electronic document CONFERENCE

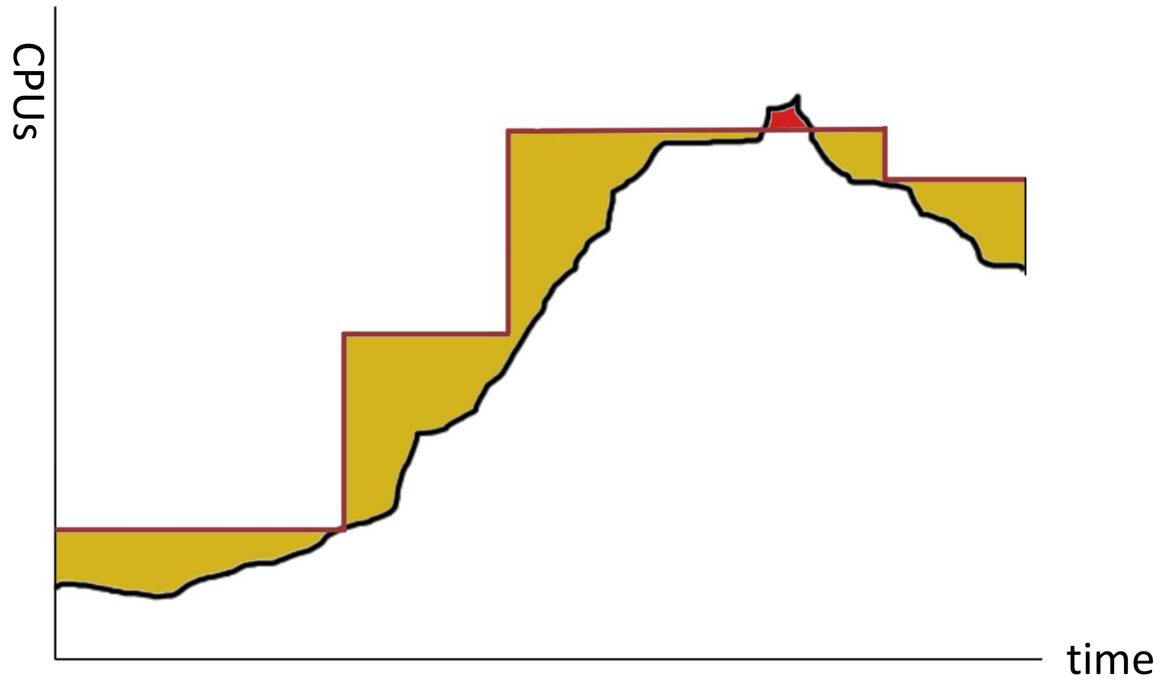
Serverless eStatements

Millions of PDFs once a month

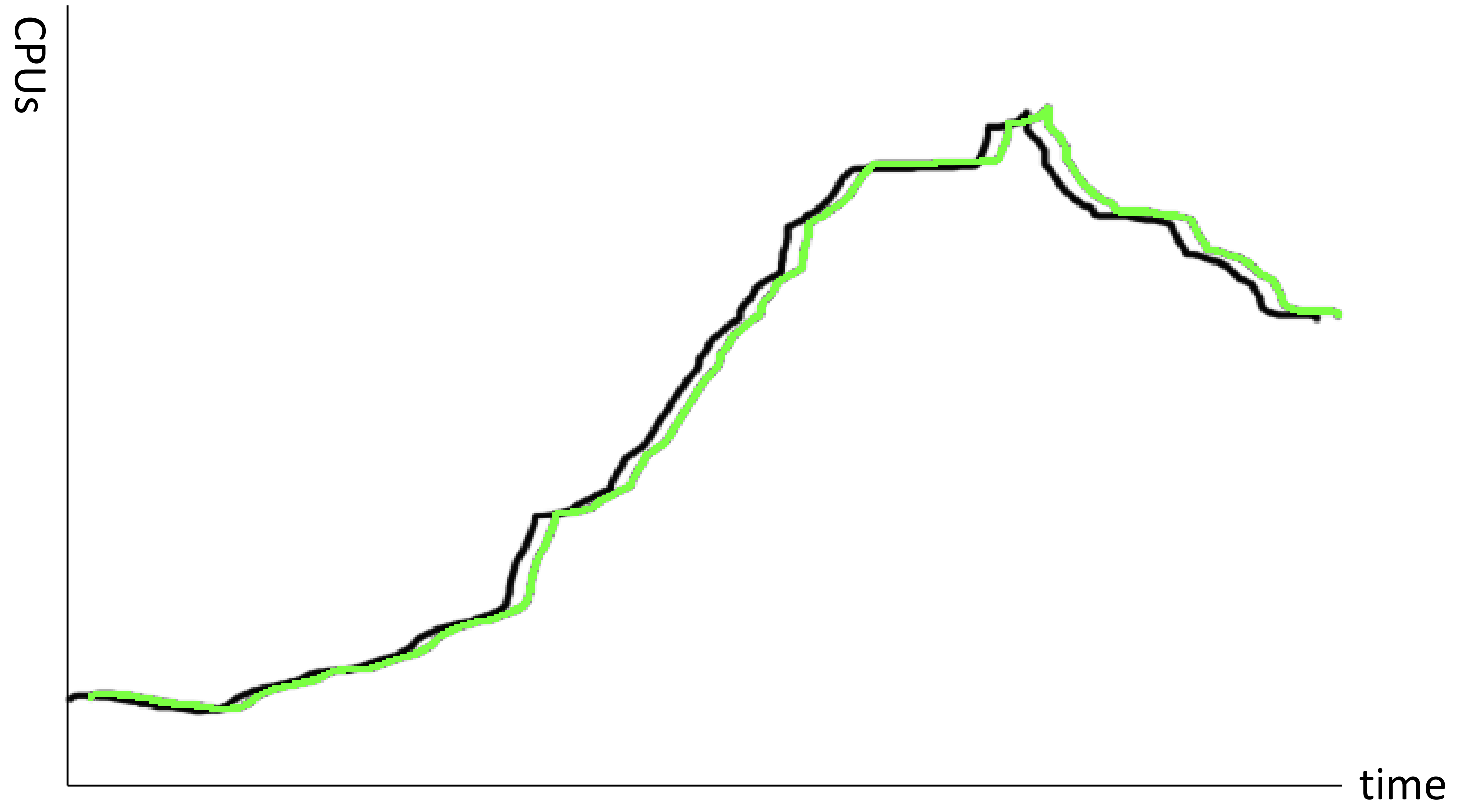
NATIVE DOCUMENTS

The Problem

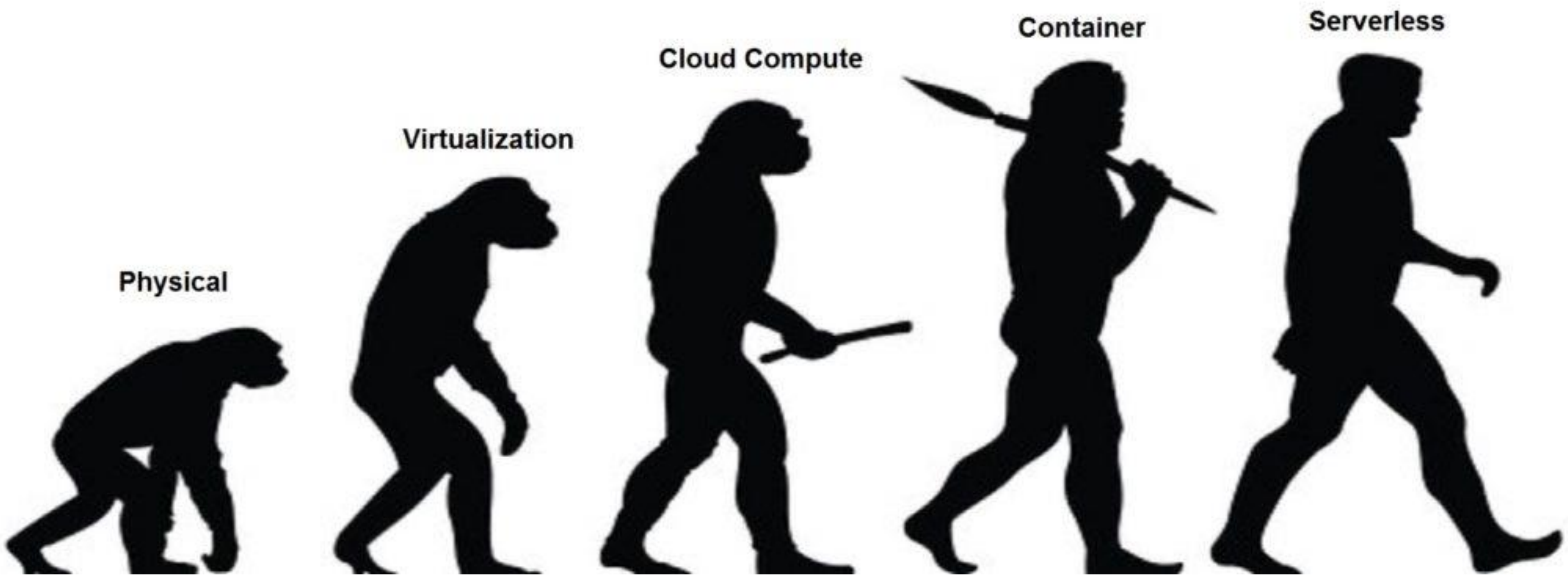
- Docx to PDF conversion is CPU intensive
  - Depends on “complexity” (number of pages, tables, page breaks etc)
- so producing lots has meant provisioning servers
- and that’s expensive: hardware, software and people



The Solution: “serverless”



