Future of DevOps

Sasha Rosenbaum
@DivineOps
Sasha Rosenbaum
@DivineOps

Dev
Ops
Product
Sales
Consulting
The Past
1990s:

Getting a new server up:
3+ months
Backup
William Herold
@willigula

Replying to @DivineOps

I don't know, but I have misguided nostalgia for getting a page & driving to our data center to physically restart a server in the middle of the night.
<table>
<thead>
<tr>
<th>Date</th>
<th>Release name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1990</td>
<td>SQL Server 1.1 (16-bit)</td>
</tr>
<tr>
<td>1992</td>
<td>SQL Server 4.2A</td>
</tr>
<tr>
<td>1993</td>
<td>SQL Server 4.21a</td>
</tr>
<tr>
<td>1995</td>
<td>SQL Server 6.0</td>
</tr>
<tr>
<td>1996</td>
<td>SQL Server 6.5</td>
</tr>
<tr>
<td>1998</td>
<td>SQL Server 7.0</td>
</tr>
<tr>
<td>2000</td>
<td>SQL Server 2000</td>
</tr>
<tr>
<td>2003</td>
<td>SQL Server 2000 64-bit</td>
</tr>
<tr>
<td>2005</td>
<td>SQL Server 2005</td>
</tr>
<tr>
<td>2008</td>
<td>SQL Server 2008</td>
</tr>
<tr>
<td>2010</td>
<td>Azure SQL database</td>
</tr>
</tbody>
</table>

Software release cadence:
2-3-year cycle
Merge hell

Merging the development branches and completing the test procedures could take months
GIT MERGE
Every company ran its own email
Software as a Service

- Hotmail 1996
- Google Search 1998
- Salesforce 1999

Revenue share in 1999
- Application Service Providers $933M
- Software Applications $74B

Source: https://www.inc.com/magazine/20000401/18093.html
Deployment Checklists
27% of server market

- File-based OS
- Maintains configuration in files
- Every device is a file

41% of server market

- Executable-based OS
- Maintains configuration in registry
- Every device has a different driver mechanism
1990s:

Maintenance windows
How many 9s is two weekends like that?

< 99%

3.65 days / year
Unavailable systems were estimated to have cost American businesses $4.54B in 1996.
Traditional IT

dev

wall of confusion

ops
Dev
Speed

Ops
Reliability
"I knew this organization was full of siloes. I just didn't think they'd be so heavily defended!"
Darmok and Jalad at Tanagra
Patrick and Andrew at Agile TO 2008
10 deploys per day: Dev and Ops collaboration at Flickr

Velocity 09: John Allspaw and Paul Hammond
Andrew Clay Shafer 雷启理

@littleidea

Don't just say 'no', you aren't respecting other people's problems... #velocityconf #devops #workingtogether
DevOpsDays Ghent 2009: Patrick Debois
devopsdays events

Bridget Kromhout

DevOpsDays ’09
“DevOps” Google Search Trend 2009-Present
Jez Humble and Dave Farley: 2010
Continuous Integration (CI)

The practice of merging code into the main branch several times a day; and automating the build and testing on every commit.
Continuous Delivery (CD)

The approach in which teams produce software in short cycles, ensuring that the software can be reliably released at any time.
ELITE PERFORMERS

Comparing the elite group against the low performers, we find that elite performers have:

- 208 TIMES MORE frequent code deployments
- 106 TIMES FASTER lead time from commit to deploy
- 2,604 TIMES FASTER time to recover from incidents
- 7 TIMES LOWER change failure rate (changes are 7x as likely to fail)

Nicole Forsgren. State of DevOps Report 2019
Software delivery is like a muscle. The more you use it, the stronger it gets.
The Present
Effective automation requires consistent APIs.
OS-level APIs
PowerShell

(Windows) configuration management framework and scripting language

Jeffrey Snover, 2006
Infrastructure-level APIs
Amazon Web Services: 2002
Amazon Cloud Computing: 2006
Azure Cloud Services: 2008
Infrastructure as code
Every wave of automation

Enables the next wave of automation
Today
We have a lot more automation
We have a lot more **automation tools**
We have much higher availability
We have better on-call
We have better incident response
We deploy a lot more frequently
BUT
Many of us are bad at monitoring
Many of us are bad at CI
Continuous Integration (CI)

The practice of merging code into the main branch several times a day; and automating the build and testing on every commit.
You cannot CI/CD Without Testing
We have convinced people that deploying faster is good.

