



Premier Operations Day

Treat your infrastructure like "cattle", not pets.

Stefan Stranger
Secure Infrastructure Consultant





designed & illustrated by endjin

Stefan Stranger

Secure Infrastructure Consultant

Blog: <https://aka.ms/stranger>

Twitter: @sstranger

Github: <https://github.com/stefanstranger>



Session objectives and takeaways

Session objectives:

- Understand the importance of Infrastructure as Code
- Leverage the strong automation capabilities of Azure
- Tell the story of Cloud DevOps and Infrastructure as Code

What is Infrastructure as Code?

Wikipedia:

“Infrastructure as Code (IaC) is the process of managing and provisioning computing infrastructure and their configuration through machine-processable definition files.”



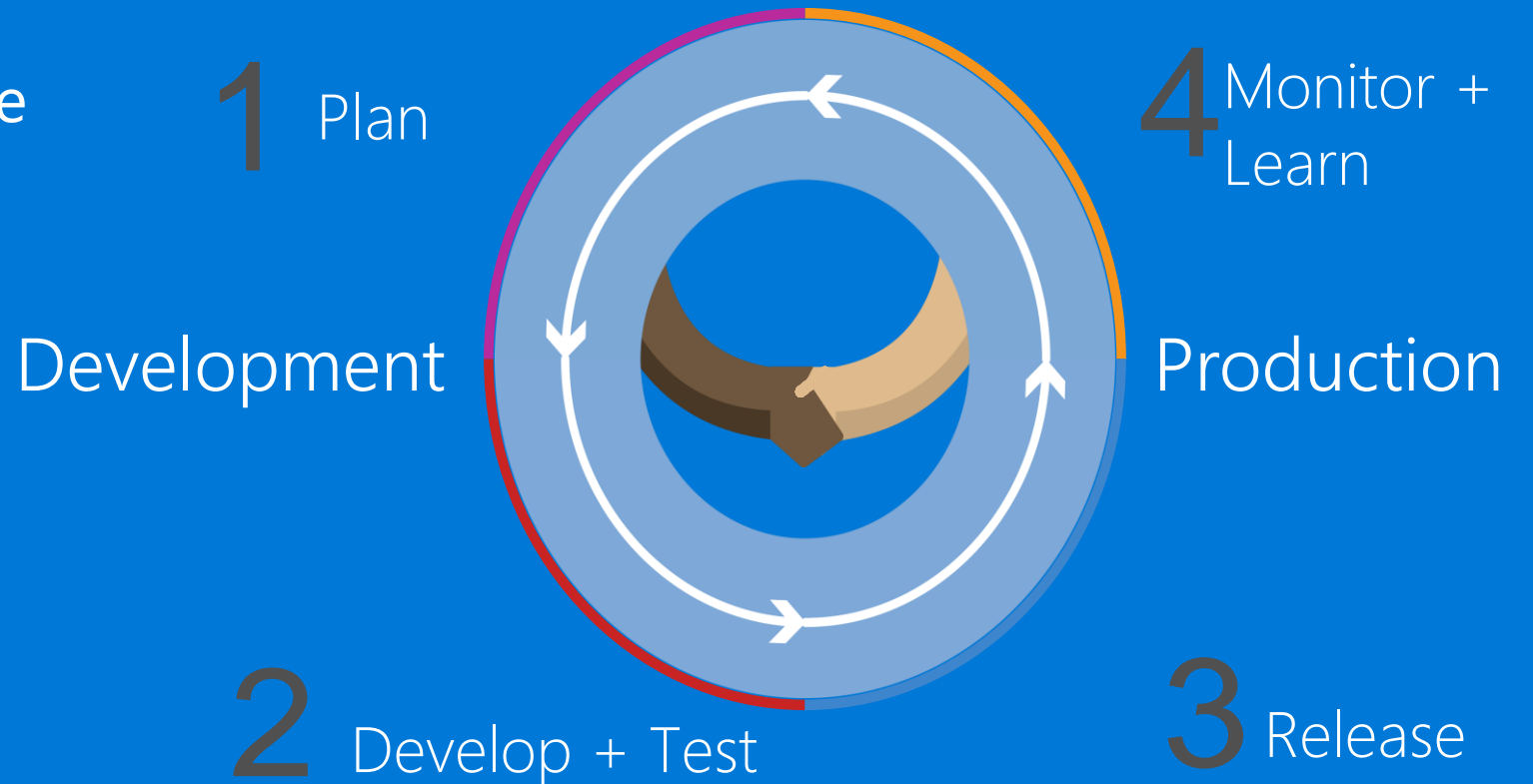
What is Infrastructure as Code?

