



Self Healing Systems using Distributed OSGi

ApacheCon: Core Europe 2015

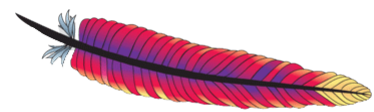
Pepijn Noltes
pepijnnoltes@gmail.com
@pnoltes

Björn Petri
bjoern.petri@sundevil.de



Agenda

- Introduction Services & OSGi ~ 5 min
- Demo Dynamic Services ~ 10 min
- Self Healing Architecture ~ 5 min
- Demo Self Healing System ~ 10 min
- Wrap-up ~ 5 min



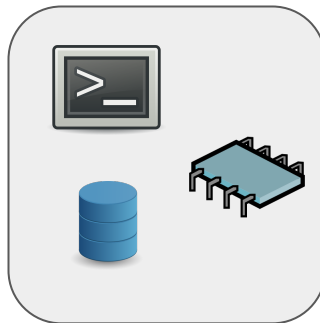
Introduction Services & OSGi



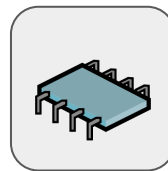
What are Microservices?

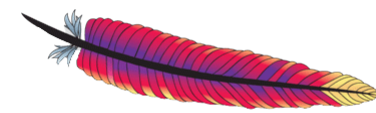
- components as services
- application partitioning
- modular responsibility
- isolated
- technology diversity

