Infrastructure-as-PowerPoint: A No-Code Approach to Cloud Infrastructure Management

Tobias Pfandzelter
Not Affiliated with a Provider of Slide Presentation Software
pfandzelter@tu-berlin.de

Abstract

Cloud computing has made scalable infrastructure a commodity. Unfortunately, current trends towards *infrastructure-as-code* hinder the adoption of cloud resources by an audience beyond nerds. In this paper, we thus present *Infrastructure-as-PowerPoint*, leveraging tools that are already being used to share infrastructure specifications. Our approach uses machine learning because that's what everybody's doing, right?

1 Introduction

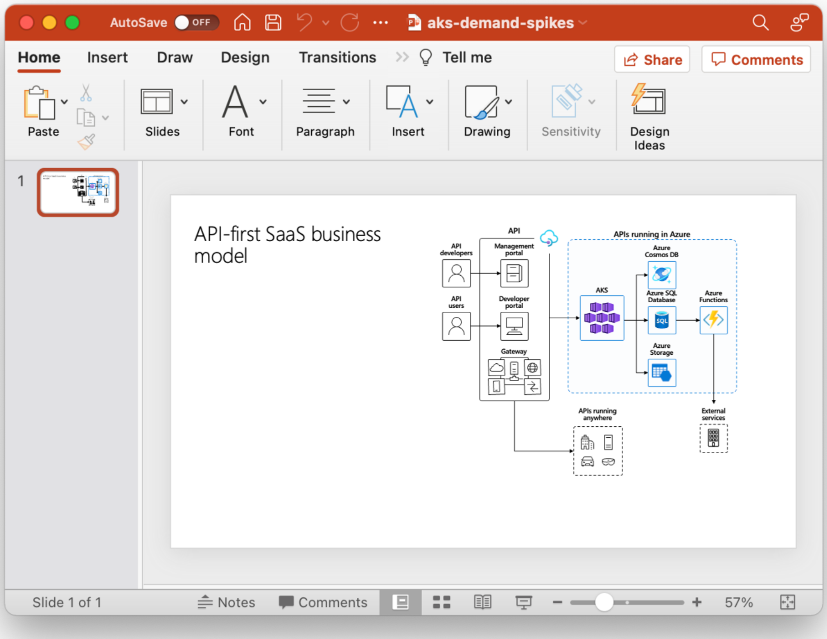
Cloud computing promises scalable, virtually infinite compute and storage resources with a pay-as-you-forget-to-turn-it-off pricing model and has developed into a considerable market. With an Everything-as-a-Service approach, it has made systems administrators all but obsolete for most companies. While some of them have joined their oppressors to become SREs, others have seen the need to support their companies as DevOps engineers[1].

Cloud computing interfaces are notoriously hard to use: A preliminary study, *i.e.*, the authors pressing `Ctrl + F`, revealed more than 230 different options to choose from on the AWS Console start page alone. The consensus has thus shifted toward using *Infrastructure-as-Code* tooling, allowing DevOps engineers to sit on their high horse and talk about things like *pipelines*, *idempotence*, and *environment drift*.

On the other hand, we can observe that such approaches have increased the barrier of entry to cloud computing for people who have never used a command line. As a result, one of today's most popular ways to share infrastructure specifications are PowerPoint slides. Figure 1 shows an example specification provided by Microsoft Azure [1]. In this paper, we present the *Infrastructure-as-PowerPoint* (EsRT[2]) paradigm that democratizes cloud computing config-

[1] To the best of our knowledge, we have yet to see a Service-as-a-Service market disruption.

[2] After considering the initialism *IaPP*, spelling it out, and laughing giddily, we have decided on using the last letter of each word in order to make it sound more serious.



API-first SaaS business model

Figure 1: An example infrastructure specification provided by Microsoft Azure as a PowerPoint slide [1].

uration by leveraging PowerPoint slides. To that end, we make the following contributions:

1. We present the general approach of EsRT, an infrastructure management approach using PowerPoint and machine learning (Section 3).
2. We forego an implementation and evaluation of our approach in order to open up the field for future work by other researchers (Section 4).
3. We discuss the limitations of our work (for obvious reasons, this is quite short) (Sections 5).

2 Background

In this section, we explain how to configure the background in PowerPoint, Cloud Computing, and Infrastructure-as-Code.

