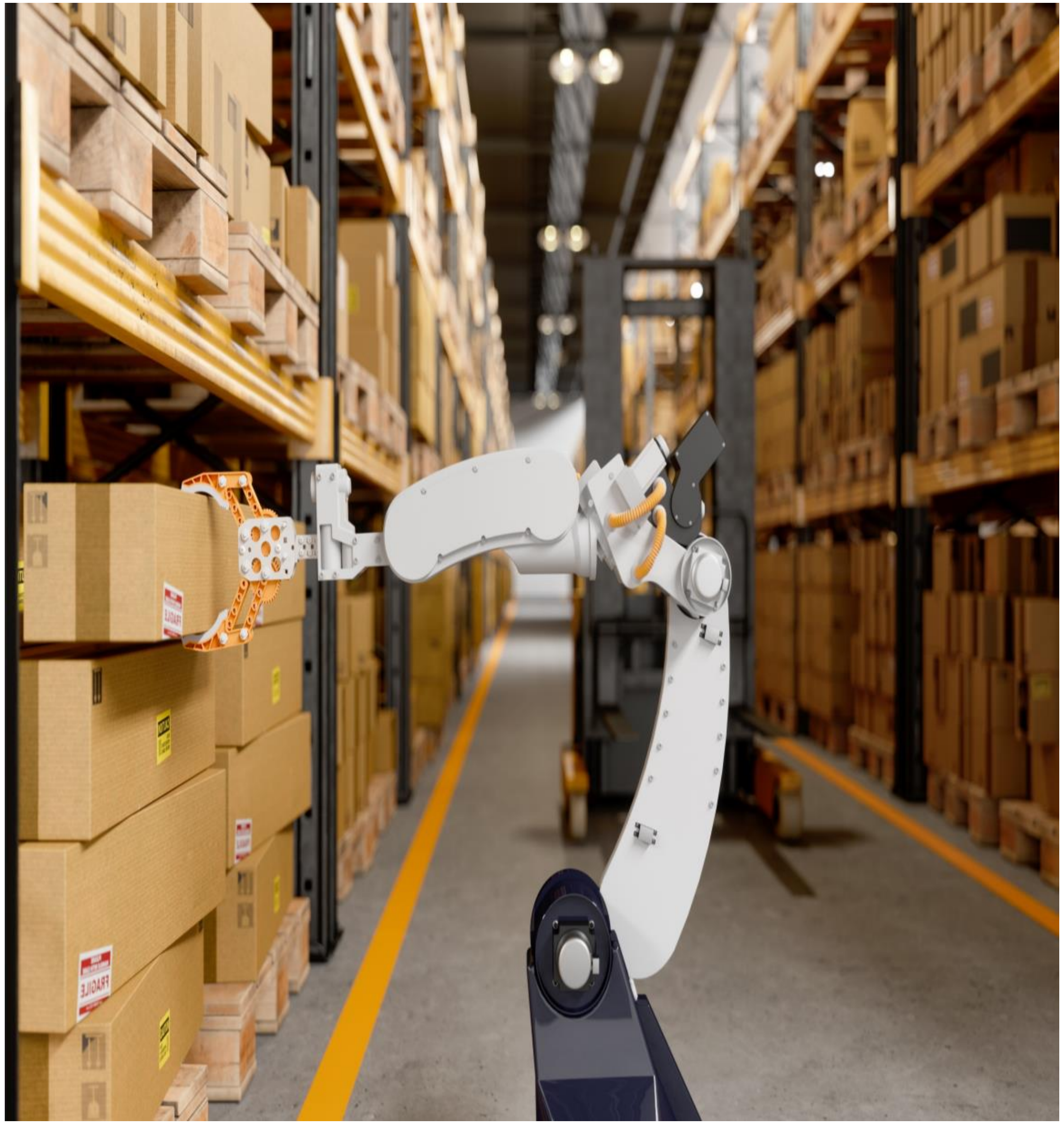
Asset - Infrastructure as Code with AWS CDK

A reusable asset packed with AWS CDK

December 2023



Agenda

01

Problem Overview and Solution

02

Advantages of IaC

03

How does Infrastructure as Code work?