Serverless: its the next step in the evolution of computing



2007: Linux KVM

2018: Firecracker

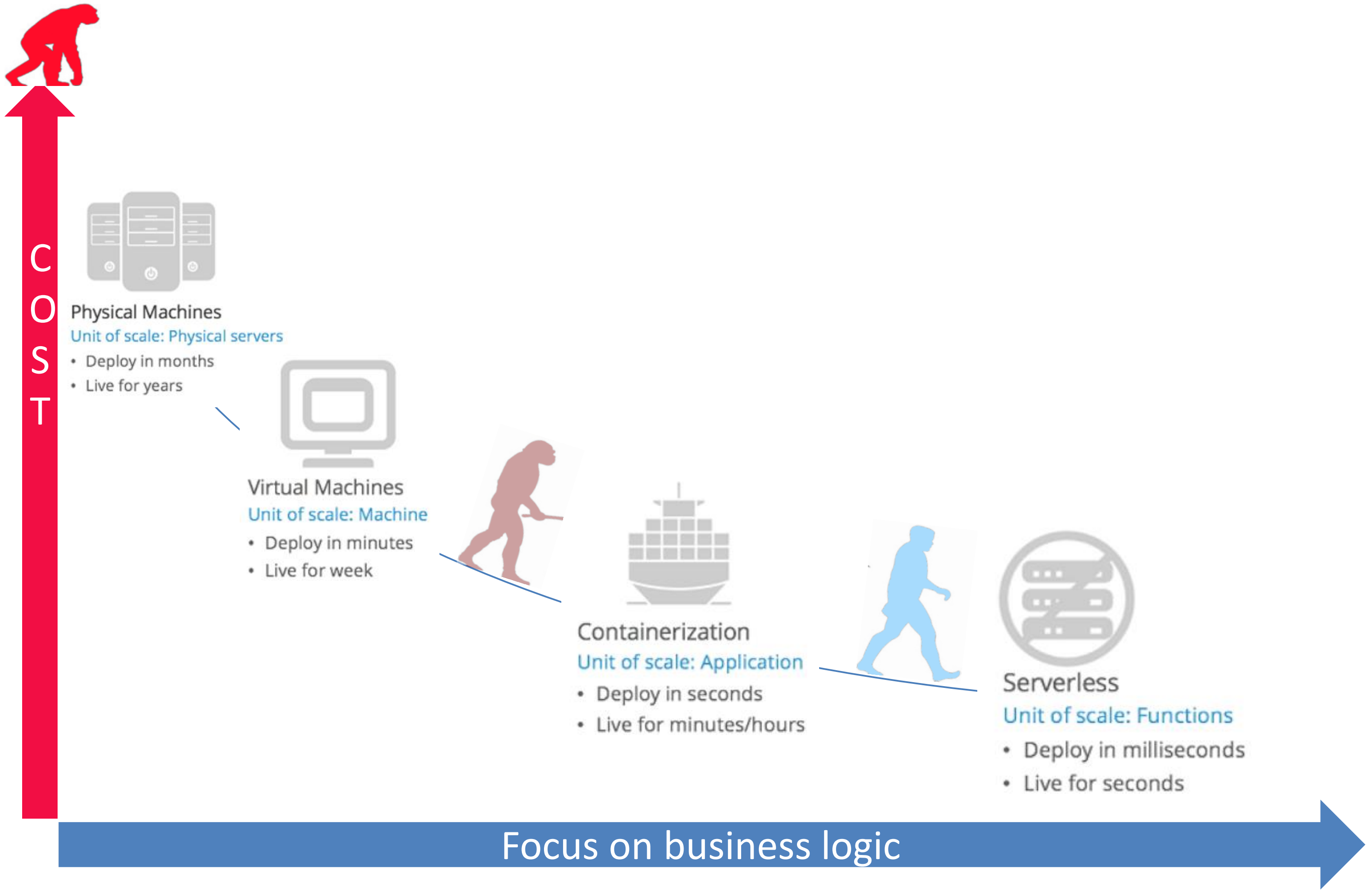
2006: EC2

2013: Docker

2015: Kubernetes

2014: AWS Lambda

Serverless: Focus on ***functions*** implementing your business logic



Adapted from source: Deloitte Consulting LLP

All the big providers now offer it

Walled Gardens:

- **AWS** Lambda
- **Azure** Functions
- **Google** Cloud Functions
- IBM, Oracle…

Cross platform

- Serverless Framework

Serverless deployment of Docker containers:

