

Mistakes I made

and how to avoid them

Ben Bridts



i made a lot of **mistakes**

But first...

The CDK



Use pre-configured 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

<https://aws.amazon.com/cdk/>



```
# L1 construct
bucket = s3.CfnBucket(
    self, "MyBucket",
    bucket_name="MyBucket",
    versioning_configuration=CfnBucket.VersioningConfigurationProperty(
        status="Enabled",
    ),
)

# L2 construct
bucket = s3.Bucket(
    self, "MyBucket",
    bucket_name="MyBucket",
    versioned=True,
)
```

Mistake 1: Being clever



```
class CdkStack(cdk.Stack):  
    def __init__(self, scope: cdk.Construct, construct_id: str, **kwargs):  
        super().__init__(scope, construct_id, **kwargs)  
  
        InstanceGroup(self, "MyServers", ["App1", "App2", "Web1", "Web2"])
```



```
class InstanceGroup(cdk.Construct):
    def __init__(self, scope: cdk.Construct, construct_id: str, names:
list[str]):
        super(InstanceGroup, self).__init__(scope, construct_id)

        vpc = ec2.Vpc.from_lookup(self, "vpc", is_default=True)
        number_of_azs = len(vpc.availability_zones)

        for i, name in enumerate(names):
            ec2.Instance(
                self,
                name,
                instance_type=ec2.InstanceType("t3.nano"),
                machine_image=latest_amazon_linux(),
                vpc=vpc,
                instance_name=name,
                availability_zone=vpc.availability_zones[i % number_of_azs],
            )
```

```
Resources:
  MyServersApp1EBF80F3E:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1a
  MyServersApp29C27C03F:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1b
  MyServersWeb155DBF4A3:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1c
  MyServersWeb2A94CD598:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1a
```



```
class CdkStack(cdk.Stack):  
    def __init__(self, scope: cdk.Construct, construct_id: str, **kwargs):  
        super().__init__(scope, construct_id, **kwargs)  
  
        InstanceGroup(self, "MyServers", ["App1", "App2", "Web1", "Web2"])
```



```
class CdkStack(cdk.Stack):
    def __init__(self, scope: cdk.Construct, construct_id: str, **kwargs):
        super().__init__(scope, construct_id, **kwargs)

        InstanceGroup(self, "MyServers", ["App1", "App2", "Web1", "Web2"])
```

```
class CdkStack(cdk.Stack):
    def __init__(self, scope: cdk.Construct, construct_id: str, **kwargs):
        super().__init__(scope, construct_id, **kwargs)

        InstanceGroup(
            self, "MyServers", ["App1", "App2", "Bastion", "Web1", "Web2"],
        )
```

```
Resources:
  MyServersApp1EBF80F3E:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1a
  MyServersApp29C27C03F:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1b
  MyServersWeb155DBF4A3:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1c
  MyServersWeb2A94CD598:
    Type: AWS::EC2::Instance
    Properties:
      AvailabilityZone: eu-west-1a
```

Resources:

MyServersApp1EBF80F3E:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1a

MyServersApp29C27C03F:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1b

MyServersWeb155DBF4A3:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1c

MyServersWeb2A94CD598:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1a

Resources:

MyServersApp1EBF80F3E:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1a

MyServersApp29C27C03F:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1b

MyServersBastion70E0BC3A:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1c

MyServersWeb155DBF4A3:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1a

MyServersWeb2A94CD598:
Type: AWS::EC2::Instance
Properties:
AvailabilityZone: eu-west-1b

Resources:

MyServersApp1EBF80F3E:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1a

MyServersApp29C27C03F:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1b

MyServersWeb155DBF4A3:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1c

MyServersWeb2A94CD598:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1a

Resources:

MyServersApp1EBF80F3E:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1a

MyServersApp29C27C03F:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1b

MyServersBastion70E0BC3A:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1c

MyServersWeb155DBF4A3:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1a

MyServersWeb2A94CD598:

Type: AWS::EC2::Instance

Properties:

AvailabilityZone: eu-west-1b

Chapter 1: Lessons learned

don't be clever

have a way to build
mechanical sympathy

Mistake 2: Custom Resources

```

class SsmStringParameter(cdk.Construct):
    def __init__(self, scope: cdk.Construct, construct_id: str,
                  name: str, value: str):
        super(SsmStringParameter, self).__init__(scope, construct_id)

        create_parameter = AwsSdkCall(
            service="SSM", action="putParameter",
            parameters={"Name": name, "Value": value, "Type": "String"},
            physical_resource_id=PhysicalResourceId.of(name),
        )
        update_parameter = AwsSdkCall(
            service="SSM", action="putParameter",
            parameters={
                "Name": name, "Value": value, "Type": "String",
                "Overwrite": True,
            },
            physical_resource_id=PhysicalResourceId.of(name),
        )
        delete_parameter = AwsSdkCall(
            service="SSM", action="deleteParameter",
            parameters={"Name": name},
            physical_resource_id=PhysicalResourceId.of(name),
        )

```

```

    AwsCustomResource(
        self, "Resource",
        on_create=create_parameter,
        on_update=update_parameter,
        on_delete=delete_parameter,
        policy=AwsCustomResourcePolicy.from_sdk_calls(
            resources=AwsCustomResourcePolicy.ANY_RESOURCE
        ),
        log_retention=logs.RetentionDays.ONE_MONTH,
    )

```

```

class CdkStack(cdk.Stack):
    def __init__(self, scope: cdk.Construct, construct_id: str, **kwargs) ->
    None:
        super().__init__(scope, construct_id, **kwargs)

        SsmStringParameter(self, "Parameter", name="test", value="something")

```

```

    AwsCustomResource(
        self, "Resource",
        on_create=create_parameter,
        on_update=update_parameter,
        on_delete=delete_parameter,
        policy=AwsCustomResourcePolicy.from_sdk_calls(
            resources=AwsCustomResourcePolicy.ANY_RESOURCE
        ),
        log_retention=logs.RetentionDays.ONE_MONTH,
    )

```


Resources:

ParameterD35CEA90:

Type: Custom::AWS

Properties:

ServiceToken:

Fn::GetAtt:

- AWS679f53fac002430cb0da5b7982bd22872D164C4C
- Arn

Create: '{"action":"putParameter","service":"SSM", [...] }'

Update: '{"action":"putParameter","service":"SSM", [...] }'

Delete: '{"action":"deleteParameter","service":"SSM", [...] }'

InstallLatestAwsSdk: true

AWS679f53fac002430cb0da5b7982bd22872D164C4C:

Type: AWS::Lambda::Function

AWS679f53fac002430cb0da5b7982bd2287LogRetentionCE72797A:

Type: Custom::LogRetention

LogRetentionaae0aa3c5b4d4f87b02d85b201efdd8aFD4BFC8A:

Type: AWS::Lambda::Function

Log groups

By default, we only load up to 10000 log groups.