04

IaC with AWS CDK

05

Contacts

01
Problem Overview and Solution

Common Challenges with Manual Deployments

Manual deployments for most of the enterprise-level come with several challenges that can impact efficiency, reliability, and consistency in the deployment process.

Human Error	Scalability Challenges	Slow-Release Cycle
Inconsistency	Difficulty in Rollbacks	Security Risks
Reduced Agility	Dependency on Specific Individuals	Limited Auditability

Common Challenges

Time ↑

Repetitive steps/Configurations/Review/Track

Cost ↑

Human effort and Time/Rollback/Redeploy

Issues ↑

Error Prone/Inconsistency/Less Traceability

Manual Vs IaC Deployments

Manual deployments and Infrastructure as Code (IaC) deployments are two different approaches to managing and deploying infrastructure in the context of IT and software deployments.

Manual Deployments

Process and Methodology

- Manual steps and procedures

Consistency and Reproducibility

- Reproducing the exact same setup

Automation

- Limited automation
- Time-consuming and error-prone

Scalability and Agility

- Struggle to scale efficiently
- Complex configurations

Version Control

- Limited scope for version control
- Hard to rollback

Auditability

- Difficulty in capturing changes manually
- Might lead to security and compliance issues

IaC Deployments

Process and Methodology

- Automated deployment of resources

Consistency and Reproducibility

- Uses same code to deploy same setup

Automation

- Emphasizes automation
- Reduces the likelihood of errors

Scalability and Agility

- Easily to accommodate changes in configuration
- enabling rapid and consistent scaling

