Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

ELBFISCH

Open-source runtime system for component-based implementation of automation solutions with Java

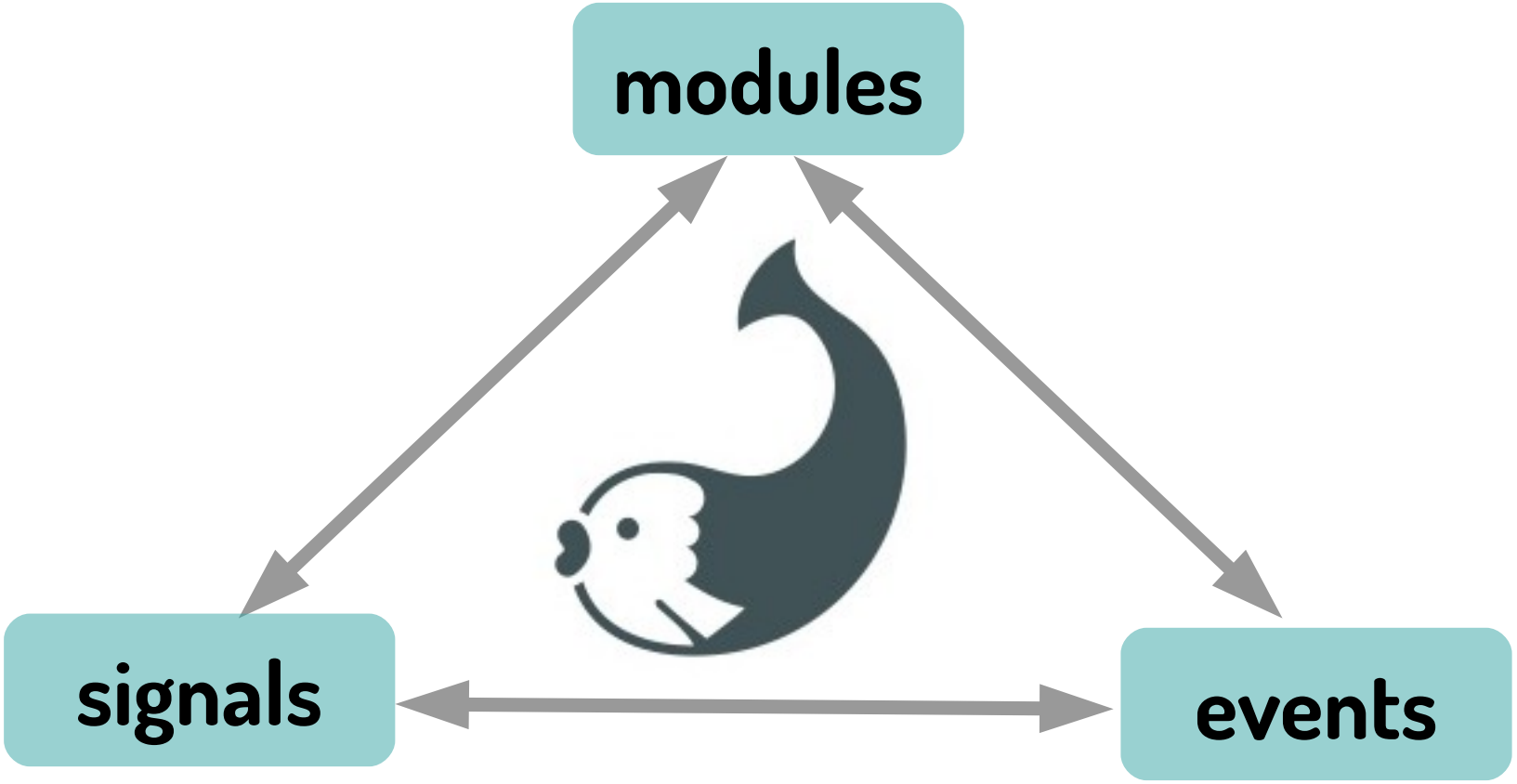
www.elbfisch.org

Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

ELBFISCH

1,2,3 ... really simple:

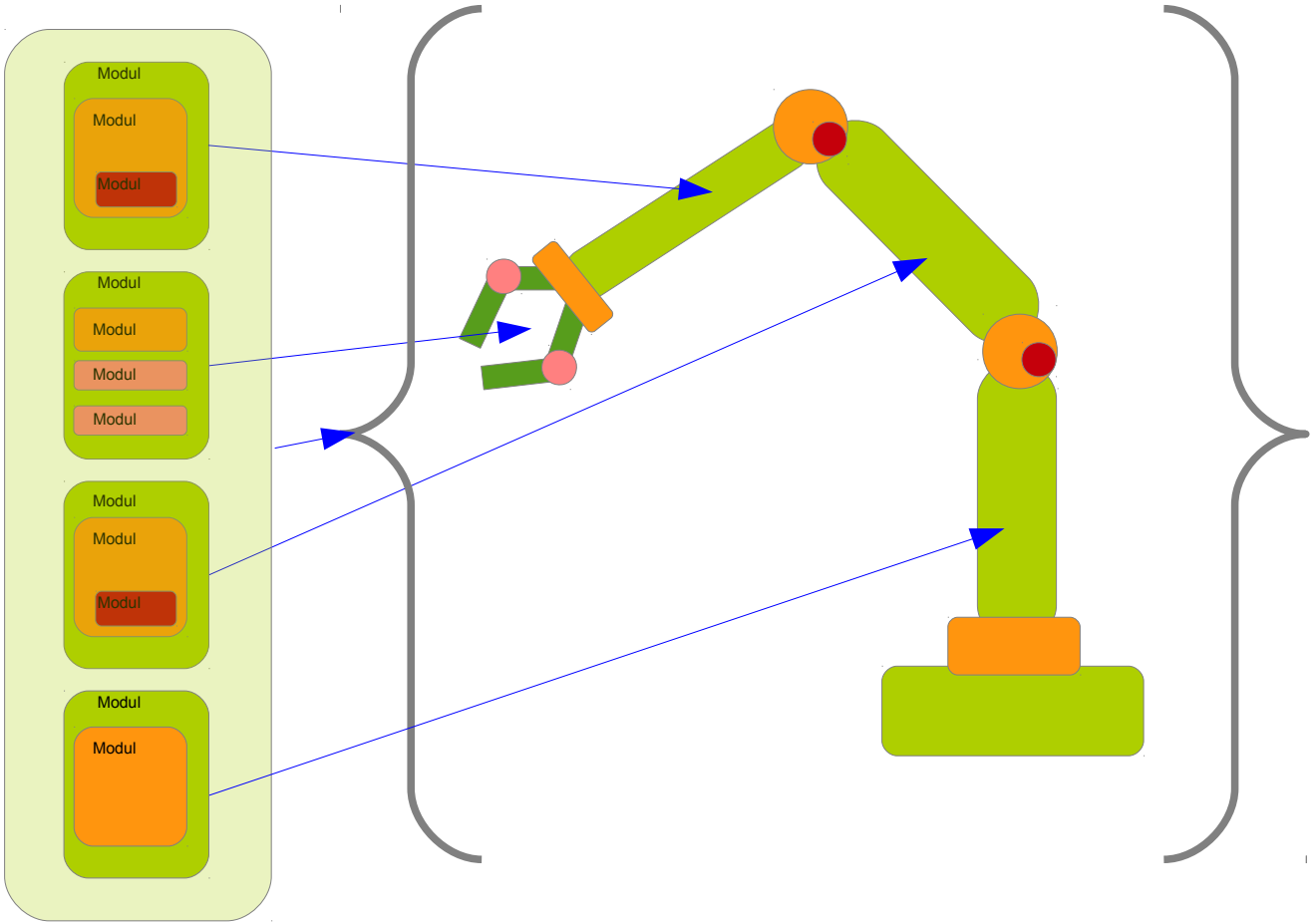


Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