2.1 PowerPoint

In PowerPoint, set the background of your slides by entering the *Design* tab and clicking the *Format Background* button on the ribbon. You may then choose from solid fills, gradients and patterns, pictures, and even fun textures!

2.2 Cloud Computing

To the best of our knowledge, neither the Amazon Web Services [2] nor the Google Cloud Platform [3] consoles support setting a custom background. It does not even support dark mode, which is likely one of the main reasons for the development of alternative cloud management approaches. We note that third-party options, such as browser extensions and user scripts, may exist but are out-of-scope for this work.

2.3 Infrastructure-as-Code

Infrastructure-as-Code is basically just text and the background behind that text will depend on the text editor in use. The *1337* DevOps engineer will likely prefer a dark gray to black environment, as seen, *e.g.*, in the Matrix ~~trilogy~~ quadrilogy that has the main character Neo configuring an AWS Kinesis pipeline by hand.

3 The EsRT Paradigm

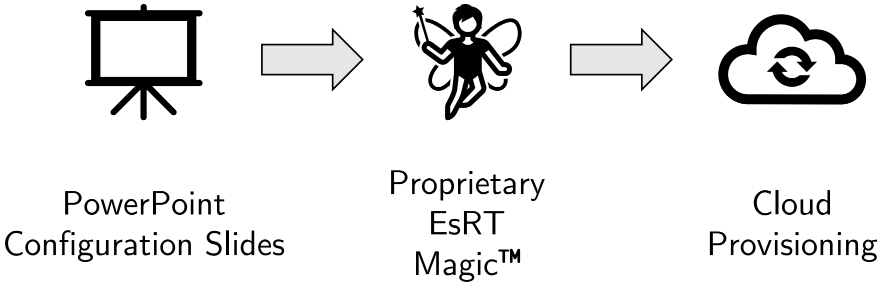


Figure 2: Proprietary magic, *i.e.*, machine learning, converts infrastructure configuration slides into cloud infrastructure.

The EsRT paradigm is illustrated in Figure 2. Users first draw their desired cloud infrastructure using icons and lines on PowerPoint slides. With the click of a button (shown in Figure 3), cloud infrastructure is provisioned and updated based on the contents of the user's slides.

At the heart of the EsRT process is proprietary magic, *i.e.*, machine learning. While a proper approach that just reads all the lines and boxes and converts

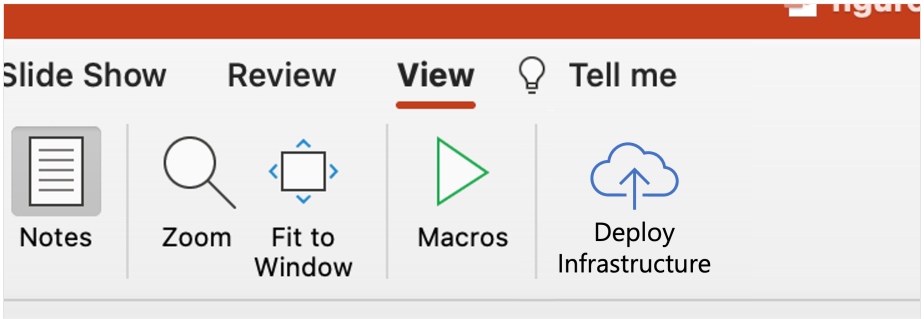


Figure 3: Mock-up EsRT user interface: Cloud infrastructure is provisioned with the click of a button – no need to use a text-based interface ever again.

them to some form of infrastructure specification is also feasible, ML is much cooler and will increase our chances of securing industry funding. We consider getting infrastructure right 9 out of 10 times to be an achievement. Unlike early 2000s *MTV Cribs* interviewees, our approach makes no assumptions on where the magic happens: Both a server-side, serverless[3] and a client-side, *e.g.*, using VBA, approach are feasible.

4 Evaluation

We leave the implementation and evaluation of the EsRT approach to future work for four main reasons: First, we have little to no experience with machine learning and don't want to embarrass ourselves on the Internet. Second, we want to enable future research in this field, *i.e.*, we need some work for students and interns. Third, we lack the necessary training data, although this may also be remedied in future student work. And fourth, we frankly don't have the time right now.

That said, we do feel like we're onto something here.

5 Discussion

In this section, we briefly ~~defend our idea against the reviewers' feedback~~ discuss the limitations of our work critically.