Version Control

- leverages version control systems
- Flexible to roll back to previous versions

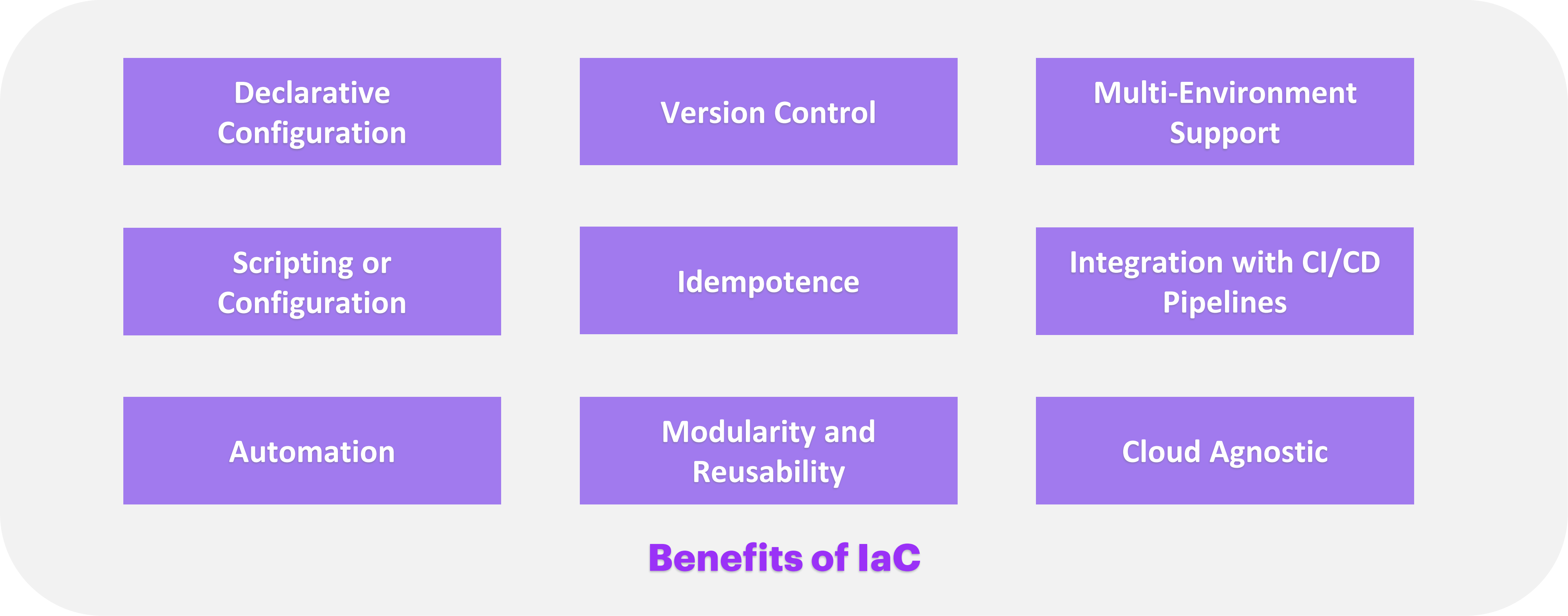
Auditability

- Changes are recorded in version control
- Enhances accountability

02
Advantages of IaC

Advantages of IaC

By adopting Infrastructure as Code, organizations can achieve greater efficiency, consistency, and agility in managing their infrastructure management.



Declarative Configuration

Version Control

Multi-Environment Support

Scripting or Configuration

Idempotence

Integration with CI/CD Pipelines

Automation

Modularity and Reusability

Cloud Agnostic

Benefits of IaC

Time ⬇

Automation/Reusability

Cost ⬇

Consistency/Flexibility to Rollback

Issues ⬇

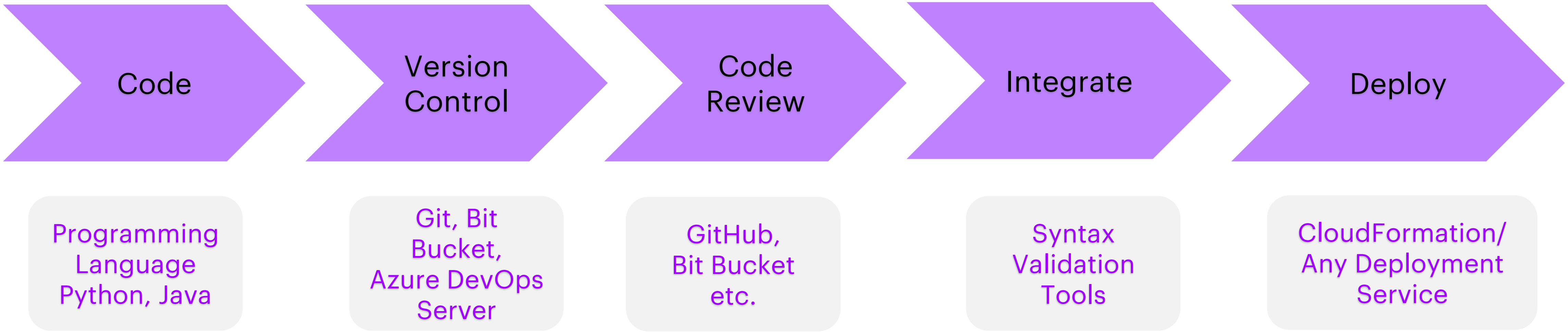
Version Control/Extensive Auditability

Infrastructure as Code (IAC)

Faster, more efficient development and accelerate software delivery

Infrastructure as Code (IAC)

Infrastructure as code (IaC) enables you to quickly setup your complete infrastructure by running a script. It is a tool for provisioning and managing resources. It is not limited to cloud works on-premise also. It is a modern approach that leverages automation and programming techniques to manage and provision infrastructure resources. With IaC, infrastructure is defined and deployed using code, which can be version-controlled, tested, and automated.



Code

Version Control

Code Review

Integrate

Deploy

Programming Language Python, Java

Git, Bit Bucket, Azure DevOps Server

GitHub, Bit Bucket etc.