Monolith

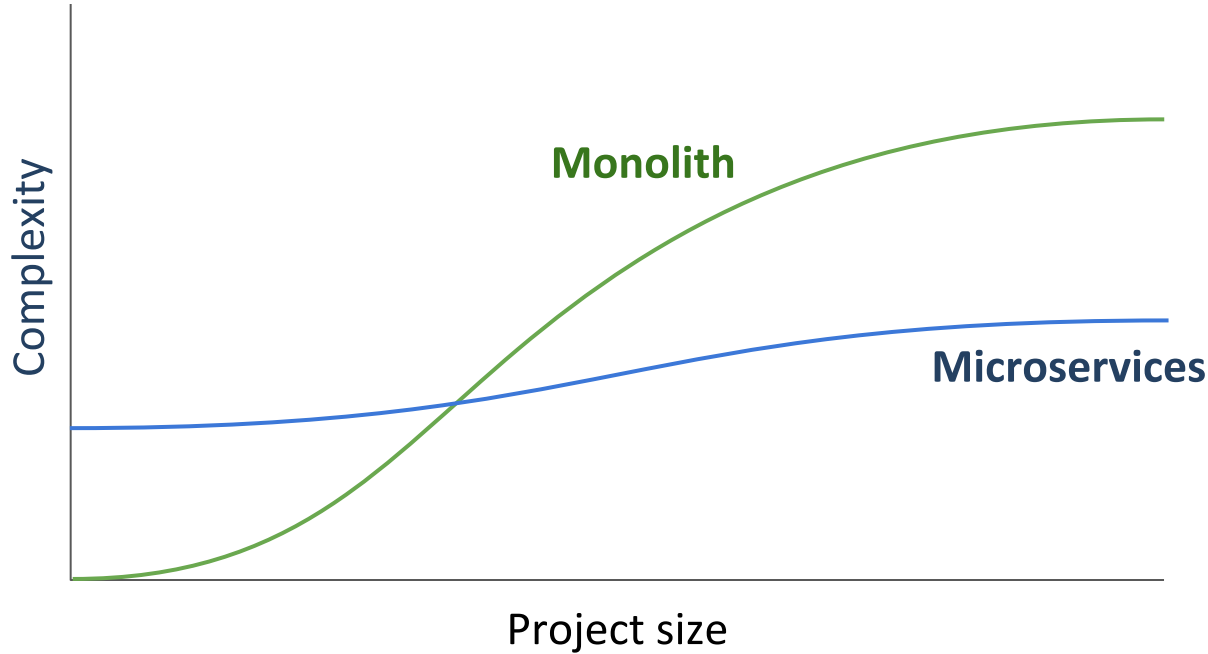


Microservices



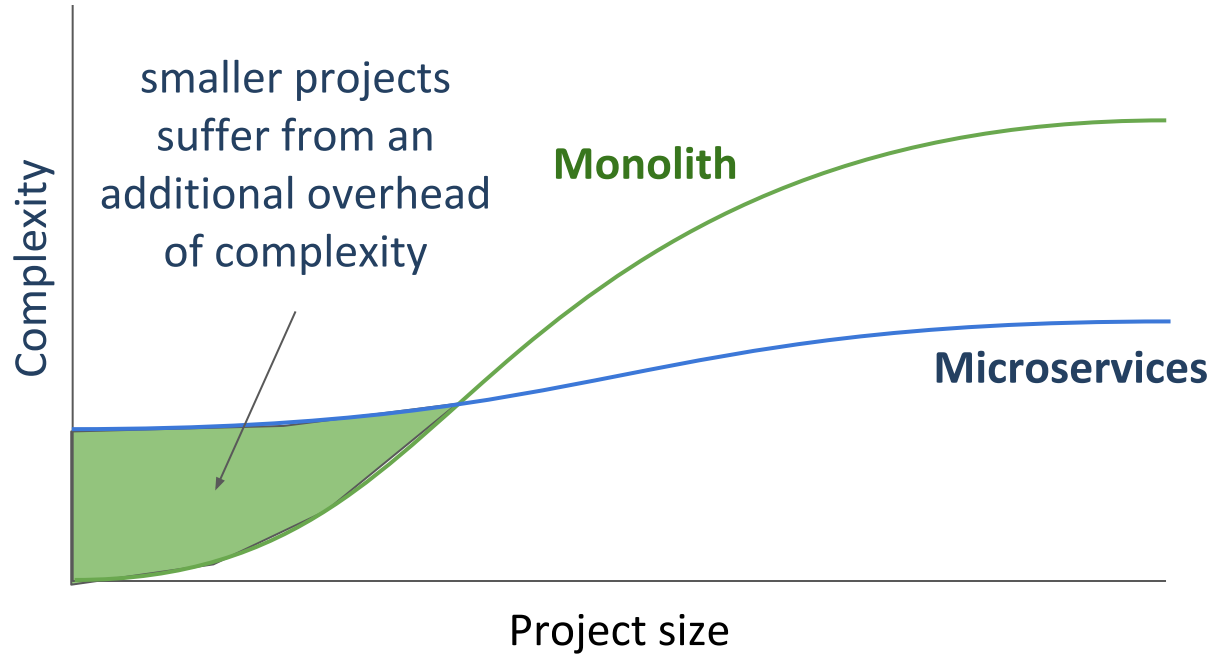


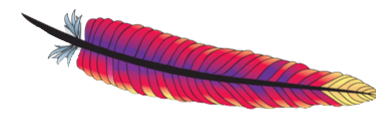
Streamline your Development



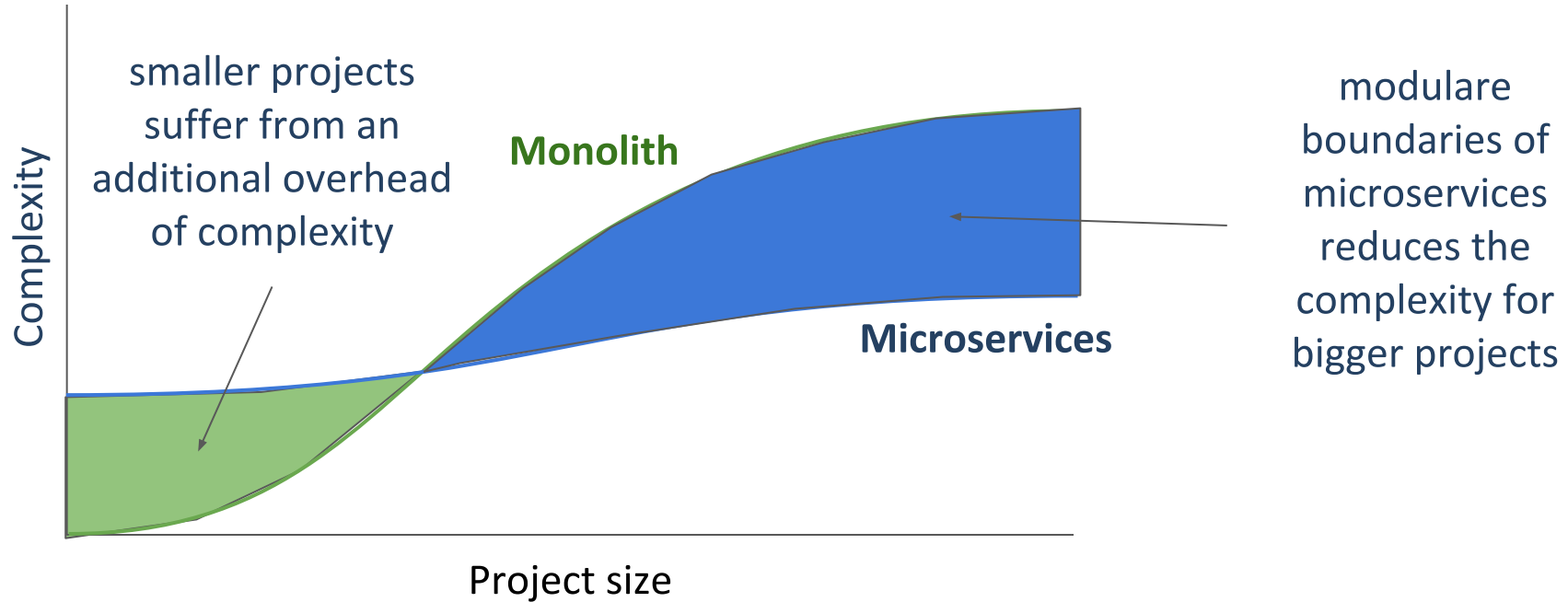


Streamline your Development





Streamline your Development





Rebuilt it all ... while running

- independently deployable
- scalable
- no technology commitment
- easily refactorable

Introduction OSGi



- Specification
- dynamic component model
- (Nano)Services Framework
- By **embedded** system
vendors & network provides





Introduction OSGi



Java implementation of the OSGi framework specification

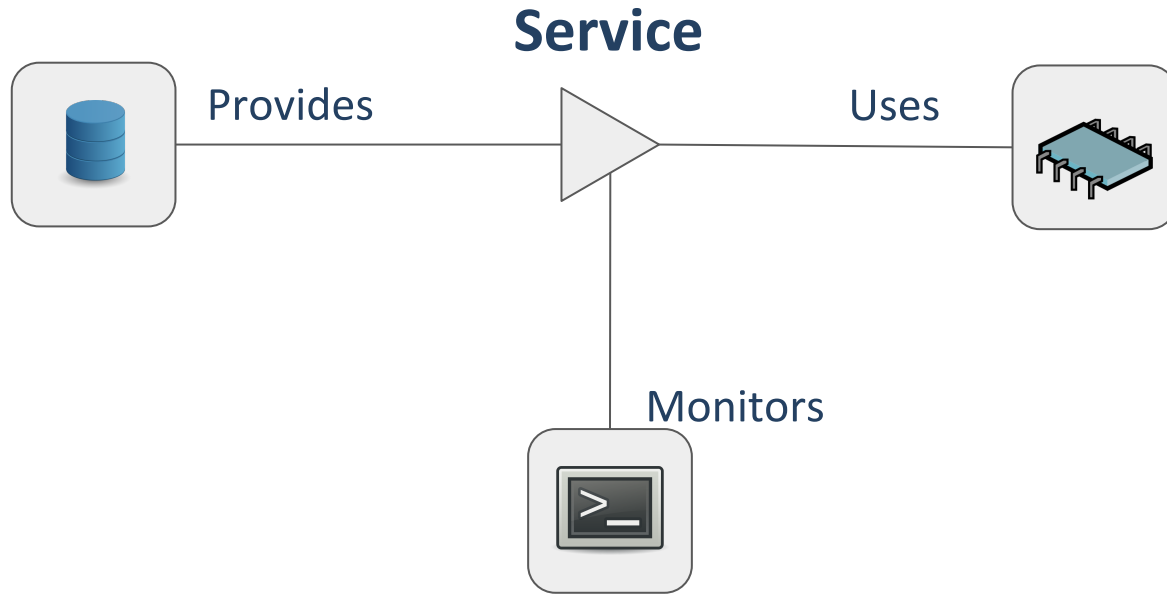


C implementation of the OSGi framework specification



Apache ACE is a provisioning framework

Services



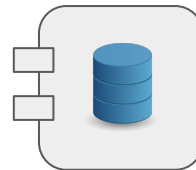
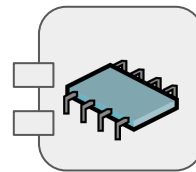
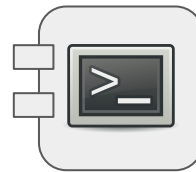
Dynamic Service-Oriented Programming



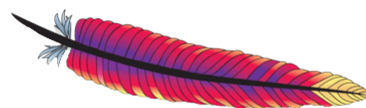
- Services are first class citizens
- Program against interface
- Promote cohesion, prevent coupling
- Favor services composition over inheritance
- Discoverable & Dynamic