1. Source Controlled
2. In code (Scripts & Templates)
3. Automated & Continuous Deployment
4. Testing
5. Feedback loop (Monitoring)

DevOps & Infrastructure as Code?

- Essential part of DevOps adoptions
- Supports the entire ALM process
- Gets "IT Ops" involved
- Treat infrastructure projects like software development

ALM? Application LifeCycle Management



IaC Best Practices

Treat your infrastructure like “cattle”,



.....Not “pets”

laC Best Practices



Include in planning

IaC Best Practices



Automate
Automate
Automate

This is Infrastructure as Code!!

1. Source Controlled
- 2. In code (Scripts & Templates)**
3. Automated & Continuous Deployment
4. Testing
5. Feedback loop (Monitoring)

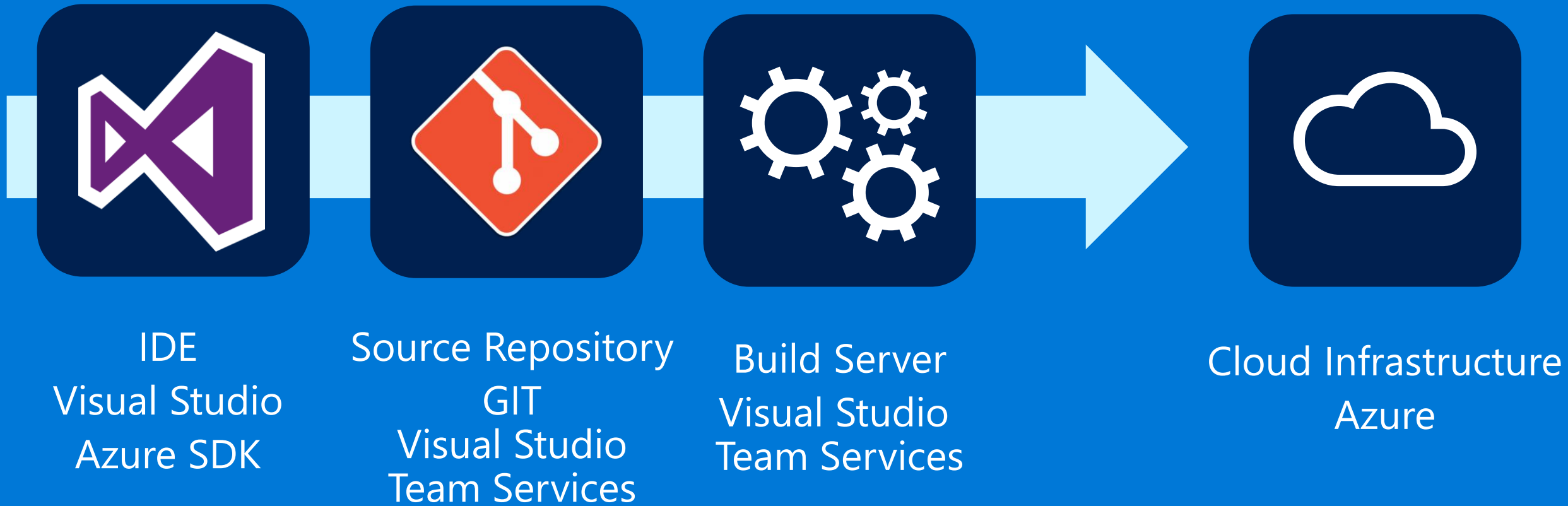
...As code Concept

- It's idempotent
- Let's infrastructure follow app lifecycle
- Supports increased release cadance
- Is essential a release pipeline
- Key part of automation

Examples:

- Azure ARM
- PowerShell
- Scripts

Demo infrastructure



Demo

....as Code

This is Infrastructure as Code!!