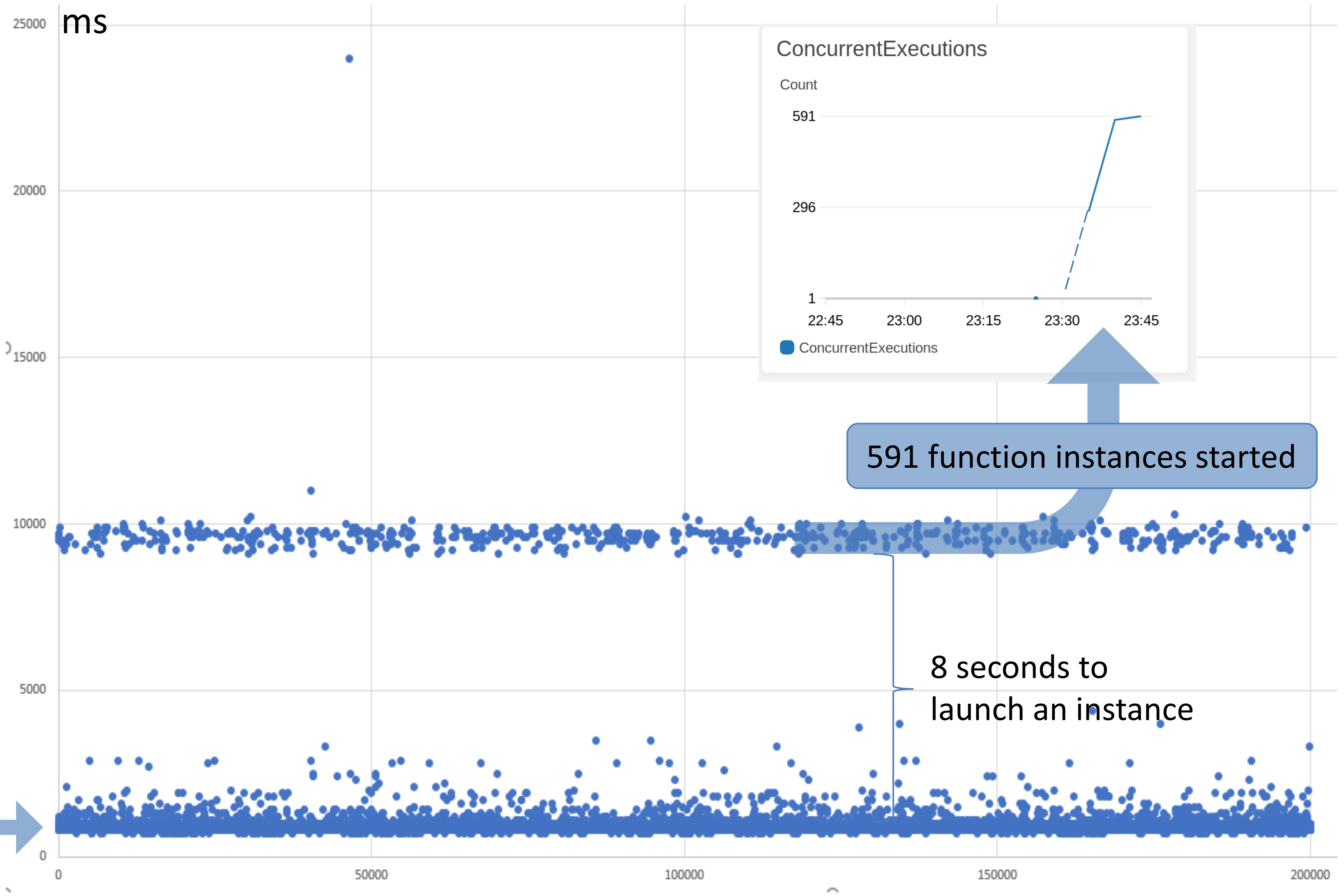
- OpenWhisk (deploy anywhere)
- For Kubernetes:
  - Deploy a Docker Container:
  - in Knative
    - OpenWhisk
    - OpenFaaS

Concepts as implemented in AWS Lambda

- Common concepts:
  - Trigger events
  - Scheduler executes Function
- Languages
  - Javascript (node.js)
  - Others, depending on platform
- Function packaging

- Lamba event triggers include:
  - Your REST API
  - S3
  - SQS (messages)
  - Step functions
  - *etc.*

200,000 PDFs in 10 mins with Native Documents on AWS Lambda

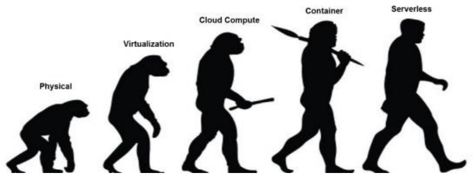


591 function instances started

8 seconds to launch an instance

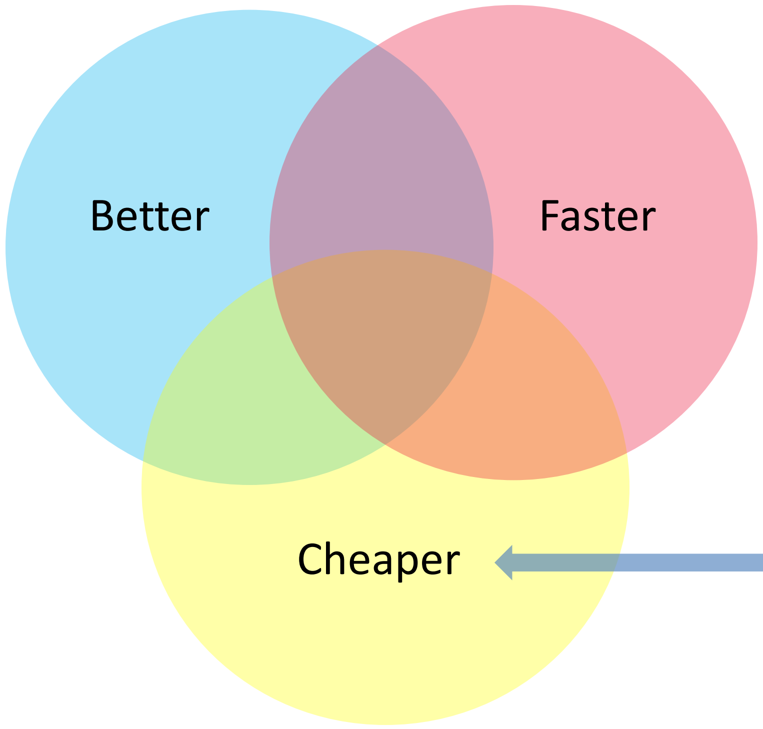
Average 1 sec per conversion

Compared to earlier paradigms



Problems overcome:

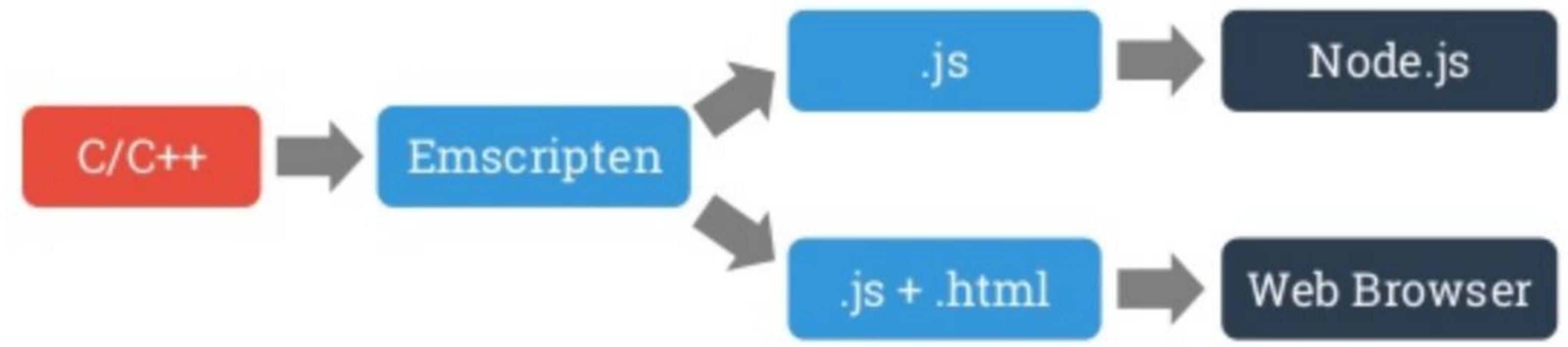
- Slow start time if a server instance needs to start
- Effort to configure server auto-scaling



AWS Lambda cost for those 200,000 PDFs: 175,000 seconds = ~$5

How? You need “serverless-ready” PDF Conversion code

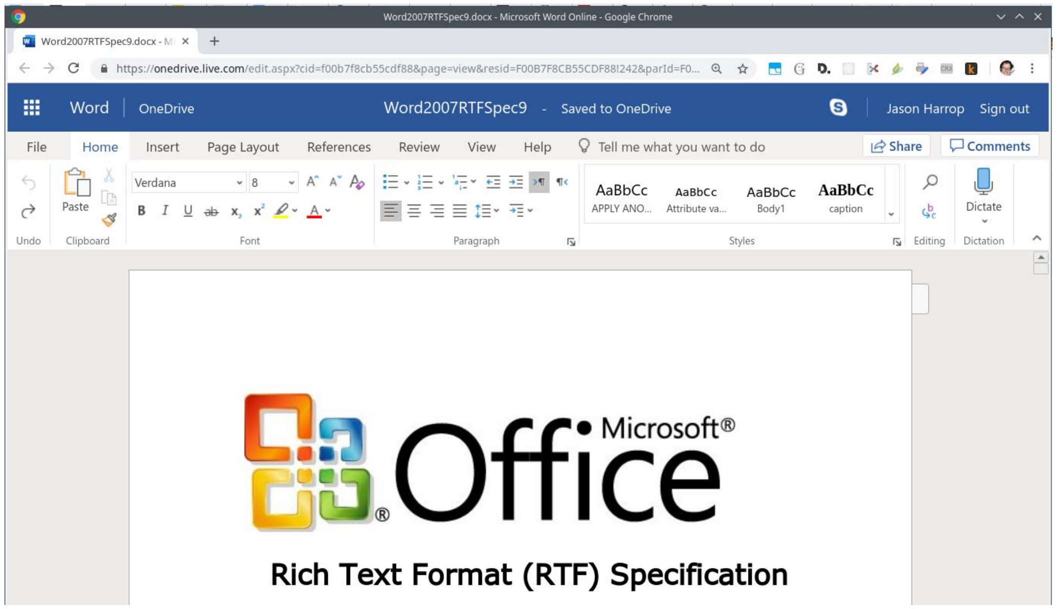
- Needs to be able to run on the platform:
  - Lowest common denominator is node.js
- Ours, for example works as follows
  - C/C++ code base
    - Proprietary doc/docx layout/editing engine
    - Exports PDF using Skia (Google 2D graphics lib) PDF backend
  - Converted to Web Assembly (wasm) using Emscripten



- https://www.npmjs.com/package/@nativedocuments/docx-wasm

Web Assembly is also a great foundation for editing Word docs

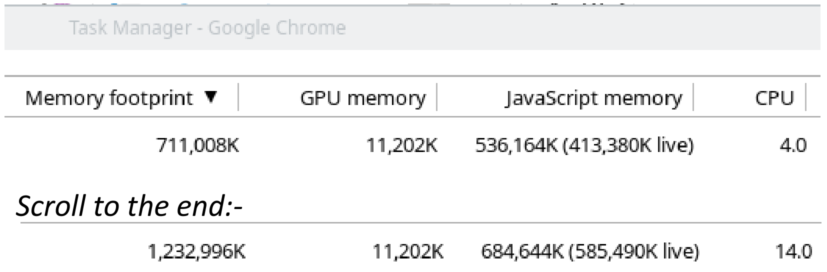
- Docx page layout
  - Must be done (accurately!) to generate a PDF
  - Done by Native Documents rendering engine
  - Rendering engine also used in Word File Editor
- Thanks to wasm, we run that rendering code in the browser
- Compared to Word Online (and Google Docs):
  - wasm approach uses under half the resources
  - User perceives better performance on long documents
- For example, the RTF spec:



Word2007RTFSpec9.docx - Microsoft Word Online - Google Chrome
https://onedrive.live.com/edit.aspx?cid=f00b7f8cb55cdf88&page=view&resid=F00B7F8CB55CDF88!242&parId=F0...
Word
OneDrive
Word2007RTFSpec9 - Saved to OneDrive
Jason Harrop
Sign out
File
Home
Insert
Page Layout
References
Review
View
Help
Tell me what you want to do
Share
Comments
Paste
Verdana
8
AaBbCc
APPLY ANO...
Attribute va...
Body1
caption
Dictate
Undo
Clipboard
Font
Paragraph
Styles
Editing
Dictation
Microsoft®
Office

Rich Text Format (RTF) Specification

In Word Online:-



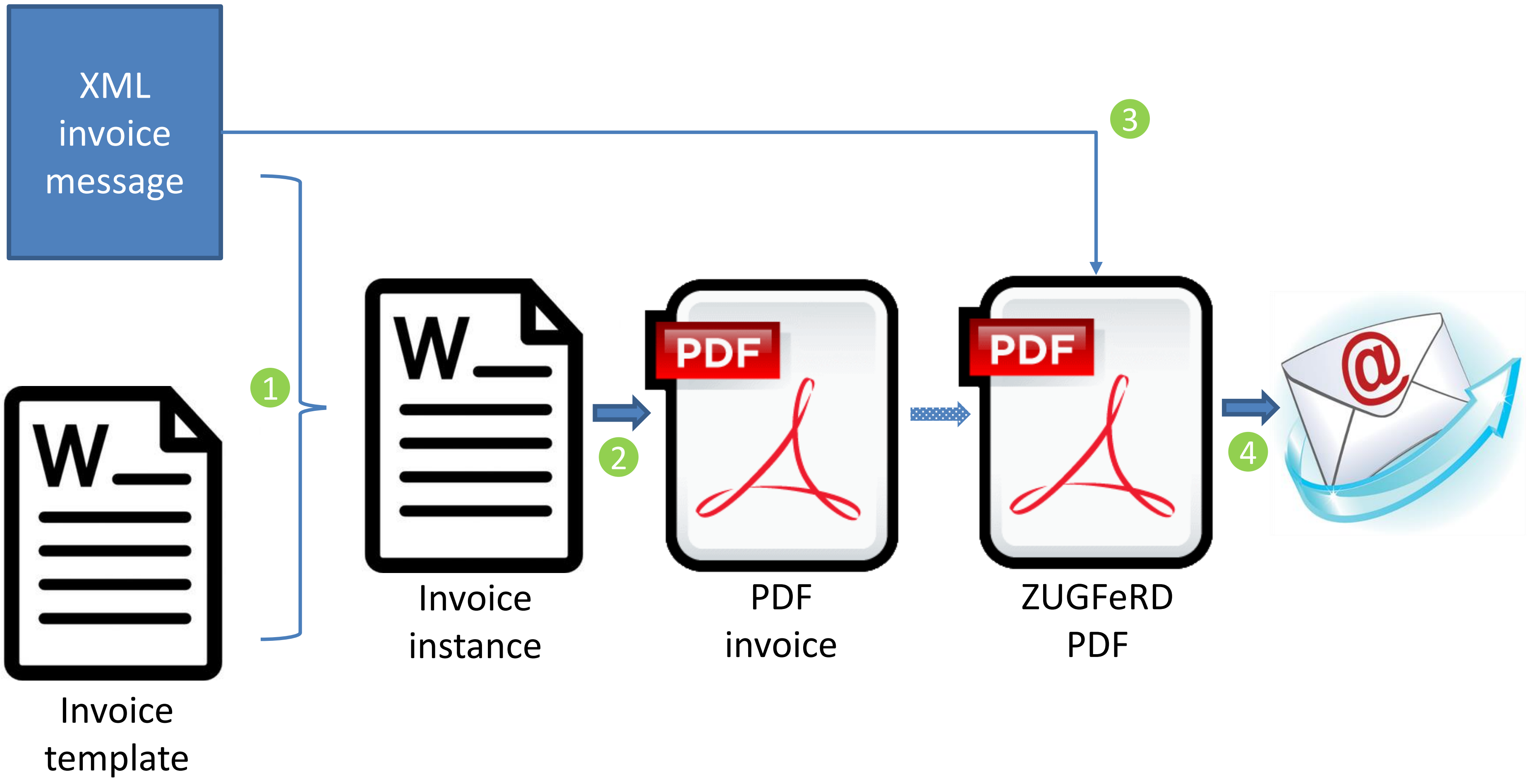
Task Manager - Google Chrome

Memory footprint ▼	GPU memory	JavaScript memory	CPU
711,008K	11,202K	536,164K (413,380K live)	4.0
Scroll to the end:-			
1,232,996K	11,202K	684,644K (585,490K live)	14.0

compare Native Documents:-

283,684K	48,148K (37,860K live)
Scroll to the end:-	
425,428K	92,180K (80,576K live)