Microservices



Service Example



```
public interface Sensor {  
    String getName();  
    Location getLocation();  
    Double getRange();  
}
```



```
struct Sensor {  
    void *handle;  
    int (*getName)(void *handle, char **name);  
    int (*getLocation)(void *handle, location_t **loc);  
    int (*getRange)(void *handle, double *range);  
};
```



Service Usage Example



```
public Viewer {  
    private final List<Sensor> sensors = ...  
    public void addSensor(Sensor s) {  
        sensors.add(s);  
    }  
    public void doGet(...) {  
        ...  
        sensor.getLocation()  
        ...  
    }  
}
```

1

```
struct Viewer {  
    array_list_pt sensors;  
    pthread_mutex_t *mutex;  
};  
void addSensor(struct Viewer *v, struct Sensor *s) {  
    pthread_mutex_lock(v->mutex);  
    arrayList_add(sensors,s)  
    pthread_mutex_unlock(v->mutex)  
}  
void doGet(...) {  
    ...  
    pthread_mutex_lock(v->mutex);  
    char *name = NULL;  
    int rc = sensor->getName(sensor->handle, &name);  
    pthread_mutex_unlock(v->mutex)  
    ...  
}
```

1



Remote Service Usage Example



```
public Viewer {  
    private final List<Sensor> sensors = ...  
    public void addSensor(Sensor s) {  
        sensors.add(s);  
    }  
    public void doGet(...) {  
        ...  
        sensor.getLocation()  
        ...  
    }  
}
```

1

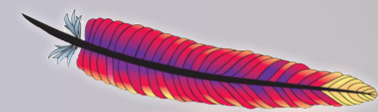
2

```
struct Viewer {  
    array_list_pt sensors;  
    pthread_mutex_t *mutex;  
};  
void addSensor(struct Viewer *v, struct Sensor *s) {  
    pthread_mutex_lock(v->mutex);  
    arrayList_add(sensors,s)  
    pthread_mutex_unlock(v->mutex)  
}  
void doGet(...) {  
    ...  
    pthread_mutex_lock(v->mutex);  
    char *name = NULL;  
    int rc = sensor->getName(sensor->handle, &name);  
    pthread_mutex_unlock(v->mutex)  
    ...  
}
```

