



DESAL

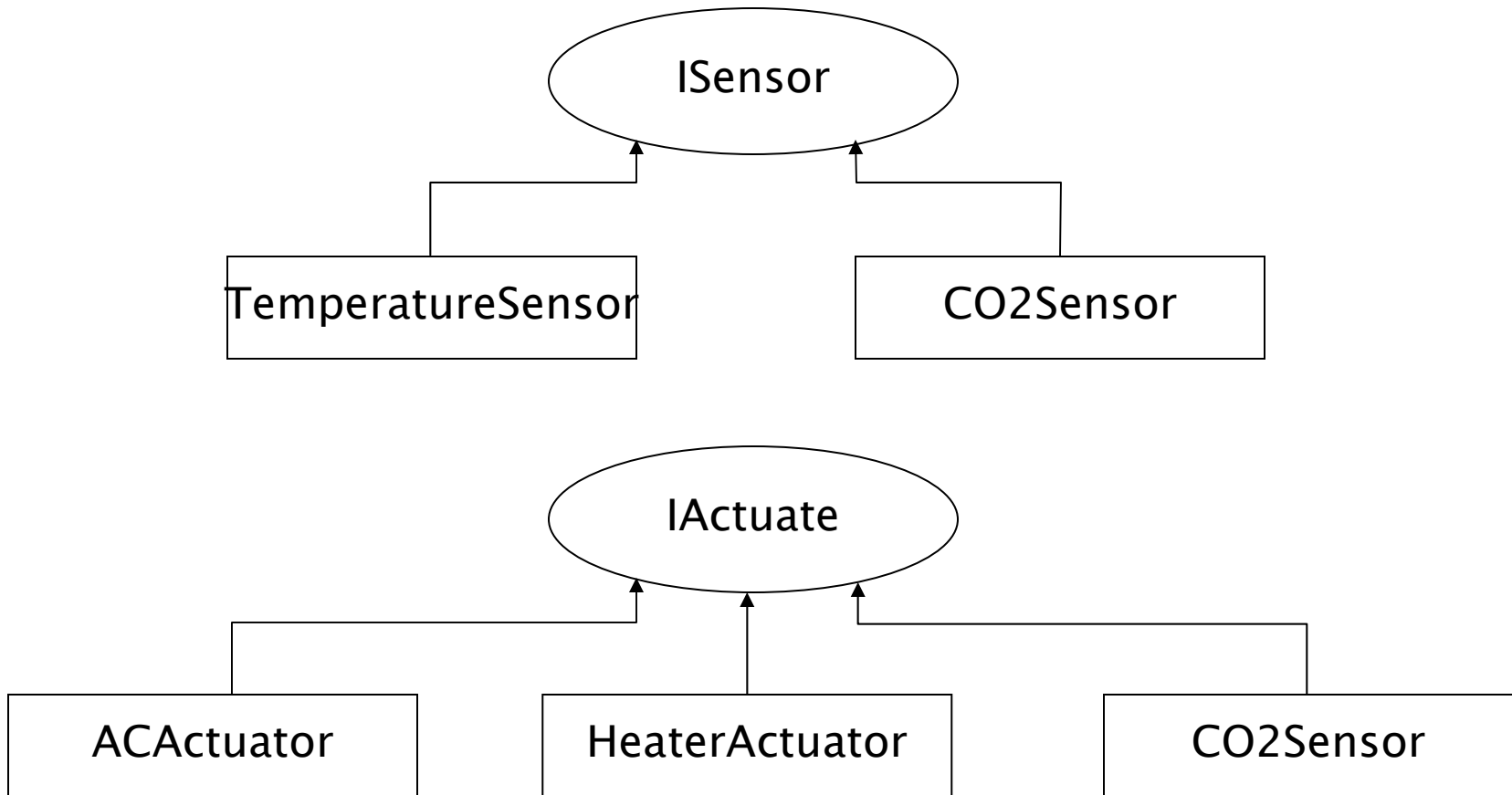
Dynamic Embedded Sensing and Actuation
Language