Actions ▼

View in Logs Insights

Create log group



Filter log groups or try prefix search



Exact
match



1



<input type="checkbox"/>	Log group ▲	Retention ▼	Metric filters ▼
<input type="checkbox"/>	/aws/lambda/CdkStack-AWS679f53fac002430cb0da5b7982bd22872D164C-173SQB5ORM...	1 month	-
<input type="checkbox"/>	/aws/lambda/CdkStack-AWS679f53fac002430cb0da5b7982bd22872D164C-1JAOXNVQ14JG2	1 month	-
<input type="checkbox"/>	/aws/lambda/CdkStack-AWS679f53fac002430cb0da5b7982bd22872D164C-1P5C7PALWQP...	1 month	-
<input type="checkbox"/>	/aws/lambda/CdkStack-AWS679f53fac002430cb0da5b7982bd22872D164C-BLZ7RCLU1H4L	1 month	-
<input type="checkbox"/>	/aws/lambda/CdkStack-AWS679f53fac002430cb0da5b7982bd22872D164C-CP127GN6PQ9X	1 month	-
<input type="checkbox"/>	/aws/lambda/CdkStack-AWS679f53fac002430cb0da5b7982bd22872D164C-GQSZPE12BW4B	1 month	-
<input type="checkbox"/>	/aws/lambda/CdkStack-AWS679f53fac002430cb0da5b7982bd22872D164C-HGV90KVKXRHB	1 month	-
<input type="checkbox"/>	/aws/lambda/CdkStack-LogRetentionaae0aa3c5b4d4f87b02d85b201efd-1R290BV0CGG8L	1 day	-
<input type="checkbox"/>	/aws/lambda/CdkStack-LogRetentionaae0aa3c5b4d4f87b02d85b201efd-4HISU6N1NFIA	1 day	-

```

class SsmStringParameter(cdk.Construct):
    def __init__(self, scope: cdk.Construct, construct_id: str,
                  name: str, value: str):
        super(SsmStringParameter, self).__init__(scope, construct_id)

        create_parameter = AwsSdkCall(
            service="SSM", action="putParameter",
            parameters={"Name": name, "Value": value, "Type": "String"},
            physical_resource_id=PhysicalResourceId.of(name),
        )
        update_parameter = AwsSdkCall(
            service="SSM", action="putParameter",
            parameters={
                "Name": name, "Value": value, "Type": "String",
                "Overwrite": True,
            },
            physical_resource_id=PhysicalResourceId.of(name),
        )
        delete_parameter = AwsSdkCall(
            service="SSM", action="deleteParameter",
            parameters={"Name": name},
            physical_resource_id=PhysicalResourceId.of(name),
        )

        AwsCustomResource(
            self, "Resource",
            on_create=create_parameter,
            on_update=update_parameter,
            on_delete=delete_parameter,
            policy=AwsCustomResourcePolicy.from_sdk_calls(
                resources=AwsCustomResourcePolicy.ANY_RESOURCE
            ),
            log_retention=logs.RetentionDays.ONE_MONTH,
        )

```

```

class CdkStack(cdk.Stack):
    def __init__(self, scope: cdk.Construct, construct_id: str, **kwargs) ->
    None:
        super().__init__(scope, construct_id, **kwargs)

        SsmStringParameter(self, "Parameter", name="test", value="something")

```

```

AwsCustomResource(
    self, "Resource",
    on_create=create_parameter,
    on_update=update_parameter,
    on_delete=delete_parameter,
    policy=AwsCustomResourcePolicy.from_sdk_calls(
        resources=AwsCustomResourcePolicy.ANY_RESOURCE
    ),
    log_retention=logs.RetentionDays.ONE_MONTH,
)

```



```
delete_parameter = AwsSdkCall(  
    service="SSM", action="deleteParameter",  
    parameters={"Nome": name},  
    physical_resource_id=PhysicalResourceId.of(name),  
)
```

```
parameters={"Nome": name},
```

CdkStack

[Delete](#)[Update](#)[Stack actions ▼](#)[Create stack ▼](#)[Stack info](#)[Events](#)[Resources](#)[Outputs](#)[Parameters](#)[Template](#)[Change sets](#)

Resources (5)



Logical ID ▲	Type ▼	Status ▼	Status reason ▼
AWS679f53fac002430cb0da5b7982bd22872D164C4C	AWS::Lambda::Function	✔ DELETE_COMPLETE	-
AWS679f53fac002430cb0da5b7982bd2287ServiceRoleC1EA0FF2	AWS::IAM::Role	✔ DELETE_COMPLETE	-
CDKMetadata	AWS::CDK::Metadata	✔ DELETE_COMPLETE	-
ParameterCustomResourcePolicy2C642806	AWS::IAM::Policy	✔ DELETE_COMPLETE	-
ParameterD35CEA90	Custom::AWS	✘ DELETE_FAILED	Received response status [FAILED] from custom resource. Message returned: There were 2 validation errors: * MissingRequiredParameter: Missing required key 'Name' in params * UnexpectedParameter: Unexpected key 'Nome' found in params (RequestId: 568cea5b-da7f-4925-94de- 03c9fd588cb0)

