than 10 items per minute for more then 5 minutes).

7.6 Rationale

In the first version of our architecture, we had to manage only two servers. When we needed to look at log files, we accessed these

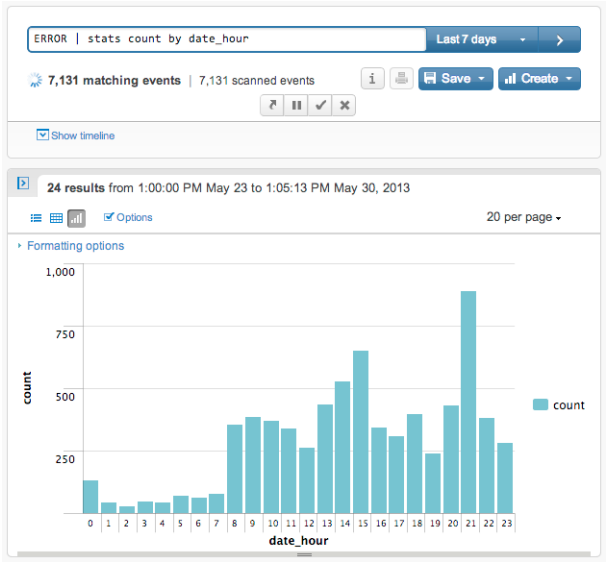

Figure 5. Graph based on logs

servers and "grep" the log files from the terminal or downloaded the files and filtered them locally. Our work with logs was limited by these simple command line operations. When log files started to rotate, we would have to join them back together when we wanted to search or group historical information.

Besides the limitation of working with raw files and bash command line tools [9], our infrastructure started to grow to more than twenty servers, each one with its own log files. Moreover, some servers would live for one or two days, and then be shut down because we did not need them anymore. We could not access log files in dead servers, so historical log data would be lost forever.

When we started to use cloud based log services, we solved all these problems. All servers now can send data to the centralized log service, and our developers can search for specific log entries from a easy-to-use web interface. And not only developers, but also product people and the customer support staff can have access to logs. Nobody needs to have access to production servers to watch relevant log information.

Sometime ago, we noticed that we were having problems with our database at specific moments of the day. We suspected that there were some background processes that were consuming a lot of database resources, but we did not know exactly at what times these processes were running. We decided to look in the log service. We did a simple query:

```
ERROR | stats count by date_hour
```

We noticed that exactly at 9pm the number of errors more than doubled compared to other hours. We found out the process that were causing these problems and optimized it to avoid the database errors (Figure 5).

7.7 Known Uses

Splunk is well known to be a very powerful log analyzing software. In the beginning, to use Splunk, you would need to install the server software in your own infrastructure and pay for a very expensive license. Today, Splunk company offers cloud based log server called Splunk Storm [15]. Other two known similar services are Loogly [3], Logentries [2] and Logstash [4].

Cloud based log services are being used to generate semi-structured time series database [6] or structured log analysis [16]. They can also be used as online visualization system for computing clusters [28].

7.8 Related Patterns/See also

Since logs are generated almost in the same time things are happening in your system, you can eventually use log graphs to monitor business metrics in realtime (Section 8). For example, in our application, when a user adds an item to a cart, we log "[INFO] item X added to cart". Our cloud based log can easily create a graph showing how many items are added to the cart per minute.

If you have a huge amount of background processing using queues, you want to monitor the progress of this processing. So, using cloud based log together with queue based processing is a good way to have control and metrics about your asynchronous jobs (see Section 3).

Some PaaS providers like Heroku offers cloud based log integration add-on. If you prefer running your application over PaaS instead of IaaS (Section 4), you can also have the benefit of using cloud based logs.

7.9 Example

Splunk [26] is a well known log aggregation software. You can use it installing a Splunk server by yourself, or by integrating with Splunk Storm Cloud Service. There are many ways to integrate your application with Splunk Storm. One of them is by configuring your rsyslogd service to send the log files to Splunk Storm [15] via UDP packages. All you need to do is to add these lines to your /etc/rsyslogd.conf file:

```
$ModLoad imfile
$InputFileName /var/log/nginx/error.log
$InputFileTag nginx:
$InputFileStateFile stat-nginx-error
$InputFileSeverity error
$InputRunFileMonitor

$InputFilePollingInterval 10

*.* @@logsX.splunkstorm.com:20000
```

8. Realtime User Monitoring (RUM)

8.1 Context

You have a complex web application being used by hundreds or thousands of users. Your need to deploy new features every day, so your system is constantly evolving. You need to make sure that every new deployment to production does not break the existing features or insert new bugs, specially for business critical operations.