Interfaces

```
public interface ISensor
{
    double Sense();
}
```

```
public interface IActuator
{
    void Actuate(double dblValue);
    bool isOn();
}
```

Component Structure



TemperatureControl Example

```
control_module TemperatureControl  
begin
```

Parameters:

```
    string strTempSensName;  
    string strHeatActName;  
    string strACActName;
```

```
    double dblTargetTemp;  
    double dblUBTrigger;  
    double dblLBTrigger;  
    double dblPeriodicity;
```

State:

```
    ISensor snsTemp=  
        Lookup(strTempSensName);  
    IActuator actHeat=  
        Lookup(strHeatActName);  
    IActuator actAC=  
        Lookup(strACActName);
```

Initially:

```
    actHeat.Actuate(0);  
    actAC.Actuate(0);
```

Body:

```
((snsTemp.Sense() <= dblLBTrigger) and  
(not actHeat.isOn()))
```

→

```
actHeat.Actuate(1);
```

```
((snsTemp.Sense() >= dblUBTrigger)  
and  
(not actAC.isOn()))
```

→

```
actAC.Actuate(1);
```

```
((snsTemp.Sense() <= Target) and  
(actAC.isOn()))
```

→

```
actAC.Actuate(0);
```

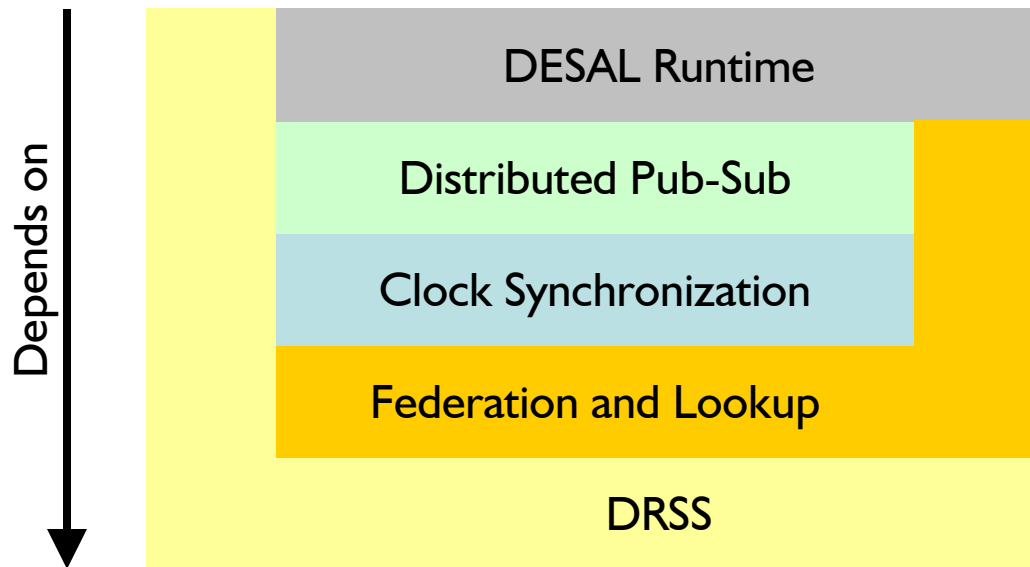
```
((snsTemp.Sense() >= Target) and  
(actHeat.isOn()))
```