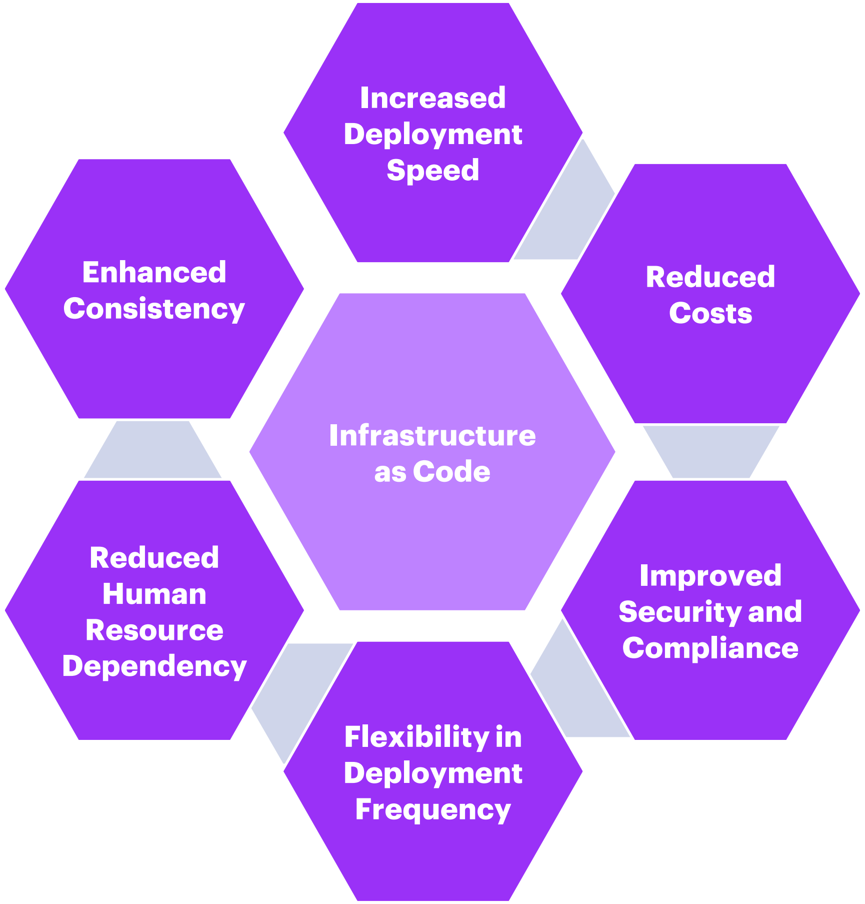
Syntax Validation Tools

CloudFormation/ Any Deployment Service

The goal of IaC is to automate and streamline the process of infrastructure deployment, configuration, and management, providing benefits in terms of efficiency, consistency, and scalability.

Benefits of IaC

Infrastructure as Code (IaC), can help organizations make informed decisions about their deployment strategies.


Increased Deployment Speed
Enhanced Consistency
Reduced Costs
Infrastructure as Code
Reduced Human Resource Dependency
Improved Security and Compliance
Flexibility in Deployment Frequency

Business Benefits

Reduction in manual deployments results in **20% decrease** in deployment related errors