Case study: High volume serverless ZUGFeRD PDF invoices



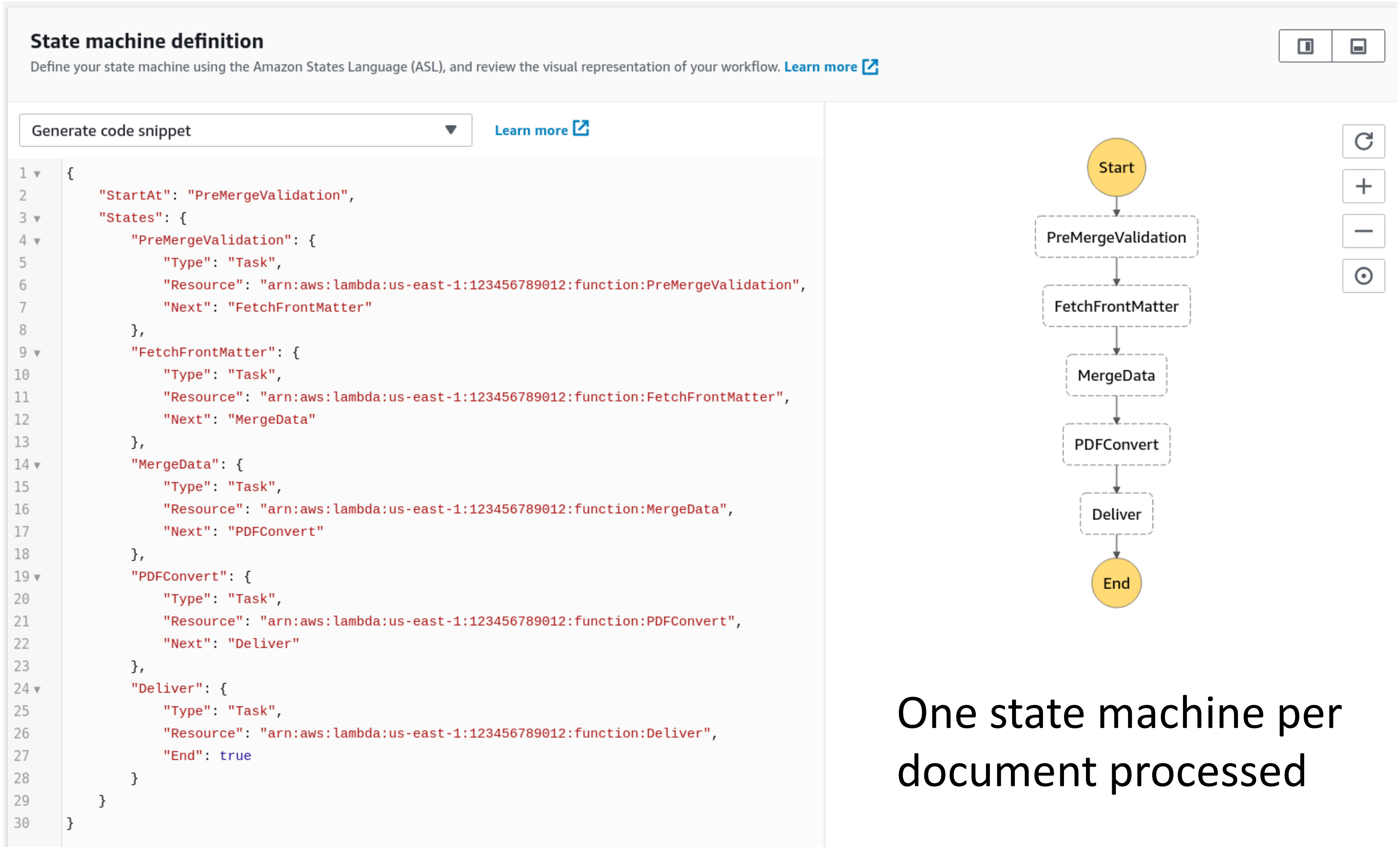
4 functions to be orchestrated here

But is eInvoicing really high-volume?

- Generally eInvoices are required in government procurement
  - Low volume for most sellers
  - Except intermediary hubs/services (billing service providers)
- (Compare eStatements)
- Increasingly corporate customers demand an eInvoice
  - Facilitates reconciliation/analysis of electricity bills
- Easy then for utilities to provide consumers with an eInvoice
  - Consumed by:
    - Personal finance software
    - Online banking
    - Government tax authority
  - A FinTech opportunity? Maybe..

NATIVE DOCUMENTS

How best to wire micro-services together?



State machine definition

Define your state machine using the Amazon States Language (ASL), and review the visual representation of your workflow. Learn more