1

2





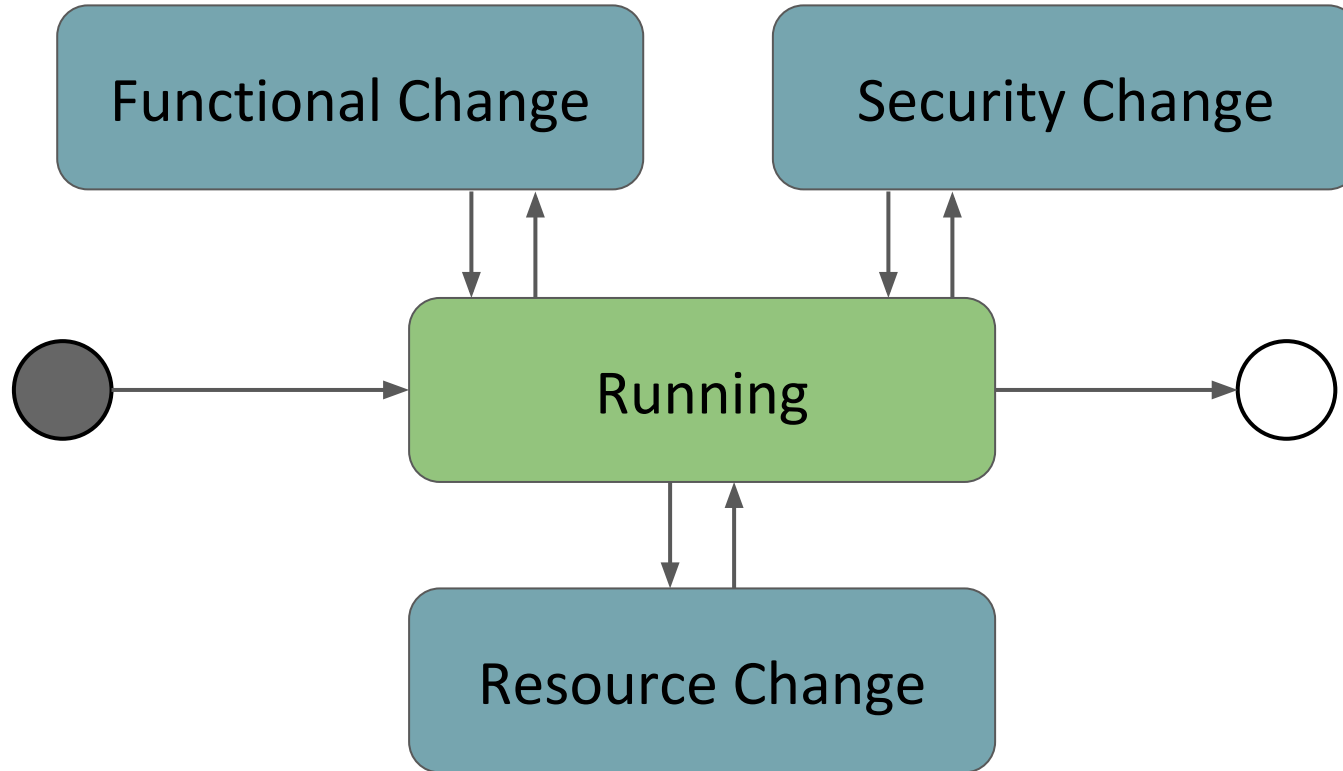
Demo

Dynamic Services



Self Healing Architecture

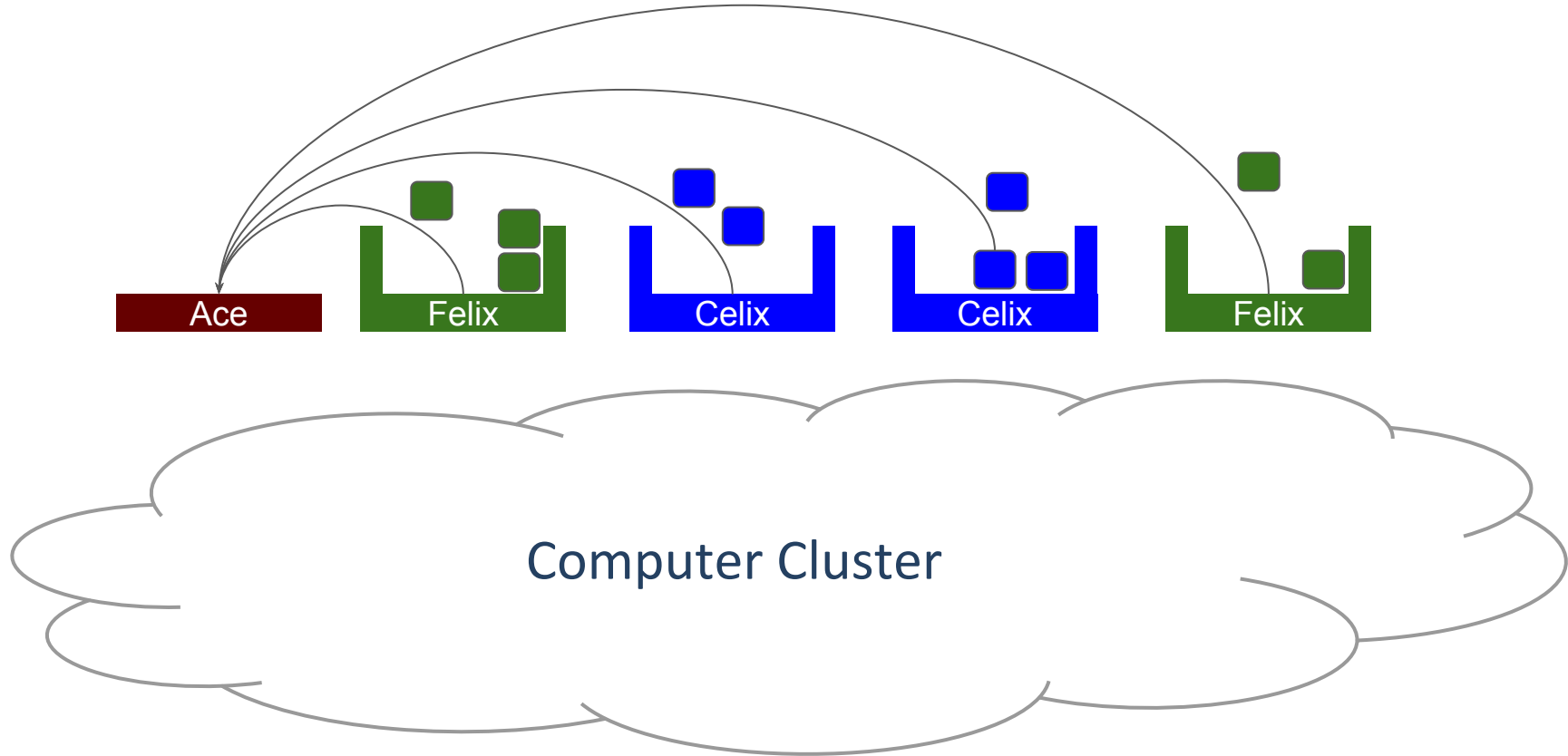
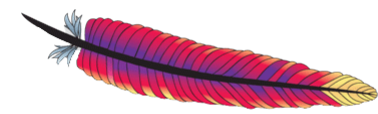
Dynamic Services delivers an evolvable Application