50% faster provisioning of resources

20% faster time-to-market for new features and products

40% decrease in the meantime to recovery from infrastructure failures

30% improvement in collaboration efficiency through version-controlled code

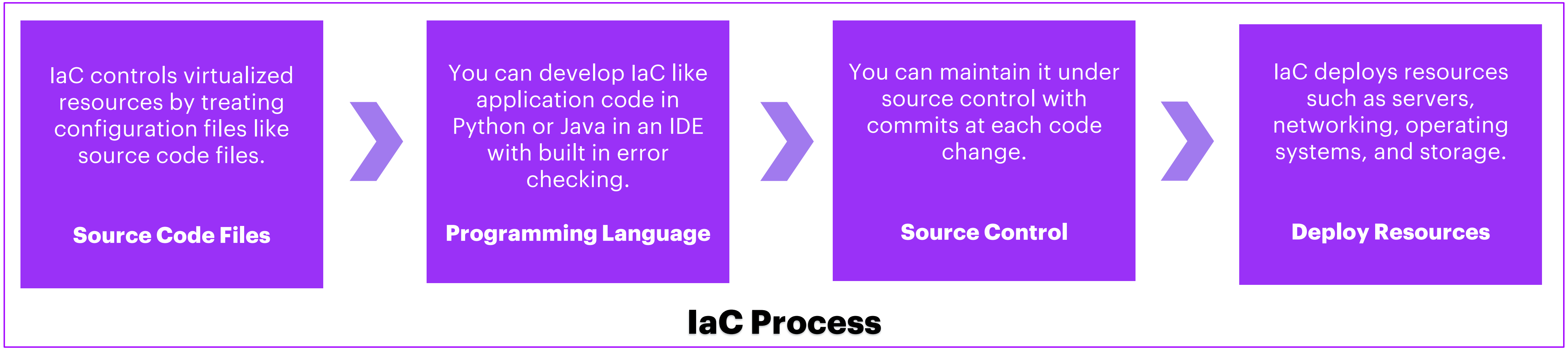
Benefits of IaC Over Time

03

How does Infrastructure as Code work?

How does Infrastructure as Code work?

IaC managing and provisioning computing infrastructure through machine-readable script files, rather than through physical hardware configuration or interactive configuration tools.



IaC controls virtualized resources by treating configuration files like source code files.

Source Code Files

You can develop IaC like application code in Python or Java in an IDE with built in error checking.

Programming Language

You can maintain it under source control with commits at each code change.

Source Control

IaC deploys resources such as servers, networking, operating systems, and storage.

Deploy Resources

IaC Process

01
Deploy to remote AWS account from local machine

- Configure the local machine with the valid AWS credentials to deploy to remote accounts.

02
Deploy to remote AWS account from Code-Pipeline

- Use AWS Code Commit as the version-controlled source code management tool.

03
Deploy to remote AWS account from any CI/CD tool

- Use a preferred source code management tool for storing the CDK application code
- Configure a preferred CI/CD tool.

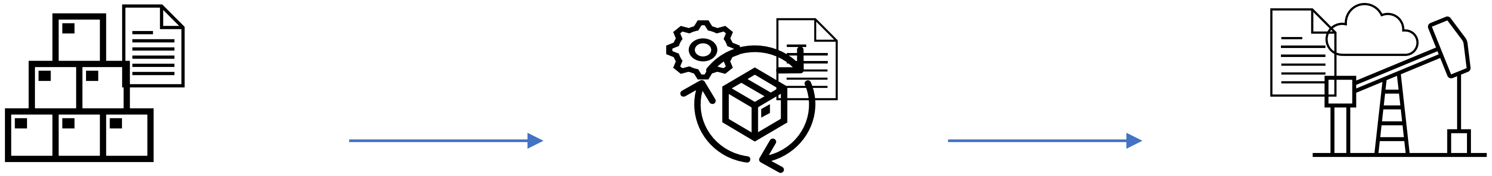
Deployment Options

04

IaC with AWS CDK

IaC with AWS CDK

AWS Cloud Development Kit (AWS CDK) accelerates cloud development using common programming languages to model your applications. When we instantiate CDK objects in Python application, those objects “compile” into a YAML template that the CDK deploys as an AWS CloudFormation stack.



Use Preconfigured application components
Download preconfigured components from a package manager or artifact repository

Model your application
Model your application logic and infrastructure in a programming language

Provision your application with AWS CloudFormation
Provision your application code and supporting infrastructure with AWS CloudFormation