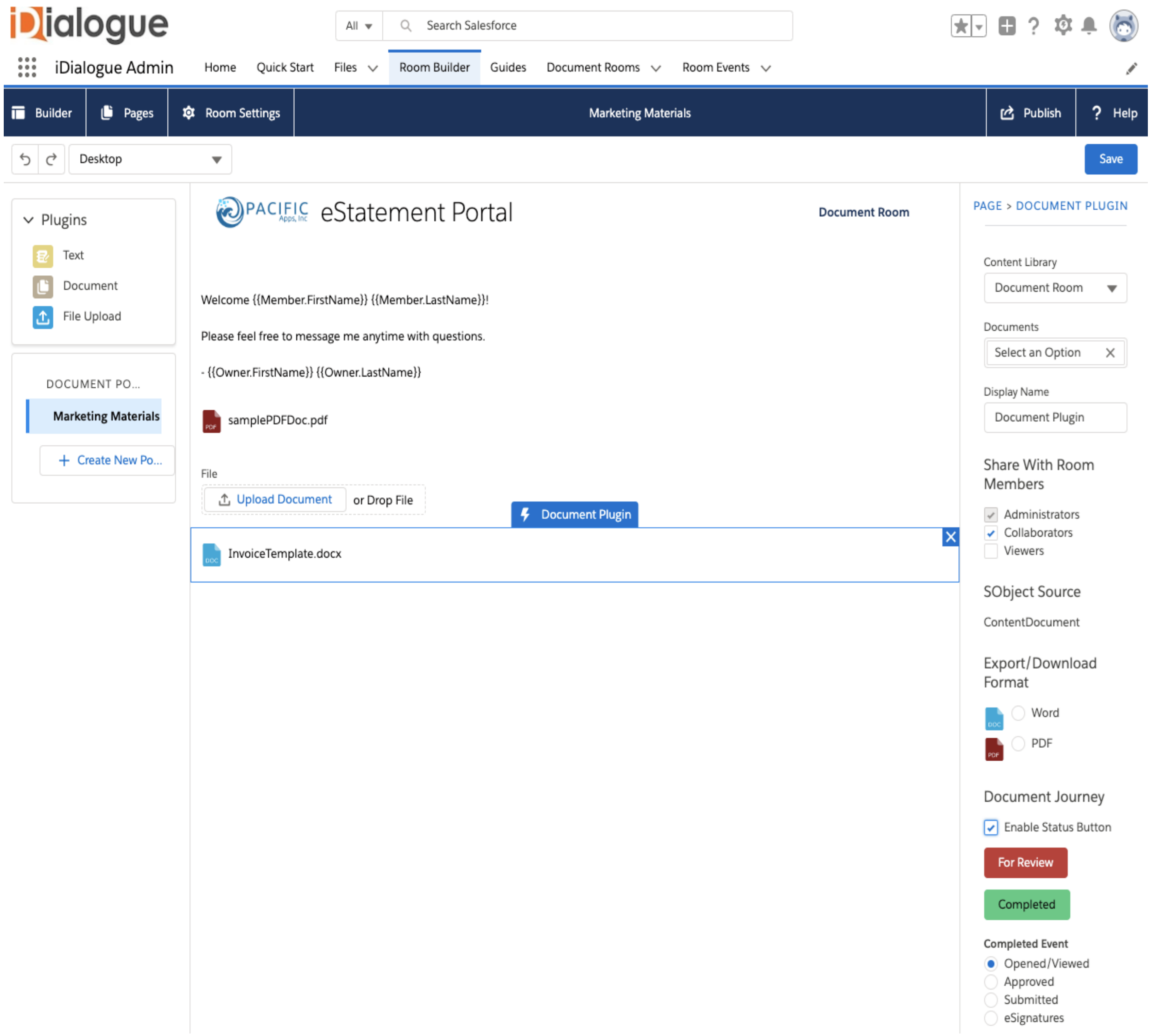
Generate code snippet

Learn more

```
{
    "StartAt": "PreMergeValidation",
    "States": {
        "PreMergeValidation": {
            "Type": "Task",
            "Resource": "arn:aws:lambda:us-east-1:123456789012:function:PreMergeValidation",
            "Next": "FetchFrontMatter"
        },
        "FetchFrontMatter": {
            "Type": "Task",
            "Resource": "arn:aws:lambda:us-east-1:123456789012:function:FetchFrontMatter",
            "Next": "MergeData"
        },
        "MergeData": {
            "Type": "Task",
            "Resource": "arn:aws:lambda:us-east-1:123456789012:function:MergeData",
            "Next": "PDFConvert"
        },
        "PDFConvert": {
            "Type": "Task",
            "Resource": "arn:aws:lambda:us-east-1:123456789012:function:PDFConvert",
            "Next": "Deliver"
        },
        "Deliver": {
            "Type": "Task",
            "Resource": "arn:aws:lambda:us-east-1:123456789012:function:Deliver",
            "End": true
        }
    }
}
```

One state machine per document processed

Case study 2: doc gen in Salesforce.com



eStatement Portal

Welcome {{Member.FirstName}} {{Member.LastName}}!

Please feel free to message me anytime with questions.

- {{Owner.FirstName}} {{Owner.LastName}}

samplePDFDoc.pdf

File

InvoiceTemplate.docx

Content Library: Document Room

Documents: Select an Option

Display Name: Document Plugin

Share With Room Members
- [✓] Administrators
- [✓] Collaborators
- [ ] Viewers

SObject Source: ContentDocument

Export/Download Format
- ○ Word
- ○ PDF

Document Journey
- [✓] Enable Status Button

Completed Event
- ● Opened/Viewed
- ○ Approved
- ○ Submitted
- ○ eSignatures

Serverless architecture scales from 0 to N documents.

Compare the previous architecture, which required up to 20 servers to execute full doc gen batches.

Lessons learnt

- Faster can sometimes be cheaper
- Web assembly:
  - Production-ready
  - Great if you have C/C++ code
  - Great fit for serverless
- Concern that cloud APIs and sensitive documents don't mix
  - Serverless makes DIY easy
- Choose serverless-ready tech
- Choose serverless-ready business models
  - Per-core/socket/CPU pricing doesn't fit
- Where is your bottleneck now?
- Be aware of cloud vendor lock-in

Thank you!

We appreciate your participation.

jason.harrop@nativedocuments.com