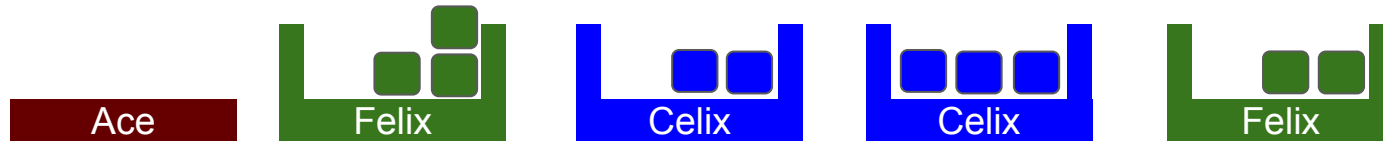
Dynamic Application - Startup



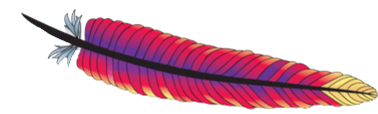
Dynamic Application - Provisioning



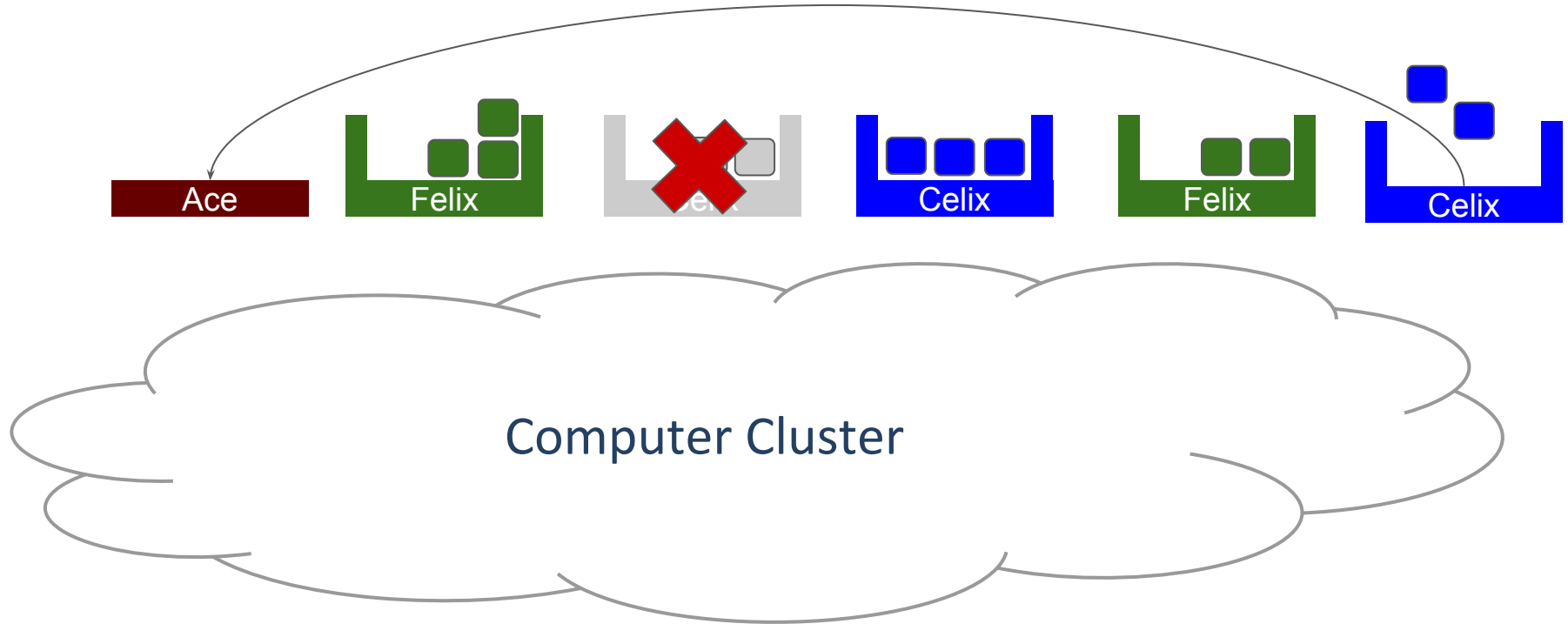
Dynamic Application - Running



Dynamic Application - Hardware failure



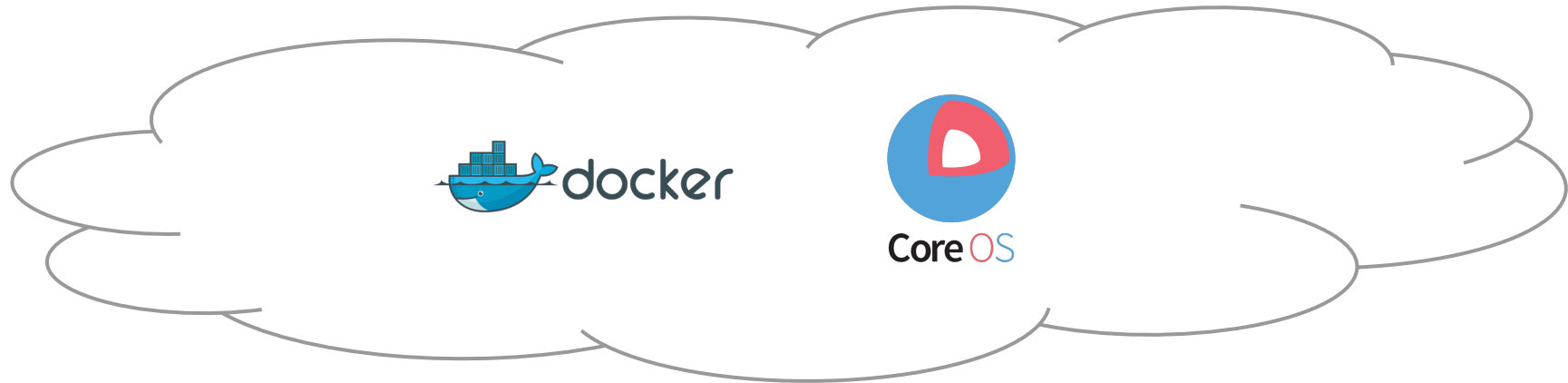