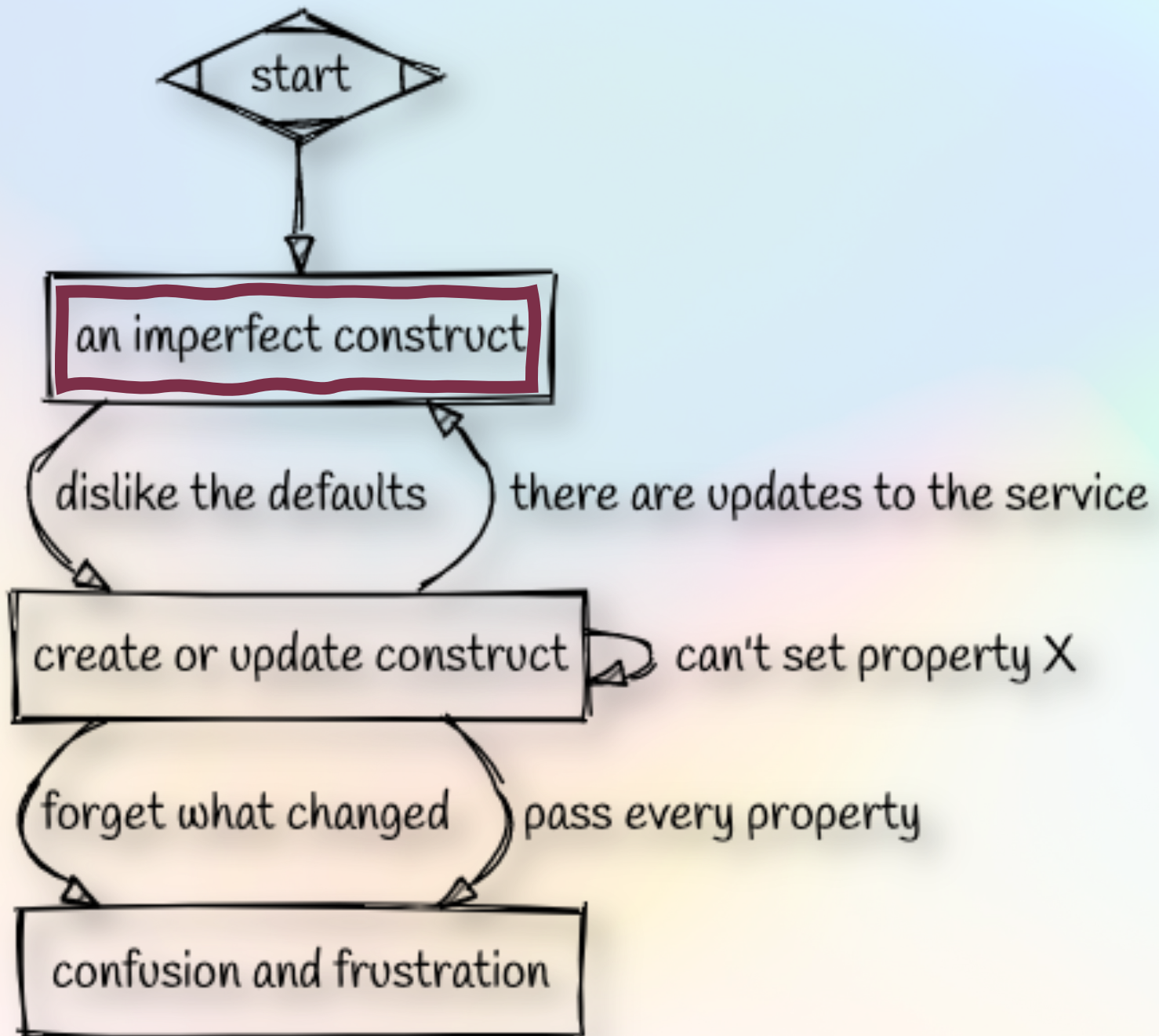
Chapter 2: Lessons Learned

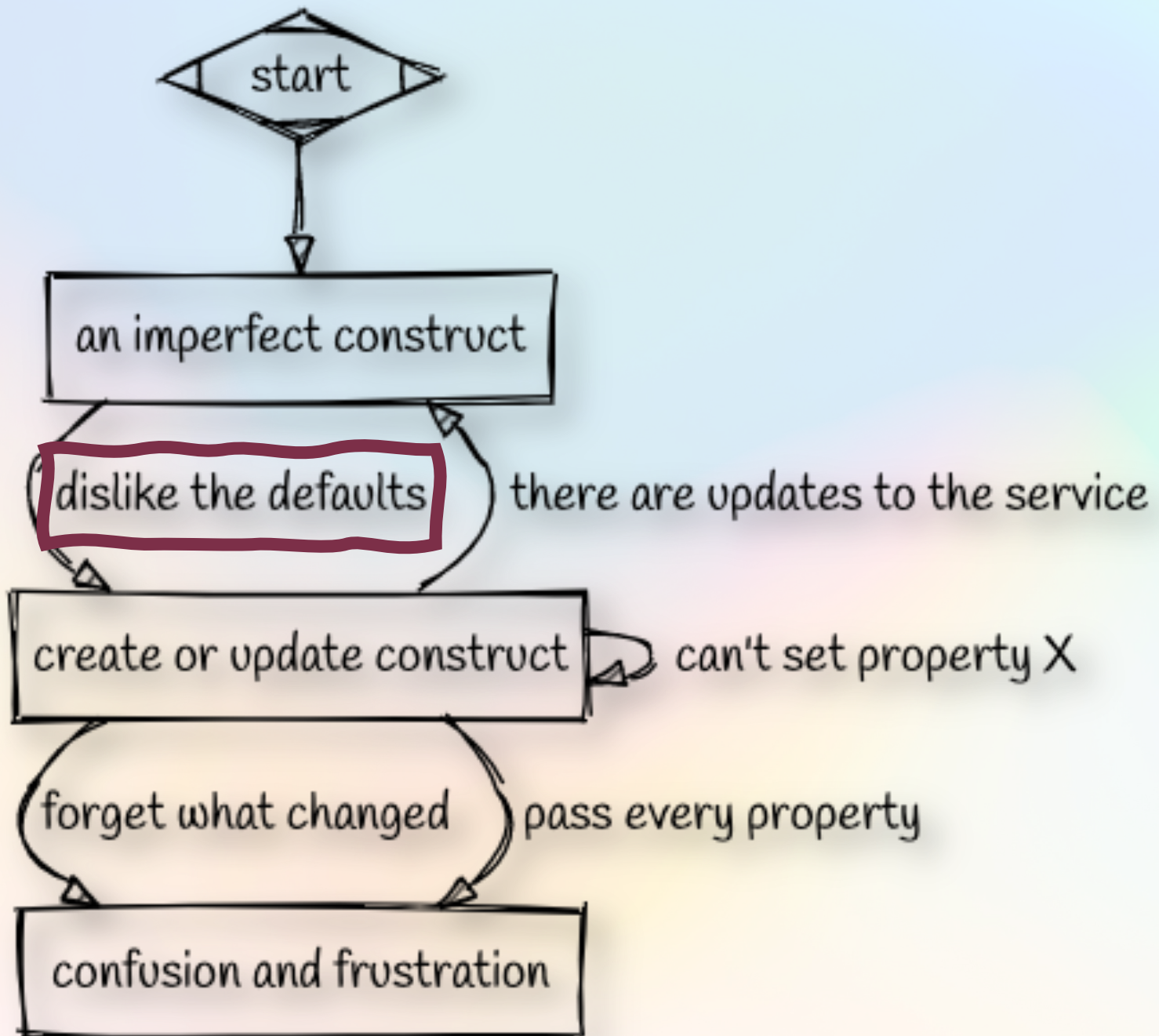
**custom resources are
applications**

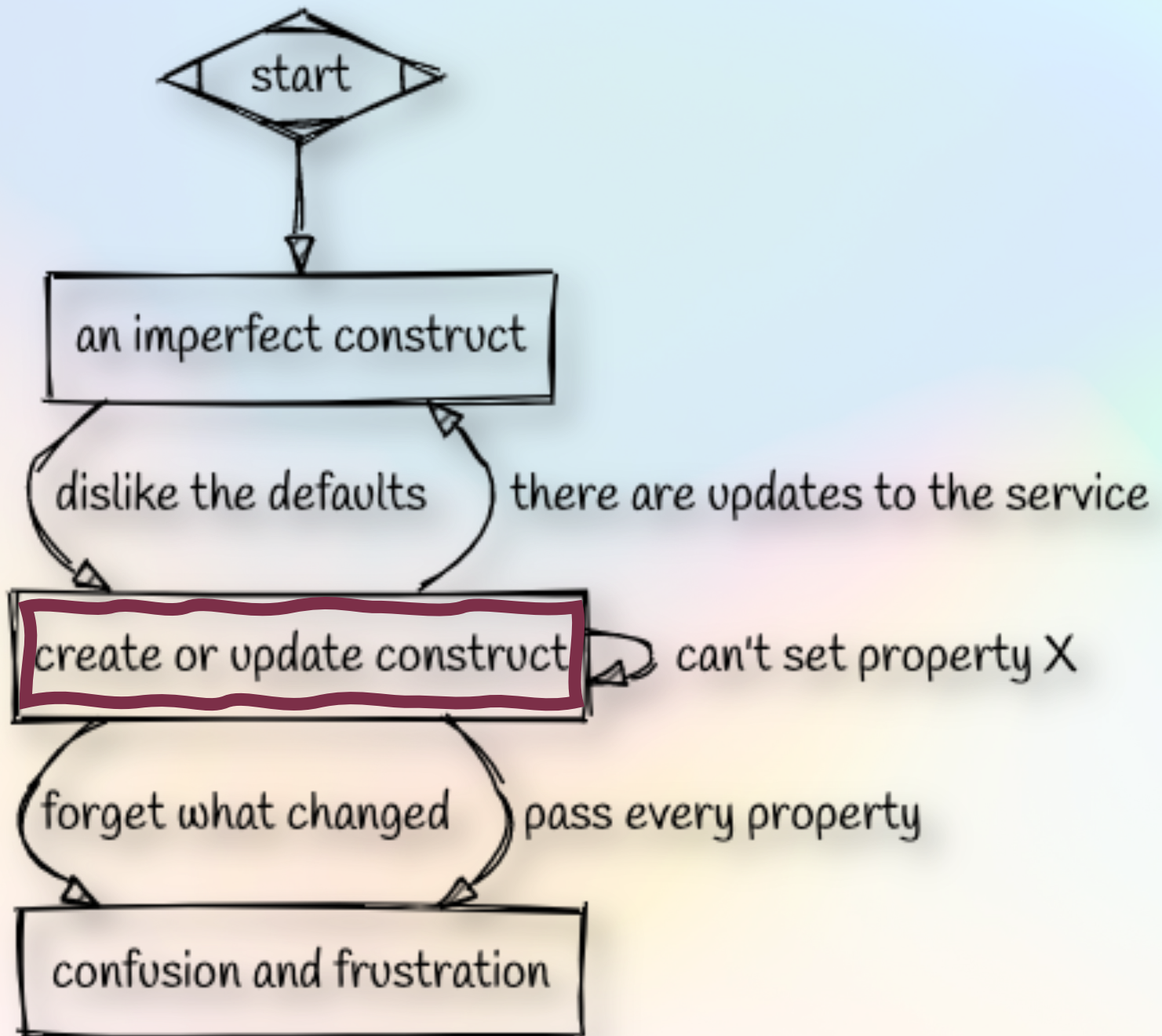
use **CloudFormation**
features

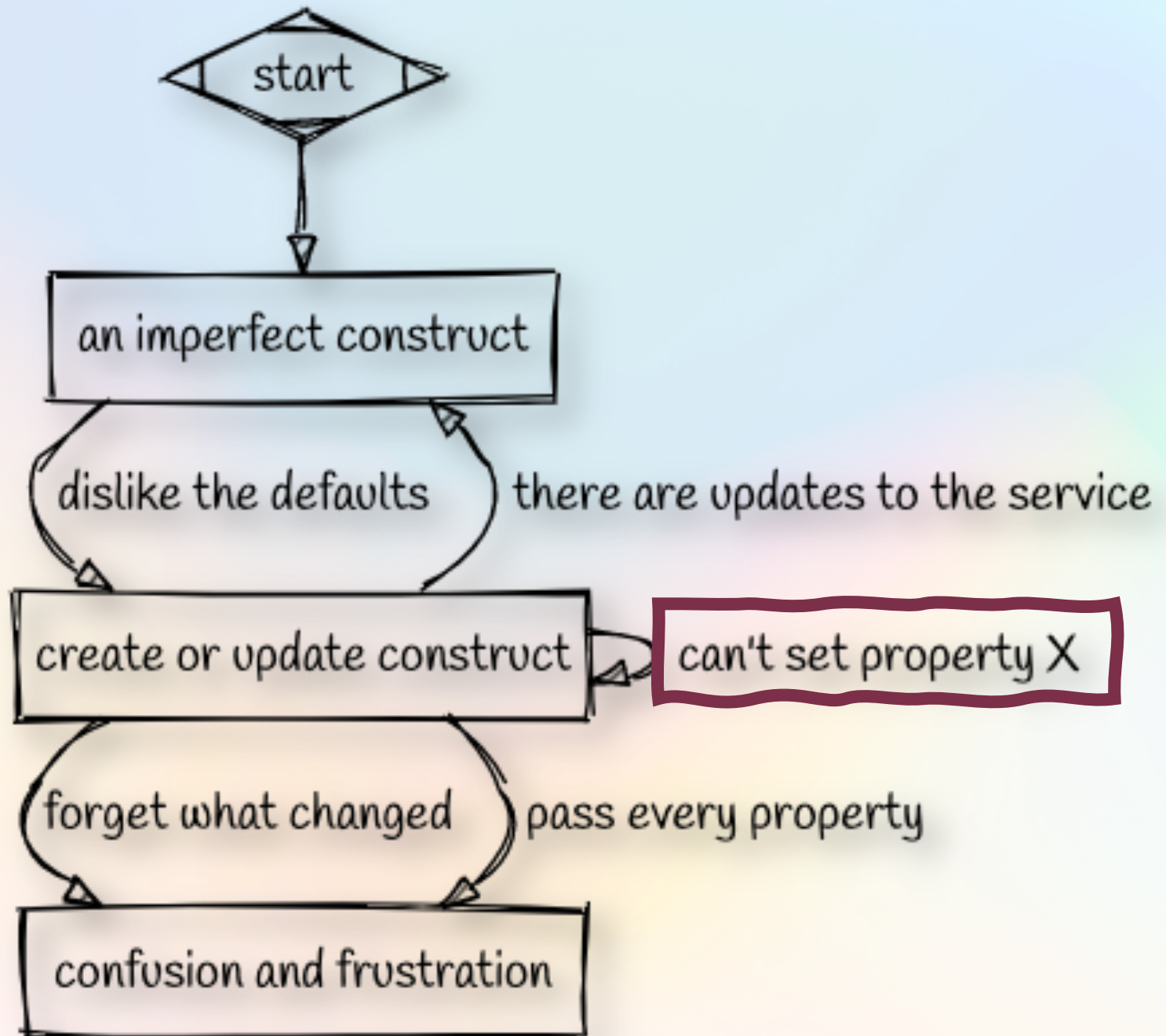
**resource providers
are the future**

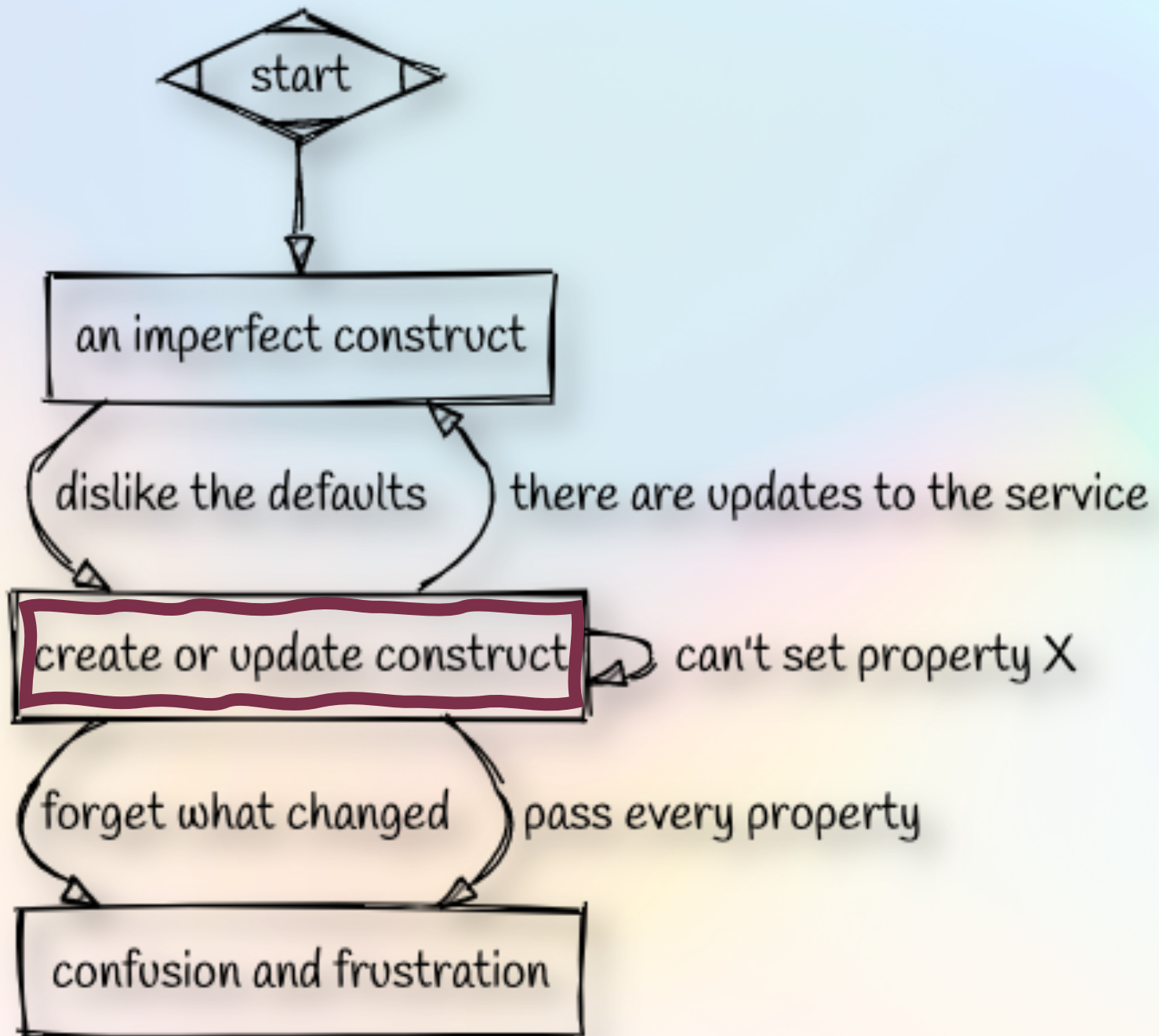
Mistake 3: Abstractions

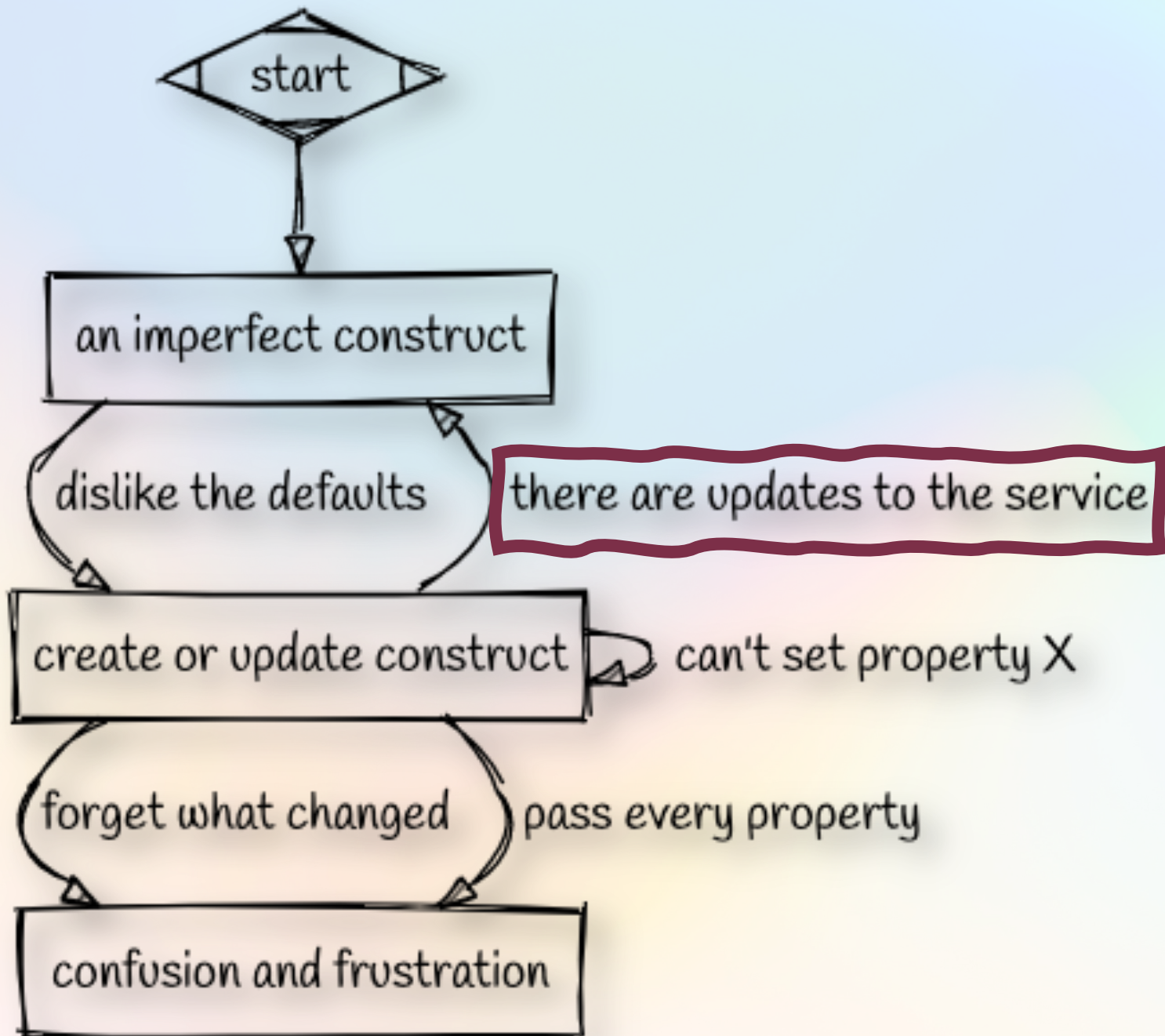


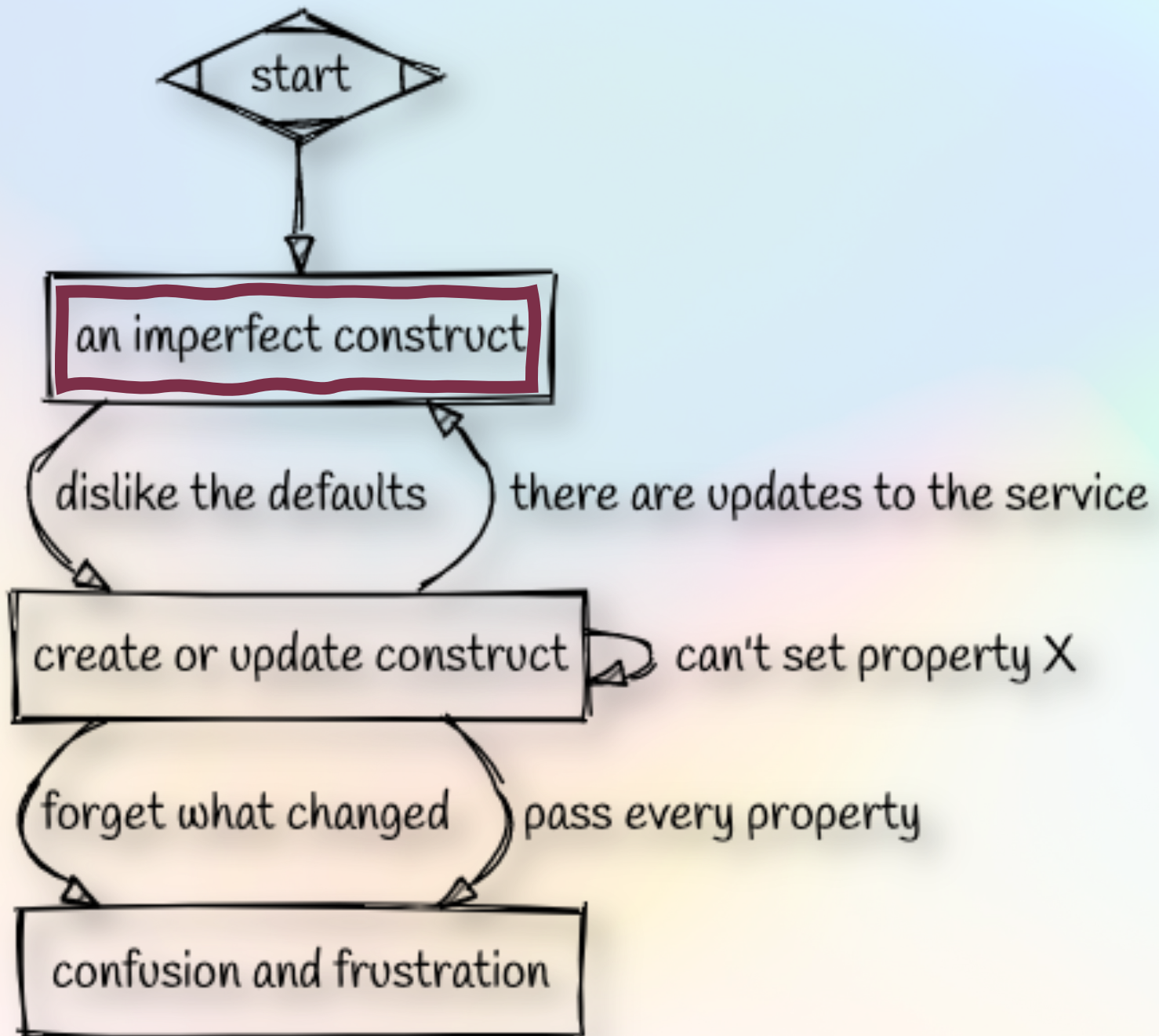


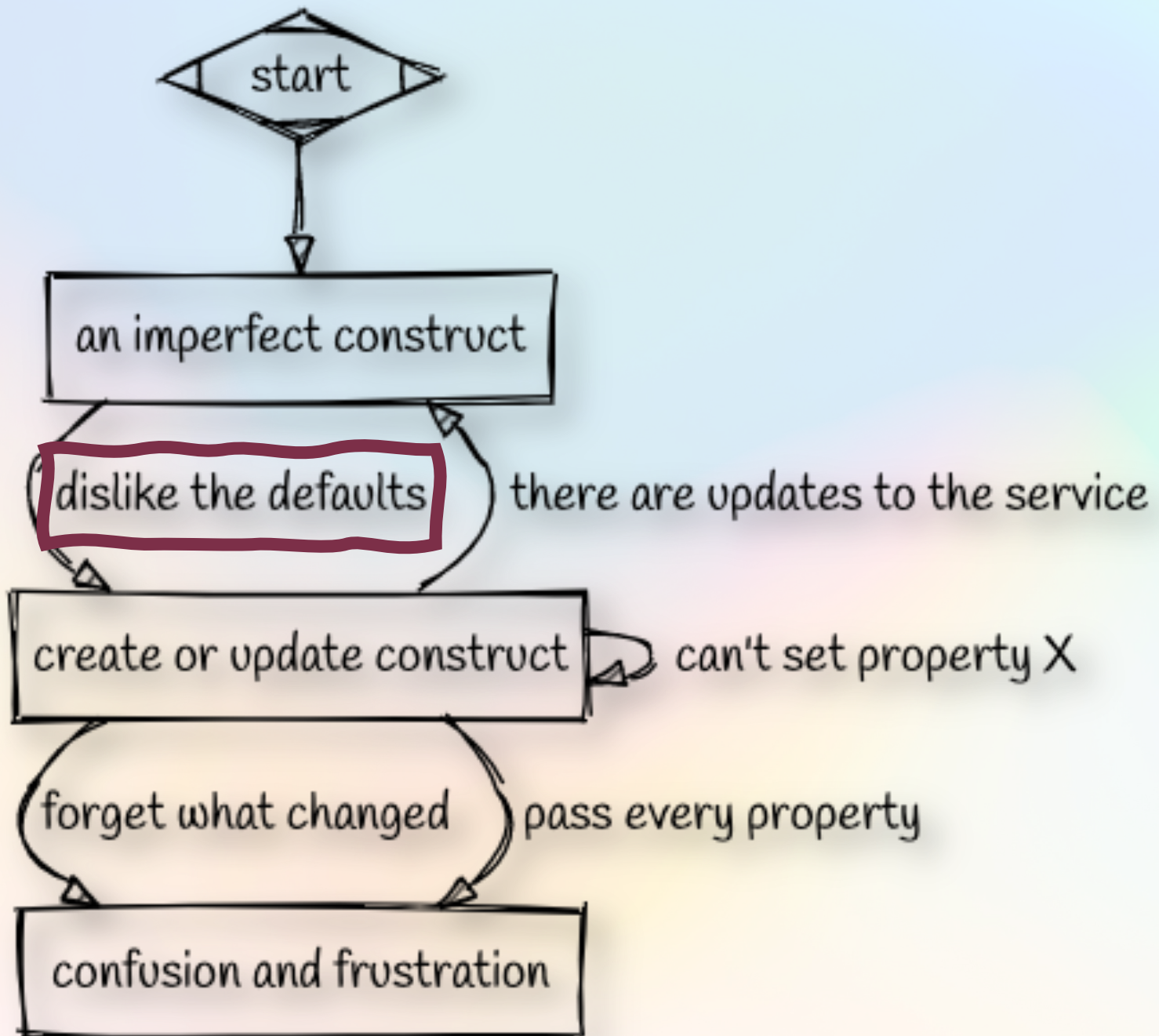