5.1 Open-Source Implementation

Instead of using the proprietary Microsoft Office suite and proprietary machine learning magic, some might argue that the community can benefit from open-source approaches. While that may allow even broader adoption and important

[3] This is why no one takes computer systems research seriously anymore.

community feedback, it also inhibits profitability and is thus not an option for the authors.

5.2 Vendor Lock-In

It might be argued that using a proprietary infrastructure management tool increases the lock-in effect, *e.g.*, because switching out all the AWS Lambda icons for Google Cloud Functions icons is a considerable manual task. Instead, we argue that this is stupid because we use machine learning and machine learning can do anything. What prevents us from training our model to interpret an Azure Functions icon as a Linode Cloud Firewall? That's right, we can do whatever we want! Existing Infrastructure-as-Code tooling and its focus on *repeatability* can't do anything of the sort. We thus conclude that EsRT actually greatly decreases the barriers for cloud migration and decreases vendor lock-in.

6 Related Work

Existing Infrastructure-as-Code tooling, such as *Terraform* [4], can only be used by trained engineers. Besides, although we haven't actually read it, research suggests that Infrastructure-as-Code "smells" [5,6]. To the best of the authors' knowledge, PowerPoint does not provide an olfactory interface.

Using PowerPoint as a development environment is not entirely new: In his seminal work [7], Wildenhain sine al. show that PowerPoint is Turing-complete with the construction of the PowerPoint Turing Machine (PPTXTM). Building on this discovery, both Wildenhain and others (possibly also Wildenhain in coordination or cooperation with others, or others with the support of Wildenhain, or others with blessing of both others and Wildenhain but indifference from Microsoft, or, although unlikely, the team behind a Microsoft competitor without the explicit support but with good will from Wildenhain) have presented `pptcc` [8] and `ppcc` [9], compilers for the C language that target PowerPoint. In [10], the author builds a presentation editor in a code editor, but we don't see how this information helps the reader at this point.

7 Conclusion

In this paper, we have motivated the need for a successor to Infrastructure-as-Code. We have presented the *Infrastructure-as-PowerPoint* (EsRT) paradigm, a new approach to democratize cloud infrastructure management. Our evaluation showed that we feel very good about ourselves. Further, we have opened up the field for future work in the area, namely getting this thing to actually run.

Acknowledgments

All figures in this manuscript created using Microsoft PowerPoint.

References

[1] API-first SaaS business model. Microsoft Azure. Accessible: no (because it's in PowerPoint). [Online]. Available: https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/media/aks-demand-spikes.pptx

[2] AWS Management Console. Accessed: multiple times in the past. [Online]. Available: https://console.aws.amazon.com/console/home

[3] Home – Google Cloud Platform. Accessed: yesterday. [Online]. Available: https://console.cloud.google.com/home

[4] Terraform by HashiCorp. Accessed: 2017-12-30 and 2018-6-15 and 2019-2-2 and sometime between 2020-5-3 and 2020-7-28 and also 2022-3-30 but this time just to confirm that it's still there. [Online]. Available: https://www.terraform.io/

[5] J. Schwarz, A. Steffens, and H. Lichter, "Code smells in infrastructure as code," in *Proceedings of the 11th International Conference on the Quality of Information and Communications Technology (QUATIC 2018).* IEEE, 2018, pp. 220–228.

[6] A. Rahman, C. Parnin, and L. Williams, "The seven sins: Security smells in infrastructure as code scripts," in *Proceedings of the IEEE/ACM 41st International Conference on Software Engineering (ICSE 2019).* IEEE, 2019, pp. 164–175.

[7] T. Wildenhain, "On the turing completeness of ms powerpoint," in *Proceedings of the eleventh annual intercalary robot dance party in celebration of workshop on symposium about* 2^6*th birthdays; in particular, that of harry q. bovik (SIGBOVIK 2017)*, 2017, p. somewhere.

[8] T. Wildenhain, E. Hollander, and Opliko. pptcc. GitHub. Accessed: yes. [Online]. Available: https://github.com/TomWildenhain/pptcc

[9] K. He, C. Choy, and D. Whitehead. PPSuite – what if your C code could be executed in PowerPoint? Accessed: maybe. [Online]. Available: https://devpost.com/software/ppcc

[10] P. Steinmann, "`NetPlop`: A moderately-featured presentation editor built in NetLogo," in *Proceedings of the fifteenth annual intercalary robot dance party in celebration of workshop on symposium about* 2^6*th birthdays; in particular, that of harry q. bovik (SIGBOVIK 2021)*, 2021.