Furthermore, you need to know how your users behave at specific times during the day (e.g. do people prefer buying online after lunch or after dinner). You want to know when you have an uncommon visiting pattern (e.g. why today at 10am people are adding more products to the cart then usual)

8.2 Motivation

- Complex big systems are difficult to have a 100% test coverage
- Even when having a complete test suite in staging environment, features can still fail in production environment, because creating a staging environment that is perfectly identical to production is cost prohibitive or to hard to do

8.3 Problem

Sometimes, when you deploy a bug into production, it takes a long time for you to realize that this bug is affecting your users. Moreover, you normally come to know that this bug is there from the users themselves, which is really bad for your business.

Unexpected behaviors sometimes demand you to provide extra infrastructure resources to keep your service available, without outages or performance degradation.

It is difficult to measure the impact of a marketing campaign in you business, because you cannot observe all your users behavior in realtime when the campaign is active.

8.4 Solution

Use a tool that provides you Realtime User Monitoring (RUM). Choose to monitor business metrics, instead of infrastructure resources metrics.

8.5 Resulting Context/Consequences/Side-effects

When our engineering team started to grow, the number of new features developed also increased. As we started to deploy new features every day, the chance of breaking and inserting bugs into production increased. We did not want our users to be the first to notice and report the bugs. When we started to look at *statsd* after every deploy, after no more than 5 minutes we could be able to notice any anomaly in the system for critical business operations like user logins, order submissions, user sign-ups and so on.

We started to notice web bots access peaks, user login behavior, new product creation patterns and so on. Now, we have a dashboard in the office showing these graphs. We monitor them throughout the day and we can take a quick action if someone notices any abnormal pattern in the graphs. Users are less affected by these problems and this impacts positively ii our business.

8.6 Rationale

System administrators are used to use monitoring tools like Nagios or Zabbix to monitor the servers infrastructure. This is essential for their operations, but not enough from the business point of view because sysadmins usually monitor CPU, memory, network and other low level metrics. When these low level metrics are healthy, it does not mean that the system is running properly. Sometimes a bug or an external factor can cause sales loss. This cannot be tracked by low-level infrastructure monitoring.

In the end, what means for your business is that your key business indicators are going well. That is, indicators like number of sales, successful transactions, new users registries, etc. This does not mean you will neglect your infrastructure monitoring. You should do both, since the look at your business in different granularities. Infrastructure monitoring is to Unit Tests what Real User Monitoring is to acceptance tests.

8.7 Known Uses

Etsy, well known craft marketplace platform, developed "statsd" as open source[11] software. They used it to monitor their own users. Etsy is a company that has as a principle to "Measure Anything, Measure Everything"[10].

Another very simple but powerful realtime monitoring tool is Pingdom, which is used by many famous e-commerce companies like eBay, Amazon, BestBuy, Shopify, as well as technology or communication companies like Apple, Microsoft, AT&T, etc. With Pingdom, you configure a specific service (by providing an URL) that you want to monitor. Their service is to "ping" the URL every minute and measure whether the server is responding, and also it response time. Then Pingdom records this data in graphs, so you can easily know your uptime, downtimes, and mean response time

Figure 6. Realtime User Monitoring with statsd

over the month. You can use these metrics as a base for your SLA with your customers.

New Relic is a service that provides both low level infrastructure monitoring as well as end user realtime monitoring. With this tool, you can track your web server end-user response time, as well as ApDex

8.8 See also

It is possible to use log files to do real user monitoring. You can save all critical or important events from your system to log files, and then graph these log files events using a cloud based log system (See section 7).

8.9 Example

We had a good experience using statsd to monitor realtime user data. We run a e-commerce marketplace platform, so our monitoring dashboard has graphics "add to cart" actions, product searches, home visits, logins, product creation, etc (Figure 6). Statsd has client implementations for most of popular languages [17] and it has the additional advantage of using UDP to send packages, so network traffic for your monitoring does not compromise your application end user.

To deploy a statsd server, you need to install Node.js as well as a Graphing backend service like Graphite [1].

9. Conclusions

When an Internet startup starts to grow in number of users and key transactions, scaling the business online platform can be very challenging. In this article, we covered some practical aspects of how to scale an Internet application. By having a good Load Balancing strategy (Section 5) for your web servers, you guarantee that the website will remain available and with good performance. You do not want to spend time managing large amount of static files, therefore using a cloud based storage (Section 2) will facilitate your job.

Monitoring (Section 8) your business metrics is essential for the decision making process. You will also need to understand what is going on with your infrastructure, so you want to have easy access to log files data (Section 7).

Every Internet business must deliver emails in some part of its business. Have your emails in good hands by using cloud based email delivery service (Section 6). For time consuming jobs or asynchronous processing, use queues (Section 3) and you will have a very robust and scalable solution.

When infrastructure is not your strength and you want to focus your developers team on business related features, use PaaS instead of IaaS (Section 4).