We forgot to mention that increasing the Operational Burden is bad.
SRE ≈ Google’s implementation DevOps
Why did SRE develop?
SaaS Market
2008-2020
What is the most important thing about the SRE discipline?
SRE is about **explicit agreements** that align incentives.
SLA, SLI, SLO
SLA

= 

Financially-backed availability
<table>
<thead>
<tr>
<th>Monthly Uptime Percentage</th>
<th>Service Credit Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less than 99.95% but equal to or greater than 99.0%</td>
<td>10%</td>
</tr>
<tr>
<td>Less than 99.0% but equal to or greater than 95.0%</td>
<td>25%</td>
</tr>
<tr>
<td>Less than 95.0%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Monthly downtime > 15 days means 100% refund
SLAs are about aligning incentives between Vendor & Customer
SLO

= 

Targeted reliability
While SLA is a single metric,

SLOs are a system of metrics
SLO

= 

Business-approved reliability
SLOs are about explicitly aligning incentives between Business & Engineering
Error Budgets

Acceptable level of unreliability

Error budget = 1 - SLO

\[ \text{EB} = 1 - 99.99\% = 0.01\% \approx 13 \text{ mins /quarter} \]
Error budgets are about aligning incentives between Dev & Ops.
If developers are measured on the same SLO, then when the error budget is drained, developers shift focus from delivering new features to improving Reliability.
SLI = Actual reliability
Monitoring
Without monitoring, you have no way to tell whether your service even works!
Good Monitoring
Without **good monitoring**, you don’t know that the service does what users expect it to do!
The Future
Something we have to talk about
AI
AI will not [yet] take your job
AI will change jobs
Ethics
Like a Library
Current AI aggregates and synthesizes the results of decades of human work.
Written works, code, art
Unlike a Library
Current **AI** charges its users for the content

But it **DOES NOT** pay royalties or give **attribution** to the creators
Fairness usually loses to profit
Are people going to stop sharing?
Something else we have to talk about
DevOps is Dead
Is it though?
Shades of DevOps Industry Terms

Note: All practices are equally important, yet some seem to be hipper than others

Patrick Debois, 2021
Someone in your org is keeping your systems alive

And their life might be hell
The future is already here.
It’s just not evenly distributed

~ William Gibson
Renaming Teams

Support -> Sys Admin -> Ops -> DevOps -> SRE -> Platform Eng (?)

wall of confusion
Technology Adoption Life Cycle

- **Innovators**: 2.5%
- **Early Adopters**: 13.5%
- **Early Majority**: 34%
- **Late Majority**: 34%
- **Laggards**: 16%

- **Chasm**

**Left side**: seeking advantage

**Right side**: seeking legitimacy
words cross the chasm before understanding and practice
Most organizations I encounter, even those running SaaS, have no SRE practice.
Most Organizations today

- Don’t **merge** code frequently
- Don’t have quality **testing** during CI
- Don’t have informative **monitoring**
- Don’t enforce **Error Budgets**
- Don’t have a **platform**
everyone wants DevOps

well actually...
what they really want

- reliability
- availability
- scalability
- operability
- usability
- observability
- all for free
- without changing anything
without changing anything
without changing anything
without changing anything
Everyone wants this
When they haven’t solved this
Or even this
Everyone wants to deliver SaaS optimally delivered on PaaS

also, Software dependent on IaaS

Also, Software
Let’s bring the future to everyone
We don’t need new words

We need to implement what we already know to work well
Let’s start small
The future is already here.
It’s just not evenly distributed

~ William Gibson
Thank you!

Sasha Rosenbaum

@DivineOps