1. Source Controlled
- ~~2. In code (Scripts & Templates)~~
- 3. Automated & Continuous Deployment**
4. Testing
5. Feedback loop (Monitoring)

Automate Deployment Concept

- Also called Continuous Delivery
- A build system
- Performs work in the environment
- Executes scripts
- Runs tests
- Connected to source control
- Connects to other systems

Examples:

- VSTS
- TFS
- Jenkins
- Team City

Demo

Automated & Continuous Deployment

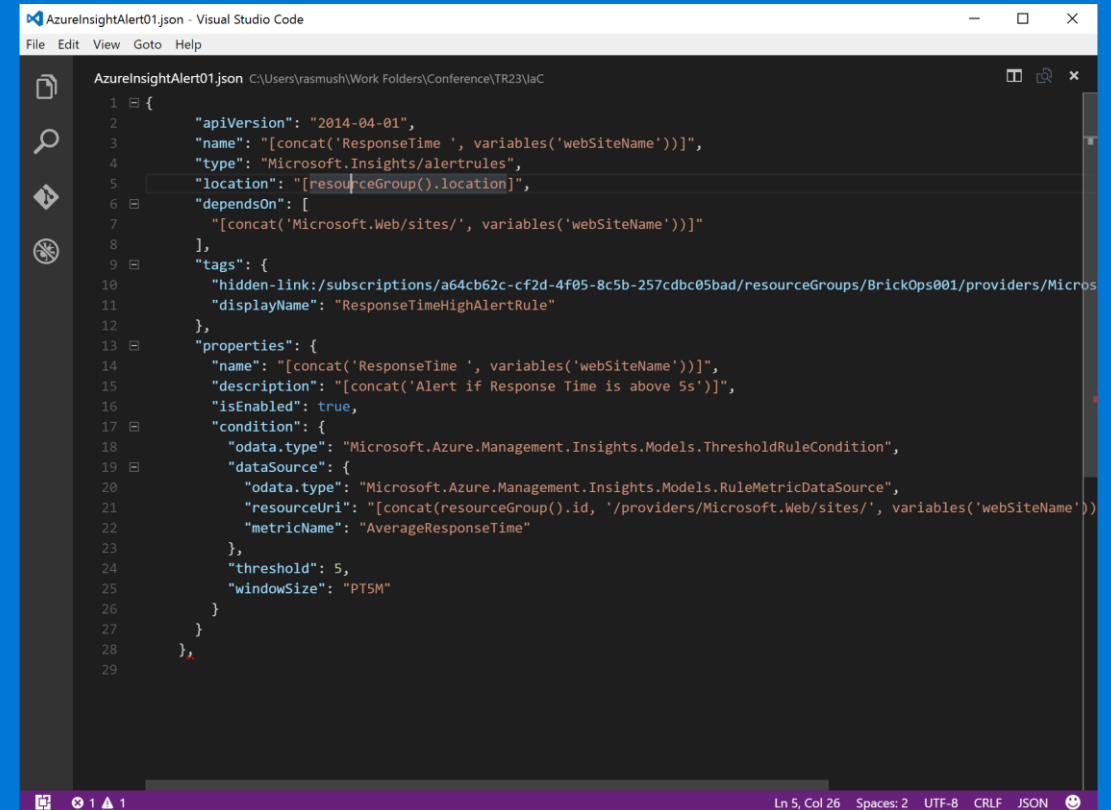
ARM Deployment modes

Increment

- Will add an update resources

Complete

- Will overwrite deployment



```
1 {
2   "apiVersion": "2014-04-01",
3   "name": "[concat('ResponseTime ', variables('webSiteName'))]",
4   "type": "Microsoft.Insights/alertrules",
5   "location": "[resourceGroup().location]",
6   "dependsOn": [
7     "[concat('Microsoft.Web/sites/', variables('webSiteName'))]"
8   ],
9   "tags": {
10    "hidden-link:/subscriptions/a64cb62c-cf2d-4f05-8c5b-257cd8c05bad/resourceGroups/BrickOps001/providers/Microsoft.Insights/alertRules/[concat('ResponseTime ', variables('webSiteName'))]": "ResponseTimeHighAlertRule"
11  },
12  "properties": {
13    "name": "[concat('ResponseTime ', variables('webSiteName'))]",
14    "description": "[concat('Alert if Response Time is above 5s')]",
15    "isEnabled": true,
16    "condition": {
17      "odata.type": "Microsoft.Azure.Management.Insights.Models.ThresholdRuleCondition",
18      "dataSource": {
19        "odata.type": "Microsoft.Azure.Management.Insights.Models.RuleMetricDataSource",
20        "resourceUri": "[concat(resourceGroup().id, '/providers/Microsoft.Web/sites/', variables('webSiteName'))]",
21        "metricName": "AverageResponseTime"
22      },
23      "threshold": 5,
24      "windowSize": "PT5M"
25    }
26  }
27 }
28
29
```

This is Infrastructure as Code!!

1. Source Controlled
- ~~2. In code (Scripts & Templates)~~
- ~~3. Automated & Continuous Deployment~~
- 4. Testing**
5. Feedback loop (Monitoring)

Testing Concept

- Process to evaluate the quality