This article is far from being a complete catalog of DevOps patterns to scale web applications using cloud services. But, using these solutions together is a good start point for scaling your online business. We did not cover very important topics like continuous delivery, deployment, configuration management, software quality process and tests. These are some issues that could complement this catalog to form a Pattern Language with DevOps practices to scale web based business using the cloud.

Acknowledgments

First, I would like to thank Luis Artola for his help on the shepherding process and Eduardo Guerra for his attention over this paper initial submission. All the engineers in Elo7 also deserve a big thanks, for helping me to develop and deploy rocking technologies.

References

[1] Graphite. Available at: `http://graphite.wikidot.com/`, 2013. Accessed in: 05/05/2013.

[2] Logentries. Available at: `http://logentries.com/`, 2013. Accessed in: 05/05/2013.

[3] Loggly. Available at: `http://loggly.com/`, 2013. Accessed in: 05/05/2013.

[4] Logstash. Available at: `http://logstash.net/`, 2013. Accessed in: 05/05/2013.

[5] Amazon.com. Amazon web services case studies. Available at: `https://aws.amazon.com/solutions/case-studies/`, 2013. Accessed in: 05/05/2013.

[6] L. Bitincka, A. Ganapathi, S. Sorkin, and S. Zhang. Optimizing data analysis with a semi-structured time series database. In *SLAML'10: Proceedings of the 2010 workshop on Managing systems via log analysis and machine learning techniques*, pages 7–7, 2010.

[7] T. Bourke. *Server load balancing*. O'Reilly Media, Incorporated, 2001.

[8] J. Brünemann. Amazon simple email service (ses) module for play 2.0. Available at: `https://github.com/ Rhinofly/ play-libraries/`, 2013. Accessed in: 05/05/2013.

[9] A. Chuvakin, K. Schmidt, and C. Phillips. *Logging and Log Management: The Authoritative Guide to Dealing with Syslog, Audit Logs, Events, Alerts and other IT 'Noise'*. Syngress, 2012.

[10] Etsy. Measure anything, measure everything. Available at: `http://codeascraft.etsy.com/2011/02/15/measure-anything-measure-everything/`, 2011. Accessed in: 05/05/2013.

[11] Etsy. Statsd. Available at: `https://github.com/ etsy/statsd/`, 2013. Accessed in: 05/05/2013.

[12] Google. Google cloud platform. Available at: `https://cloud.google.com/customers/`, 2013. Accessed in: 05/05/2013.

[13] M. Grotzke. memcached-session-manager: Tomcat high-availability clusters with memcached. Available at: `https://code.google.com/p/memcached-session-manager/`, 2013. Accessed in: 05/05/2013.

[14] Heroku. Heroku success. Available at: `http://success.heroku.com/`, 2013. Accessed in: 05/05/2013.

[15] S. Inc. Splunk storm user manual. Available at: `http://docs.splunk.com/ Documentation/ Storm/`, 2013. Accessed in: 05/05/2013.

[16] D. Jayathilake. Towards structured log analysis. In *Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on*, pages 259–264. IEEE, 2012.

[17] D. Josephsen. Changing the game, part 2. *;login:*, pages 55–59, Feb 2012.

[18] J. Murty. *Programming Amazon Web Services: S3, EC2, SQS, FPS, and SimpleDB*. O'Reilly Media, 2009.

[19] S. Radicati and Q. Hoang. Email statistics report, 2011-2015. *Retrieved May*, 25:2011, 2011.

[20] W. Raisch and G. Foreword By-Gartner. *The eMarketplace: Strategies for success in B2B eCommerce*. McGraw-Hill Professional, 2000.

[21] M. Sacks. Devops principles for successful web sites. In *Pro Website Development and Operations*, pages 1–14. Springer, 2012.

[22] M. Sacks. *Pro Website Development and Operations: Streamlining DevOps for large-scale websites*. Apress, 2012.

[23] SendGrid. Getting started with sendgrid. Available at: `http://sendgrid.com/docs/`, 2013. Accessed in: 05/05/2013.

[24] D. M. Smith. Hype cycle for cloud computing, 2011. *Gartner Inc., Stamford*, 2011.

[25] B. Snyder, D. Bosnanac, and R. Davies. *ActiveMQ in action*. Manning, 2011.

[26] J. Stearley, S. Corwell, and K. Lord. Bridging the gaps: joining information sources with splunk. In *Proceedings of the 2010 workshop on Managing systems via log analysis and machine learning techniques*, pages 8–8. USENIX Association, 2010.

[27] A. Videla and J. J. Williams. *RabbitMQ in action*. Manning, 2012.

[28] J. Xia, F. Wu, F. Guo, C. Xie, Z. Liu, and W. Chen. An online visualization system for streaming log data of computing clusters. *Tsinghua Science and Technology*, pages 196–205, April 2013.