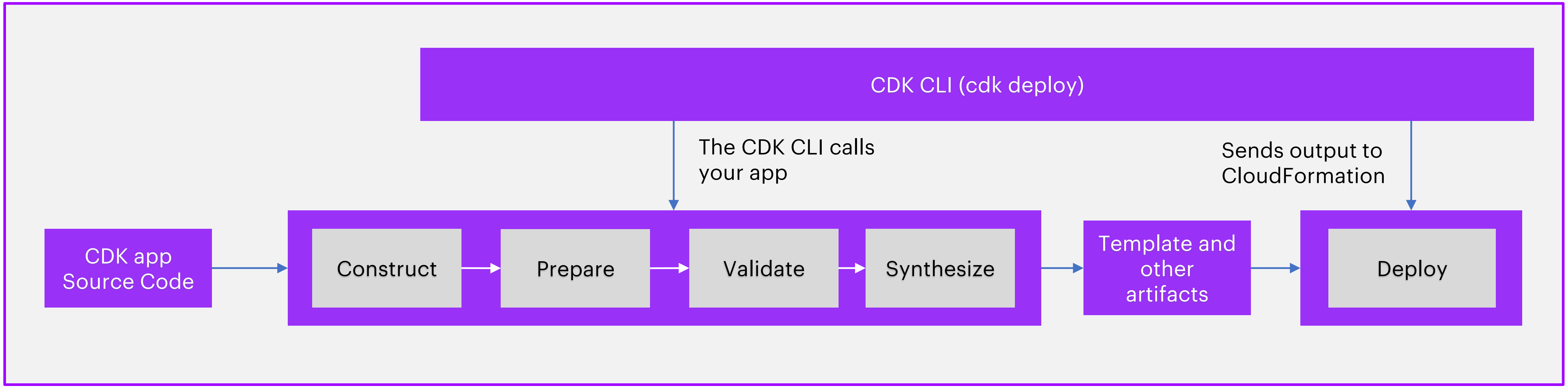
aws Cloud Development Kit

Define your application resources in your familiar programming language and accelerate development.

Simplify your AWS onboarding by using constructs that preconfigure cloud resources with proven defaults.

Design and share reusable components that meet your organization's security, compliance, and governance requirements.

AWS CDK App Life Cycle


CDK CLI (cdk deploy)
The CDK CLI calls your app
Sends output to CloudFormation
CDK app Source Code
Construct
Prepare
Validate
Synthesize
Template and other artifacts
Deploy

In this stage, all of the constructs (app, stacks, and their child constructs) are instantiated and the constructor chain is executed.

Construction

The preparation phase happens automatically. It's rare to need to use the "prepare" hook, and generally not recommended.

Preparation

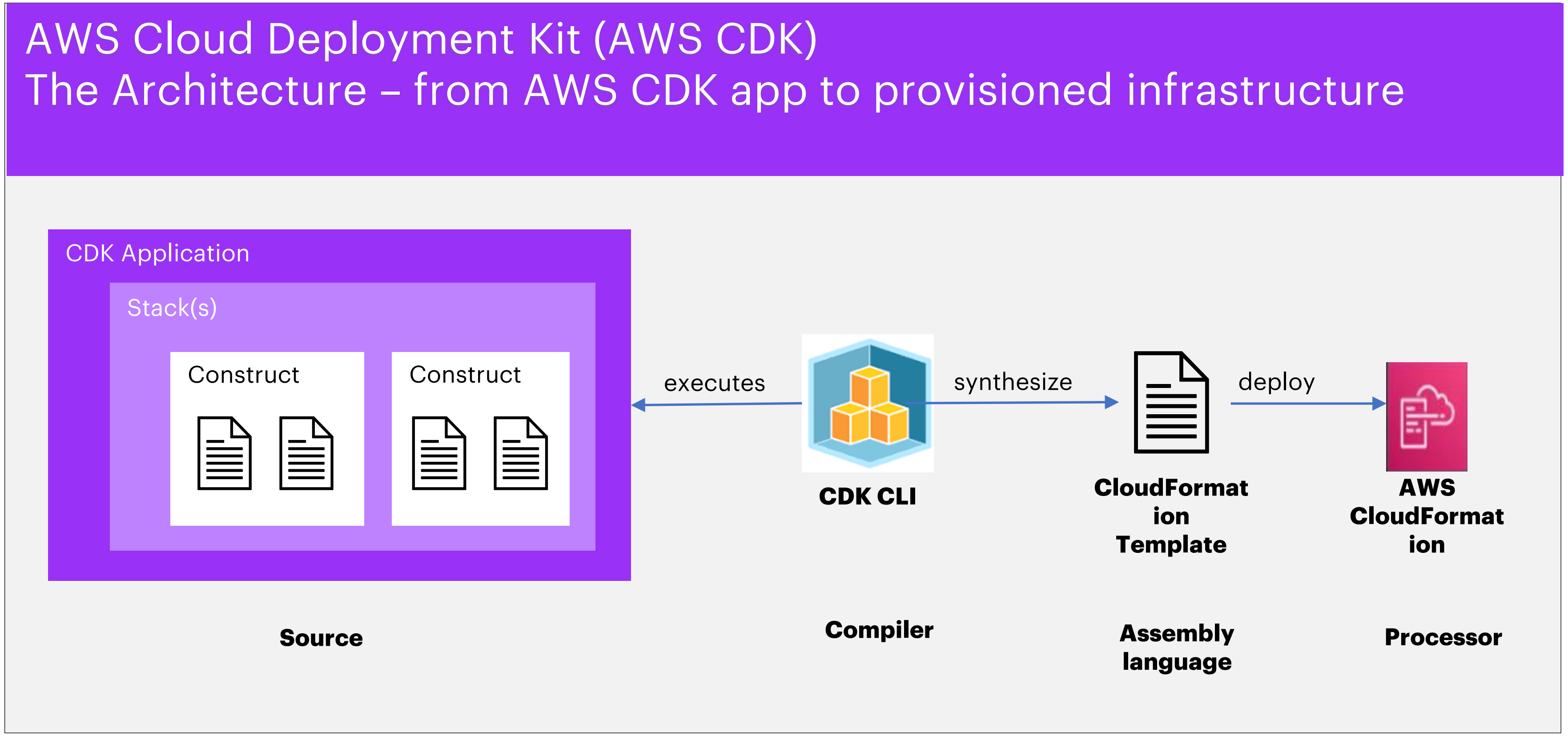
All constructs that have implemented the validate method can validate themselves to ensure that they're in a state that will correctly deploy.

