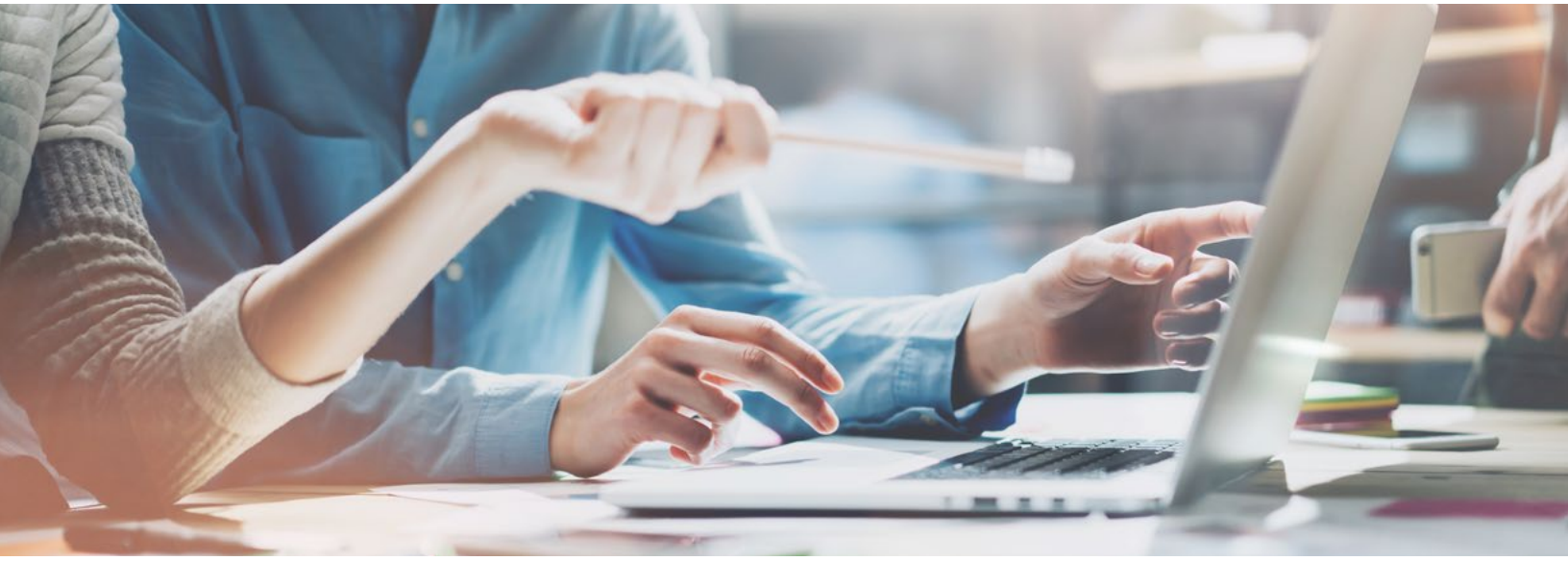
Enterprise Migration – Cloud Factory Automation
for one of the world's largest beverage corporations

CANDID PARTNERS



Candid Partners proposed an automation framework based on infrastructure as code to significantly reduce the lead time for onboarding new application engineering teams, enforce information risk policies, and improve reliability of infrastructure deployments.

BACKGROUND

One of the world's largest beverage corporations, our client, made the decision to accelerate departure from their on-premise enterprise data center and move all workloads to AWS. The movement involved a massive coordination between networking, AWS IAM, platform engineering, architecture, information risk management, business owners, IT operations and executive leadership.

THE PROBLEM

Our client had defined networking, security, capacity, billing and naming standards but had challenges enforcing them. Our client also had long lead times for onboarding new projects, quality issues implementing current projects already in-flight and were unable to enforce governance over change management procedures. As a result, delivering application infrastructure fell short of client expectations and was often taking longer than allocation of new hardware in the existing data center.

THE CANDID PARTNERS SOLUTION

The Candid Partners team proposed an automation framework based on infrastructure as code using HashiCorp Terraform language. The framework significantly reduced the lead time for onboarding new application engineering teams, enforced company information risk policies, and improved reliability of infrastructure deployments.

Actions speak louder than advice.

ACTION

The automation framework implemented a GUI front-end to onboard new projects and specify application requirements. A Lambda function was used to initiate all application meta infrastructure including; Active Directory entitlements, AWS CodeCommit repositories, Amazon S3 buckets, Amazon SNS Topic creation, CI/CD pipeline creation.

The transformation from application infrastructure requirements to Hashicorp Terraform infrastructure code was accomplished with a Lambda function fronted by AWS API Gateway using an api_key as authentication. The function generated a standardized Hashicorp Terraform directory structure into a AWS CodeCommit repository where a GoCD system automatically verified the following:

- *Hashicorp Terraform was in a standardized directory structure*
- *Infrastructure layers were used based on company standards*
- *All AWS resources specified met company standards definitions for I/O, networking, tags schema, AMIs, database and load-balancer configurations, Amazon EFS, Amazon EC2, AWS KMS pairs, etc.*

The automation system allowed the incremental promotion of infrastructure specification from DEV, QA, TEST, STAGING and PROD environments. State management between infrastructure layers was achieved by revisioning the Hashicorp Terraform state with the declarative Terraform and used locally sourced Hashicorp Terraform state.

We also developed custom Hashicorp Terraform rules that allowed for unified architectural standards to be enforced by the CI/CD system.

OUTCOME

- *The time for initial builds was reduced from 4-6 weeks to less than 2 days*
- *Application owners were able to specify infrastructure requirements without needing to code in Hashicorp Terraform*
- *Operation teams no longer needed to translate application requirements into infrastructure code*
- *Application developers are able to write their own Hashicorp Terraform code*
- *Security teams encoded standardized rules in the infrastructure CI/CD pipeline*
- *All infrastructure deployments followed company standards*
- *Non-compliant infrastructure definitions were caught before implementation*
- *Users received instant feedback on the consistency of their specifications and alignment with company standards*

RESULTS

- *Standardized intake requirements for infrastructure builds accelerated key decisions earlier in the process*
- *Infrastructure code templates provided faster processing pipelines*
- *Automated testing of infrastructure code provided more reliable infrastructure build-outs*
- *Automated notification of changes to infrastructure provided more acceptance of the DevOps practice in enterprise environments*
- *Instituting Jira workflow processes on existing email driven work-streams enabled evolution of human processes into programmatic API's*

AWS SERVICES USED

- *Amazon API Gateway*
- *AWS Lambda (Nodejs 4.3)*
- *AWS IAM*
- *AWS CodeCommit*
- *Amazon EFS*
- *Amazon EC2*
- *AWS KMS*
- *Amazon EBS*

THIRD PARTY APPLICATIONS USED

- *GoCD*
- *Atlassian JIRA™*
- *Terraform (HashiCorp)*

CANDID PARTNERS

Contact Candid Partners to find out how we can put advice into action for you.

404.815.4599 | INFO@CANDIDPARTNERS.COM

Revised: October 2, 2017 8:35 PM)