Dynamic Application - Self Healing

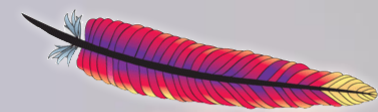


Dynamic Application - Running



Dynamic Application - Running



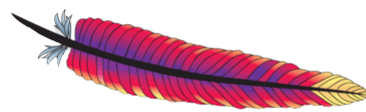


Demo

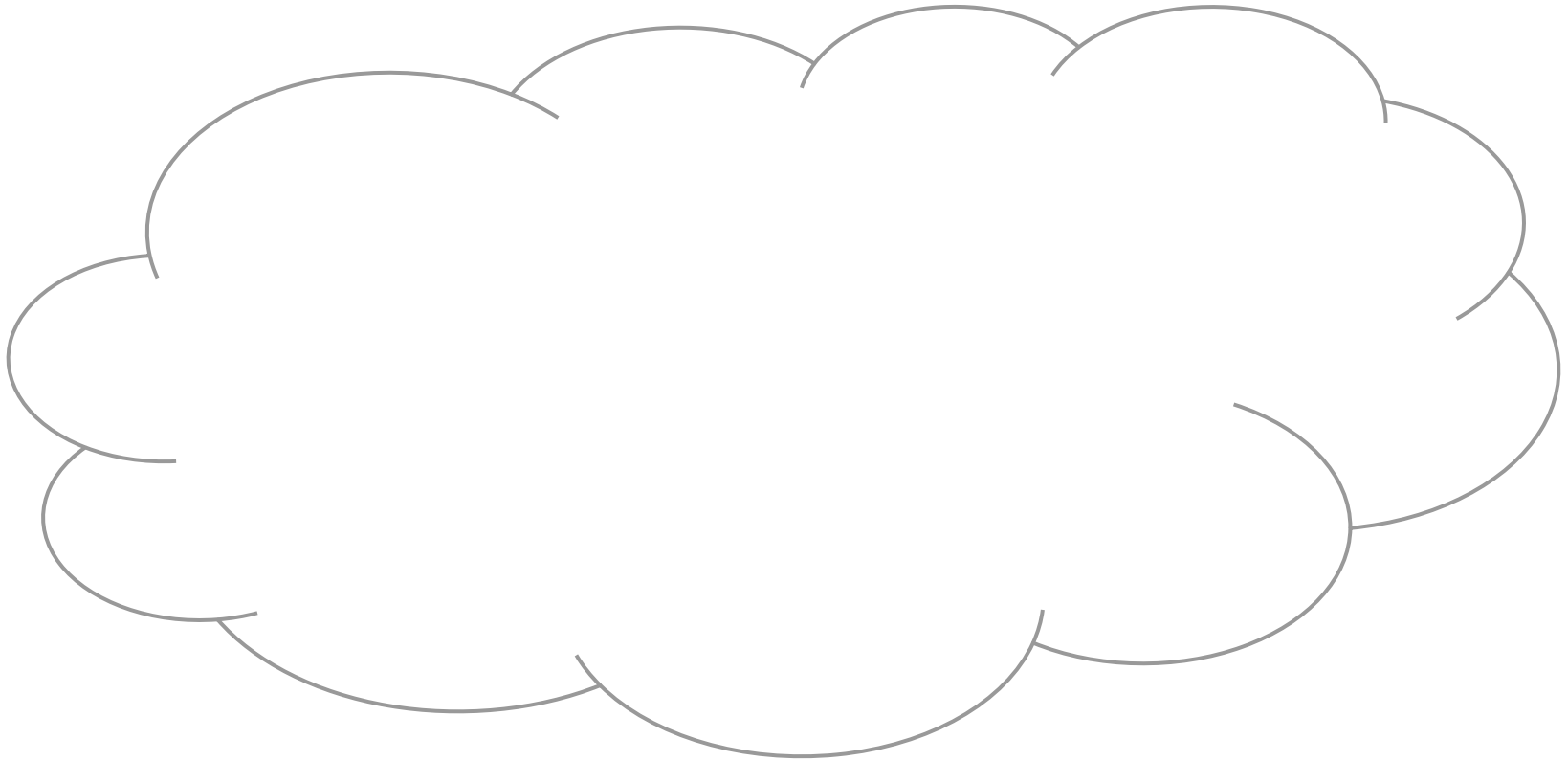
Self Healing System



Wrap-up

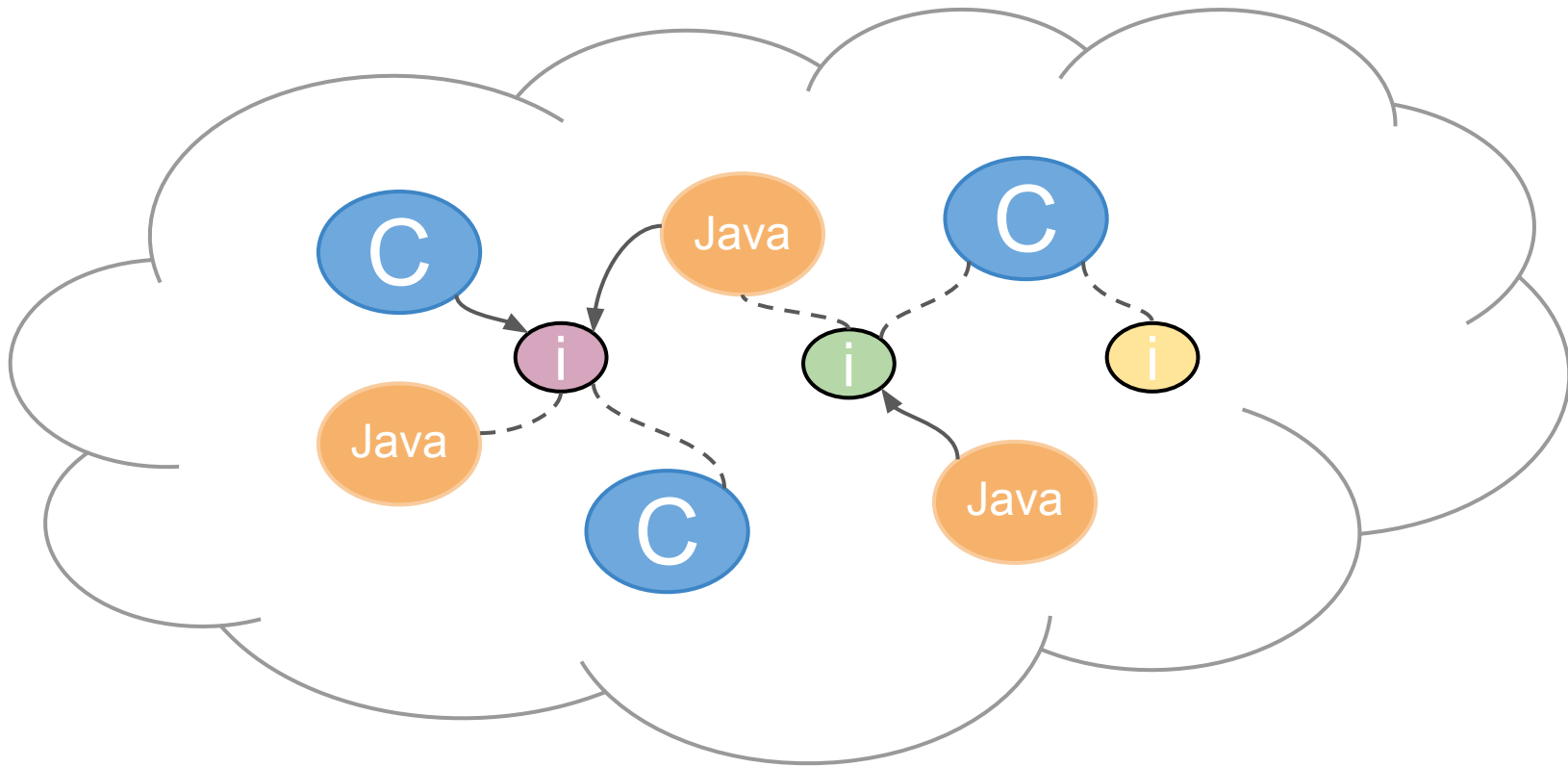