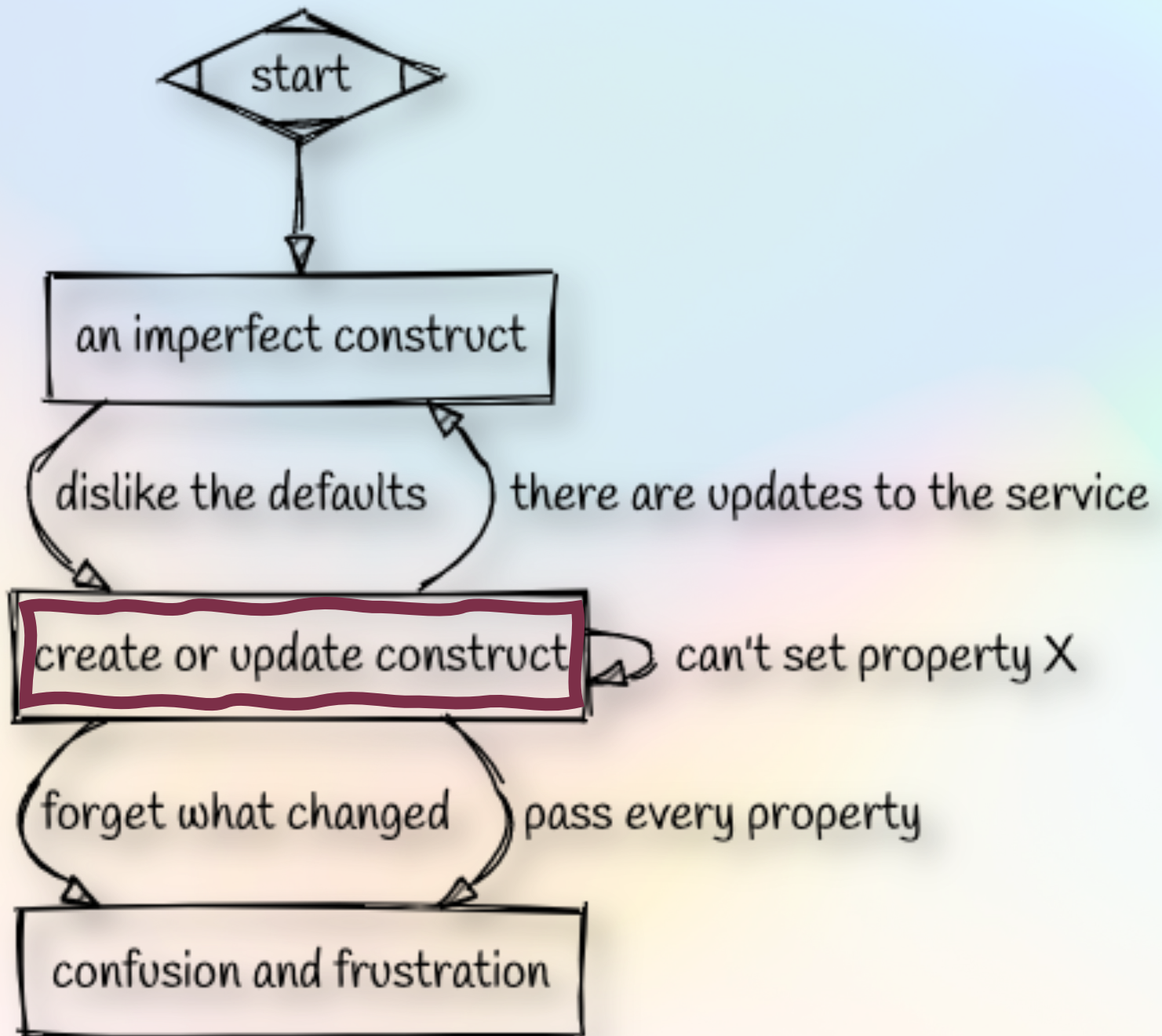


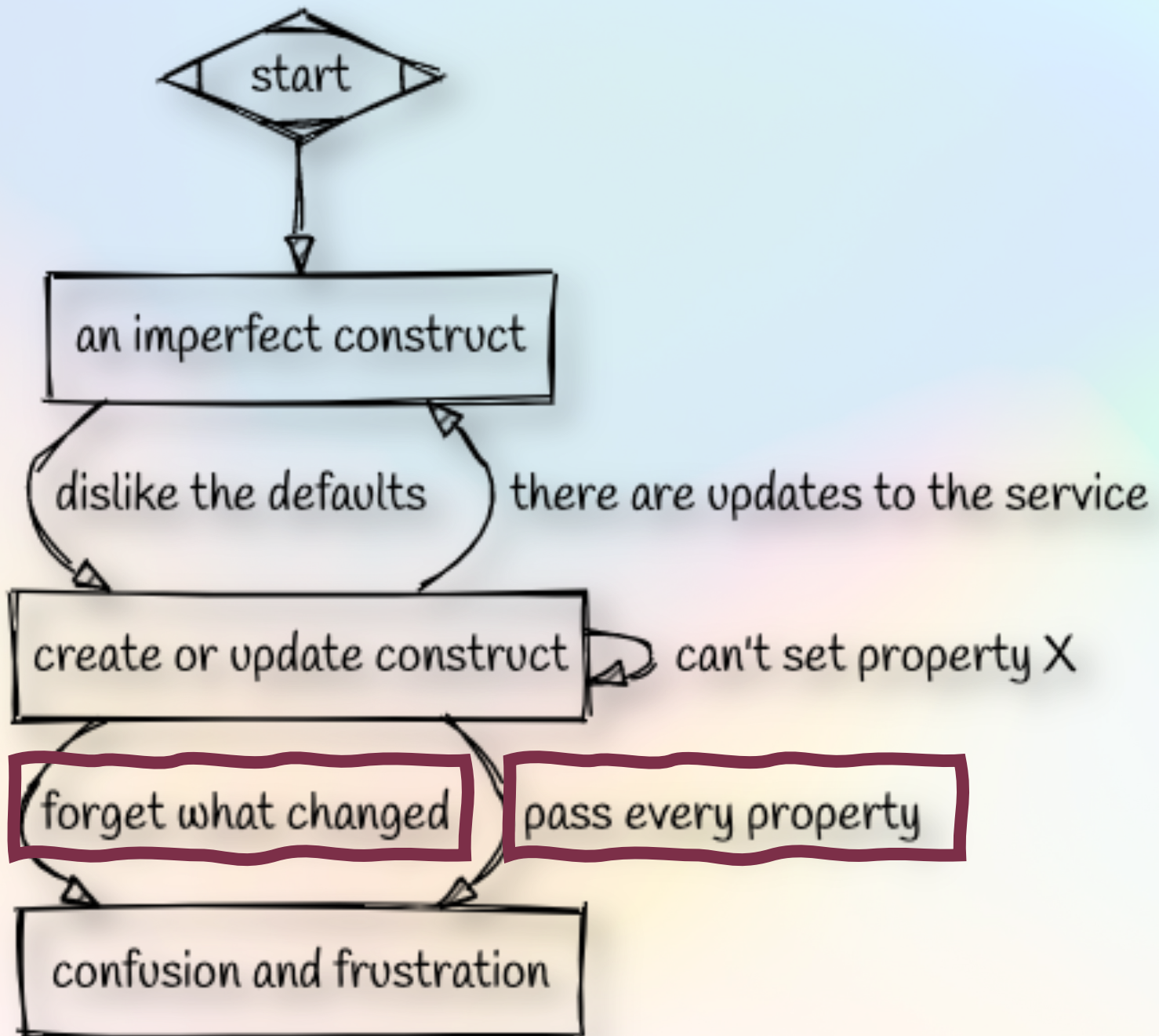


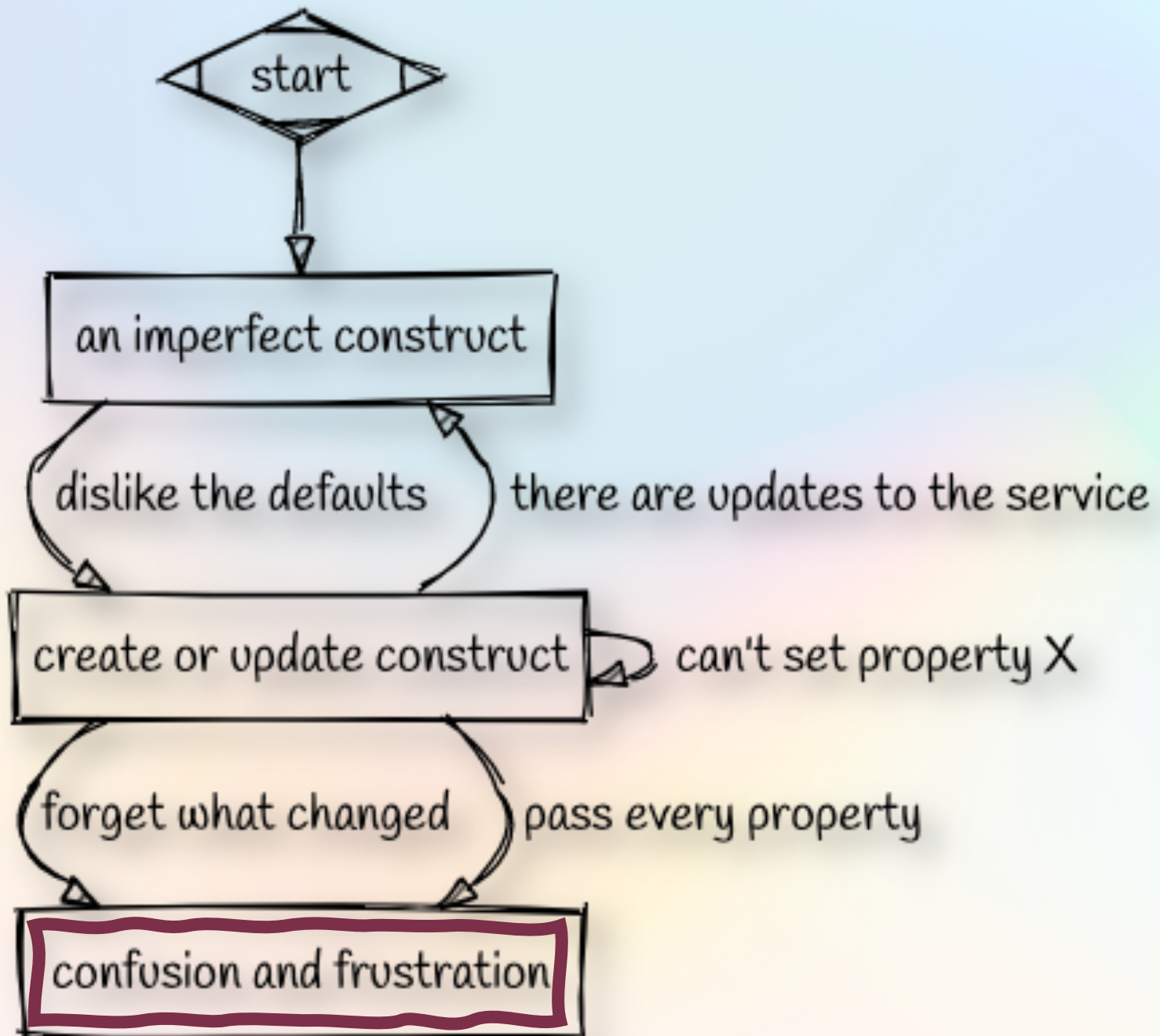












New – Simplify Access Management for Data Stored in Amazon S3

by Marcia Villalba | on 30 NOV 2021 | in [Amazon Simple Storage Service \(S3\)](#), [Announcements](#), [AWS IAM Access Analyzer](#), [AWS Re:Invent](#), [News](#), [Storage](#) | [Permalink](#) | [Share](#)

▶ 0:00 / 0:00

Voiced by [Amazon Polly](#)