- Usually consists of several steps.
 - Linting
 - Unit testing
 - Integration test
- Is executed by the Build system

```
Describe -Name 'Resource Group Resource' -Tags 'ARM' -Fixture {
  It -name 'Passed Resource Group existence' -test {
    Find-AzureRmResource -ResourceGroupNameContains $ResourceGroupName | Should Not Be $null
  }

  It -name 'Passed Hosting Plan existence' -test {
    Find-AzureRmResource -ResourceNameEquals $HostingPlanName | Should Not Be $null
  }

  It -name 'Passed App Service existence' -test {
    Find-AzureRmResource -ResourceNameEquals $AppServiceName | Should Not Be $null
  }
}

Describe -Name 'WebSite Availability' -Tags 'WebSite' -Fixture {
  $Response = Invoke-WebRequest -Uri $URI

  It -name 'Passed Website Response' -test {
    $Response.StatusCode | Should Be 200
  }

  It -name 'Passed Website Content' -test {
    $Response.RawContent -match $WebSiteContent | Should Be $True
  }
}
```


Demo

Testing

This is Infrastructure as Code!!

1. Source Controlled
- ~~2. In code (Scripts & Templates)~~
- ~~3. Automated & Continuous Deployment~~
- ~~4. Testing~~
- 5. Feedback loop (Monitoring)**

Feedback Concept

- Monitoring solution
- Detects faults and anomalies
- Raises alerts
- Reports usage

Examples:

- Applications Insight
- SCOM
- OMS

Demo

Feedback Loop

This is Infrastructure as Code!!

1. Source Controlled

- ~~2. In code (Scripts & Templates)~~
- ~~3. Automated & Continuous Deployment~~
- ~~4. Testing~~
- ~~5. Feedback loop (Monitoring)~~

Source Control Concept

- A version control system
- View and merge changes
- Resolve conflicts
- Change lists
- Supports multiple authors
- Connects to other systems

Examples:

- GitHub
- VSTS
- TFS
- Subversion

Demo

Source Controlled

Infrastructure as Code: Best practices

1. Version control system
2. Test your infrastructure
3. Use scripts or declarative definitions
4. All Code, Scripts and configuration follows App Lifecycle

Dank voor de
aandacht en
geluk bij het
implementeren