In a nutshell



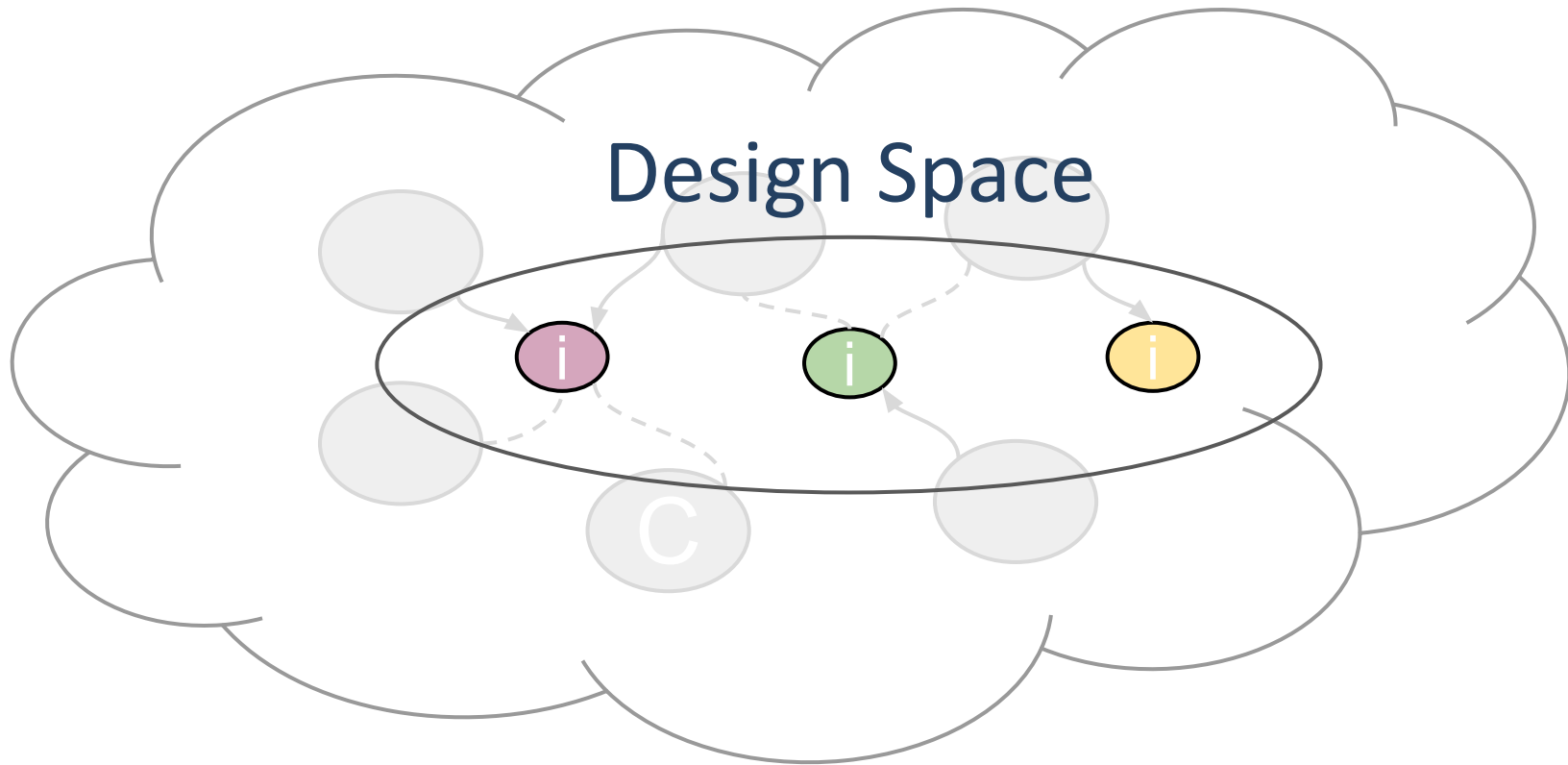


In a nutshell



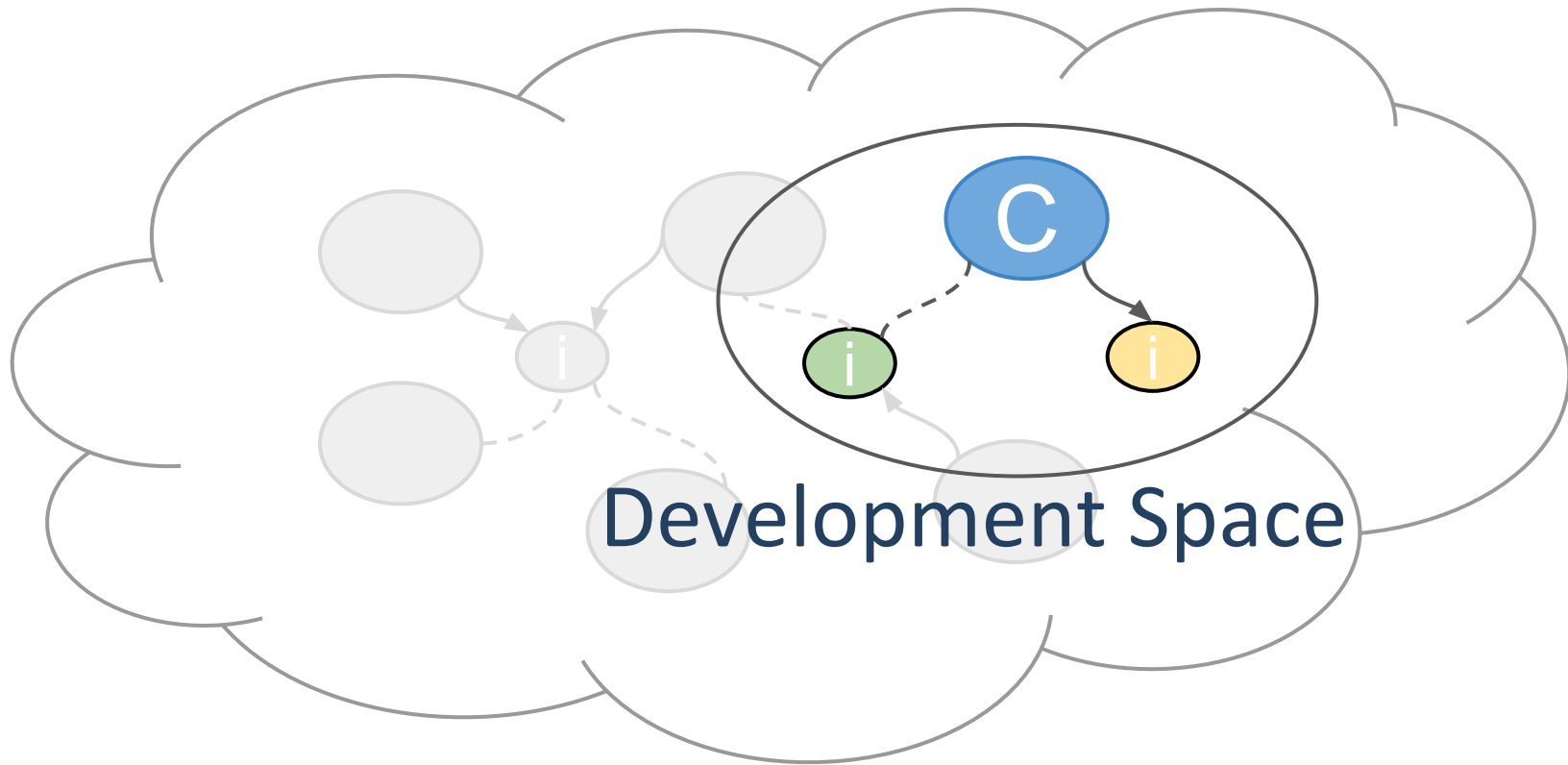


In a nutshell



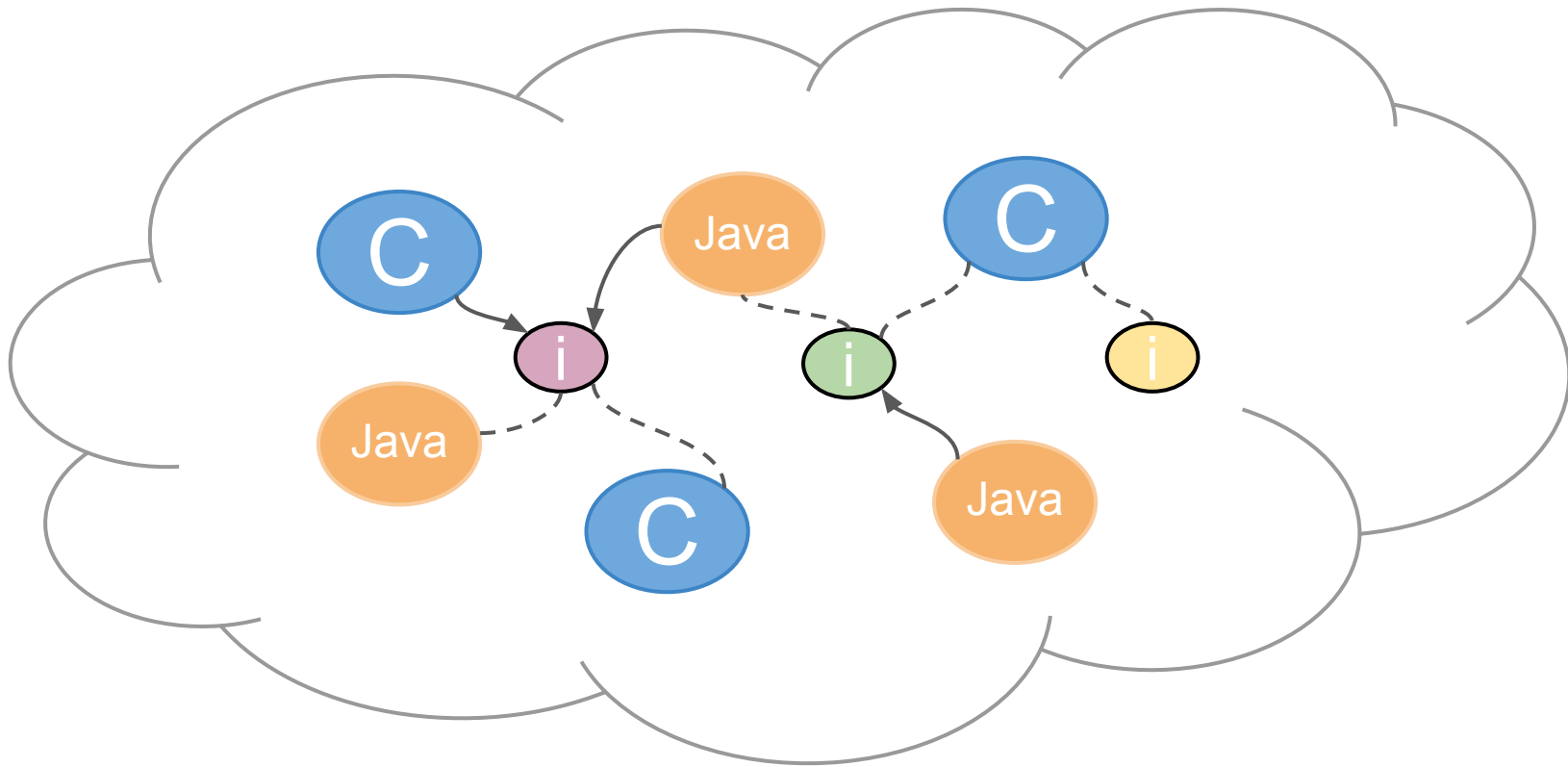


In a nutshell



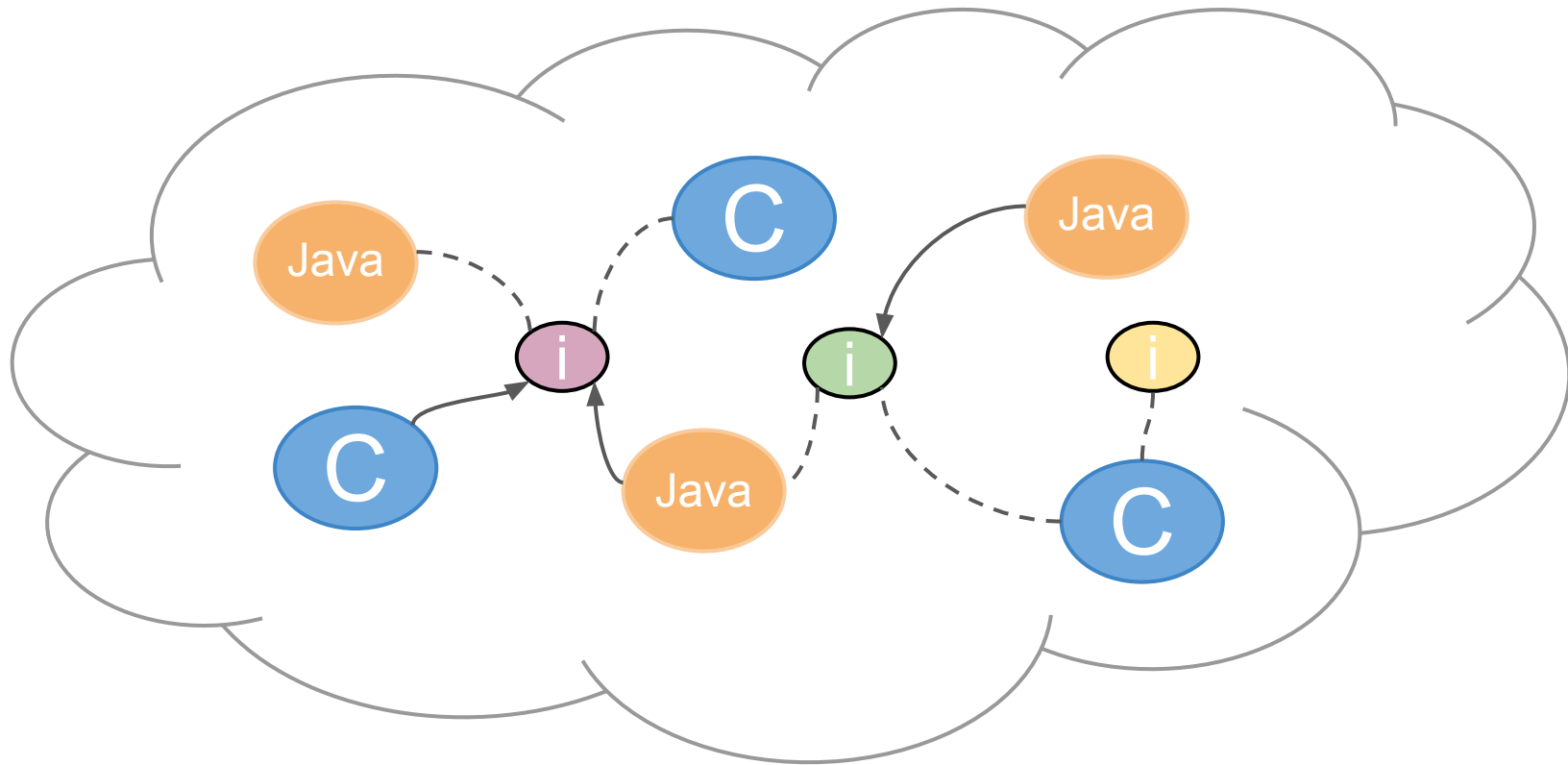


In a nutshell





In a nutshell





Thank You!



Links

- Apache Ace - <https://ace.apache.org>
<https://github.com/apache/ace>
- Apache Celix - <https://celix.apache.org>
<https://github.com/apache/celix>
- Apache Felix - <https://felix.apache.org>
<https://github.com/apache/felix>
- INAETICS - <http://www.inaetics.org>
<https://github.com/INAETICS>