Today, we are introducing a couple new features that simplify access management for data stored in [Amazon Simple Storage Service \(Amazon S3\)](#). First, we are introducing a new [Amazon S3 Object Ownership](#) setting that lets you disable access control lists (ACLs) to simplify access management for data stored in Amazon S3. Second, the Amazon S3 console policy editor now reports security warnings, errors, and suggestions powered by [IAM Access Analyzer](#) as you author your S3 policies.

<https://aws.amazon.com/blogs/aws/new-simplify-access-management-for-data-stored-in-amazon-s3/>

Disabling ACLs for all new buckets and enforcing Object Ownership

[PDF](#) | [RSS](#)

We recommend that you disable ACLs on your Amazon S3 buckets. You can do this by applying the bucket owner enforced setting for S3 Object Ownership. When you apply this setting, ACLs are disabled and you automatically own and have full control over all objects in your bucket. You can require that all new buckets are created with ACLs disabled by using AWS Identity and Access Management (IAM) policies or AWS Organizations service control policies (SCPs), as described in the next section.

<https://docs.aws.amazon.com/AmazonS3/latest/userguide/ensure-object-ownership.html>

Chapter 3: Lessons Learned

don't create
L2.5 constructs

**use governance
tools instead**

**abstractions have
a name**

**abstractions do
one thing**

**abstractions are
hard**

In summary

In summary

- Do you need to be clever? **Examples** and **scanning** works too
- have a way to build **mechanical sympathy**
- custom resources are **applications**
- use CloudFormation features like **private resource types**
- **abstractions** are hard

- Bonus: avoid side effects and state

Thank you!

Ben Bridts

ben@cloudar.be

@BenBridts | @WeAreCloudar

www.cloudar.be



Premier
Consulting
Partner

Solution Provider

Public Sector Partner

MSP Partner

Immersion Day Partner

DevOps Competency

Migration Competency

Government Competency

Well Architected

AWS Lambda