Validation

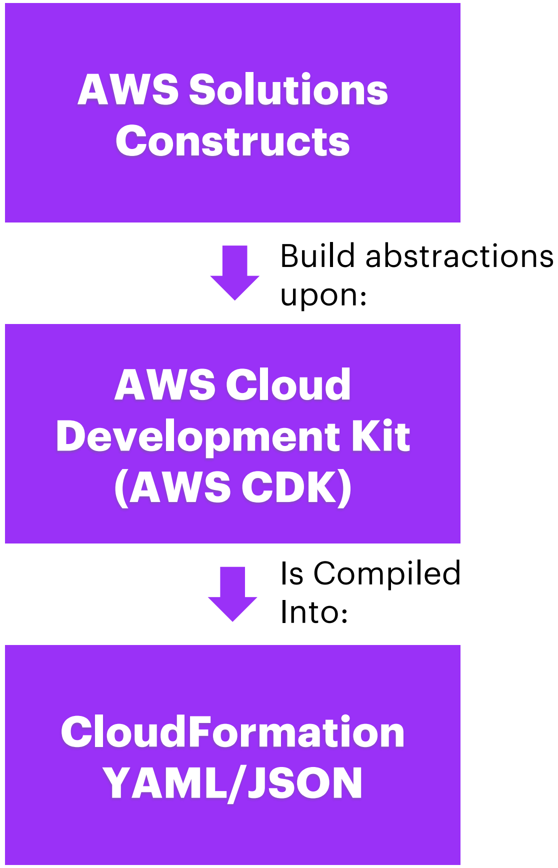
It's triggered by a call to app.synth(), and it traverses the construct tree and invokes the synthesize method on all constructs.

Synthesis

High Level Architecture


AWS Cloud Deployment Kit (AWS CDK)
The Architecture – from AWS CDK app to provisioned infrastructure

Infrastructure as Code



Key Solution Components of our Asset

As part of this asset, we have delivered code for commonly used contact center specific AWS resources along with example patterns. This code can be used for any contact center specific or independent application, projects, use cases or POC builds.

API Gateway

Connect

DynamoDB

IAM

Kinesis

KMS

Lambda

Lex

S3

VPC

Parameter Store

05

Contacts

Key Contacts

Santhosh Natarajan
Accenture Leadership

Arjun Balaraman
Management Consulting Senior Manager

Keerthi Kode
MC Delivery Manager

Venkatasantosh V
Management Consultant

Shalini Krishnamoorthy
MC Delivery Analyst

Key References

Getting Started: https://cdkworkshop.com
CDK Developer Guide: https://docs.aws.amazon.com/cdk/v2/guide/home.html
CDK Best Practices: https://docs.aws.amazon.com/cdk/v2/guide/best-practices.html

Thank you