→

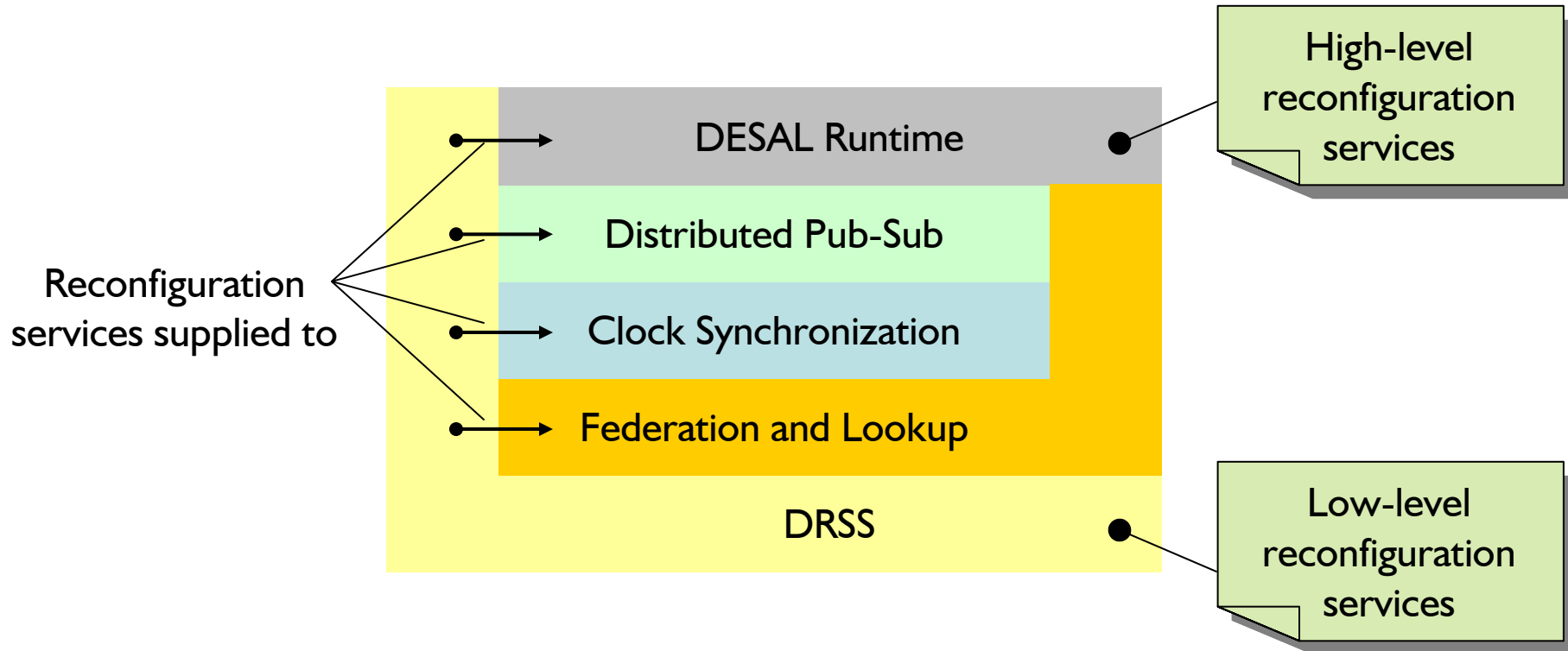
```
actHeat.Actuate(0);
```

```
end TemperatureControl;
```

DESAL Runtime Stack



Reconfiguration in DESAL



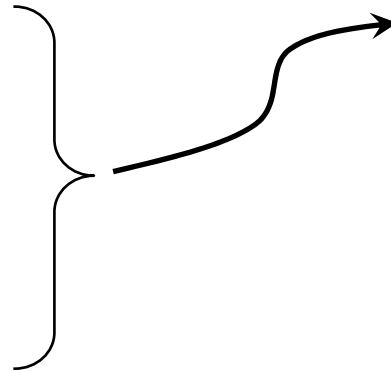
High-Level Reconfiguration Services

- Parameter tuning
- Addition, removal and substitution of control modules
- Module evolution
 - Addition and removal of guarded actions

Low-Level Reconfiguration Services

Objects / Interceptors:

- Deployment
- Removal
- Hot-Swapping
- Observation



Interceptor Chains:

- (Repeated)
- Binding
- Rebinding

Tasks

- Implement ISensor modules
- Implement IActuator modules
- Implement Federation and Lookup services
- Implement Clock Synchronization
- Implement Distributed Pub-Sub
- Implement DESAL Runtime