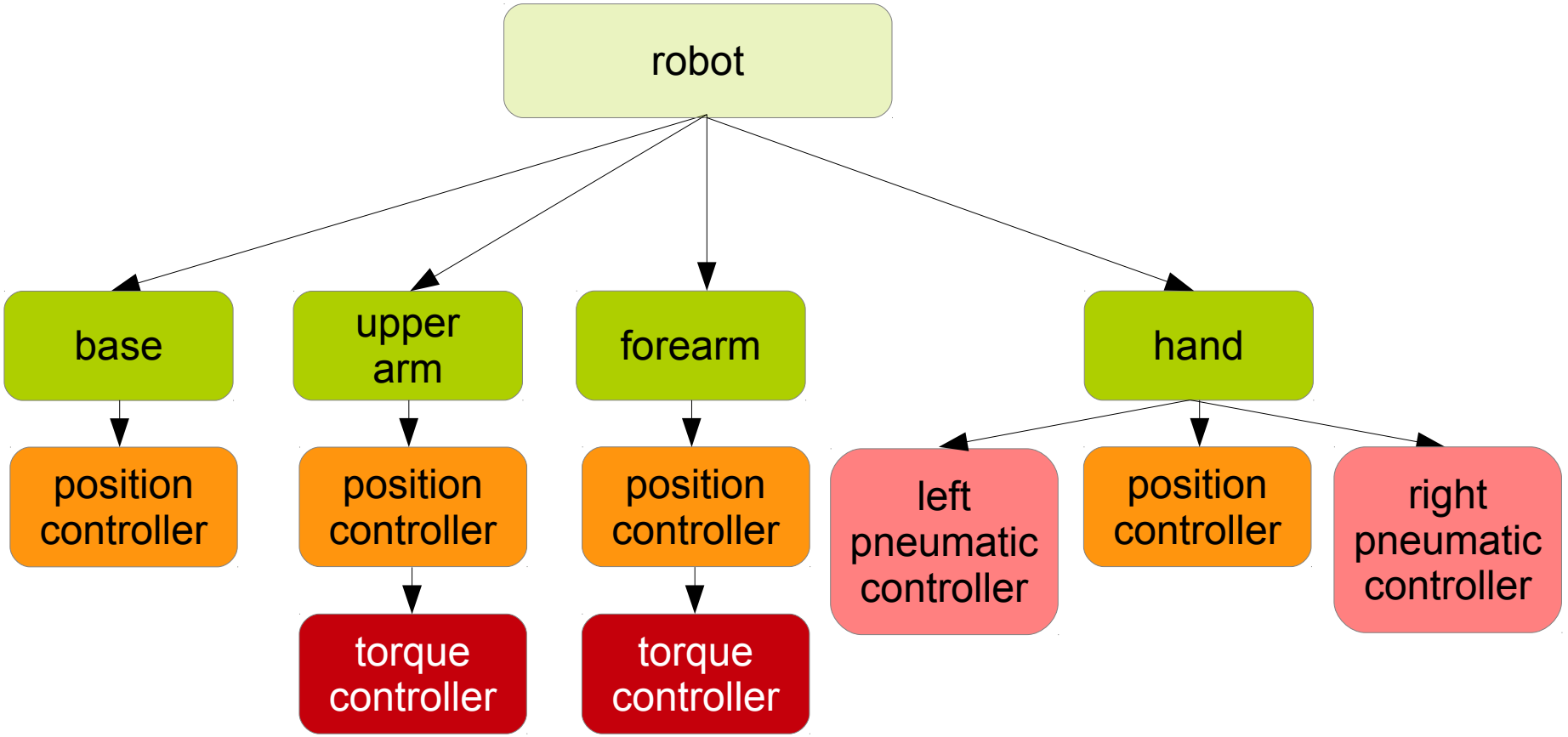
ELBFISCH

Modules – image the world, as it is:

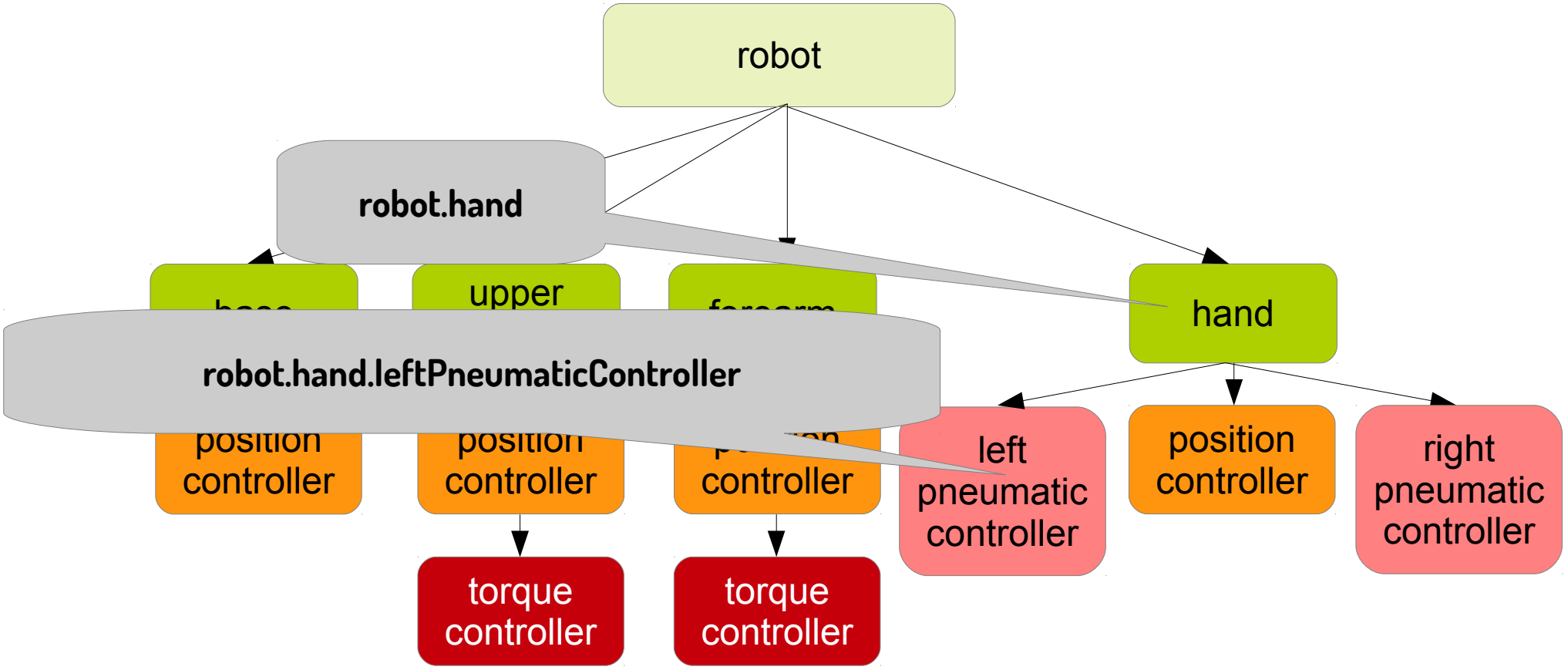


Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

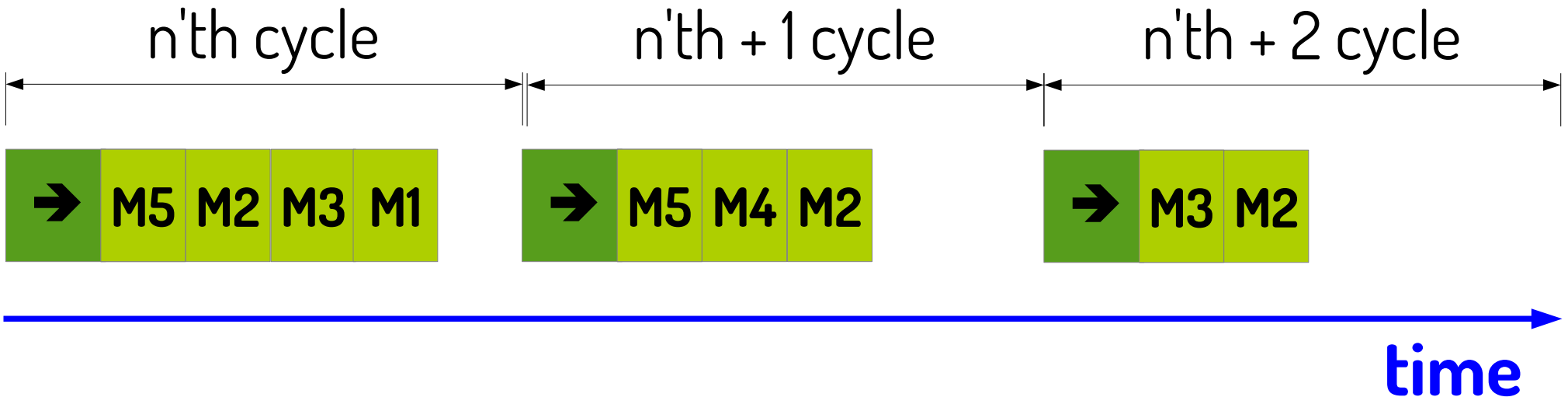
Modules – embedded into hierarchies:



Modules – have unique identifiers:



Modules – from cycle to cycle:



Modules – let's recap ...

Are the IT reproduction of a real world system

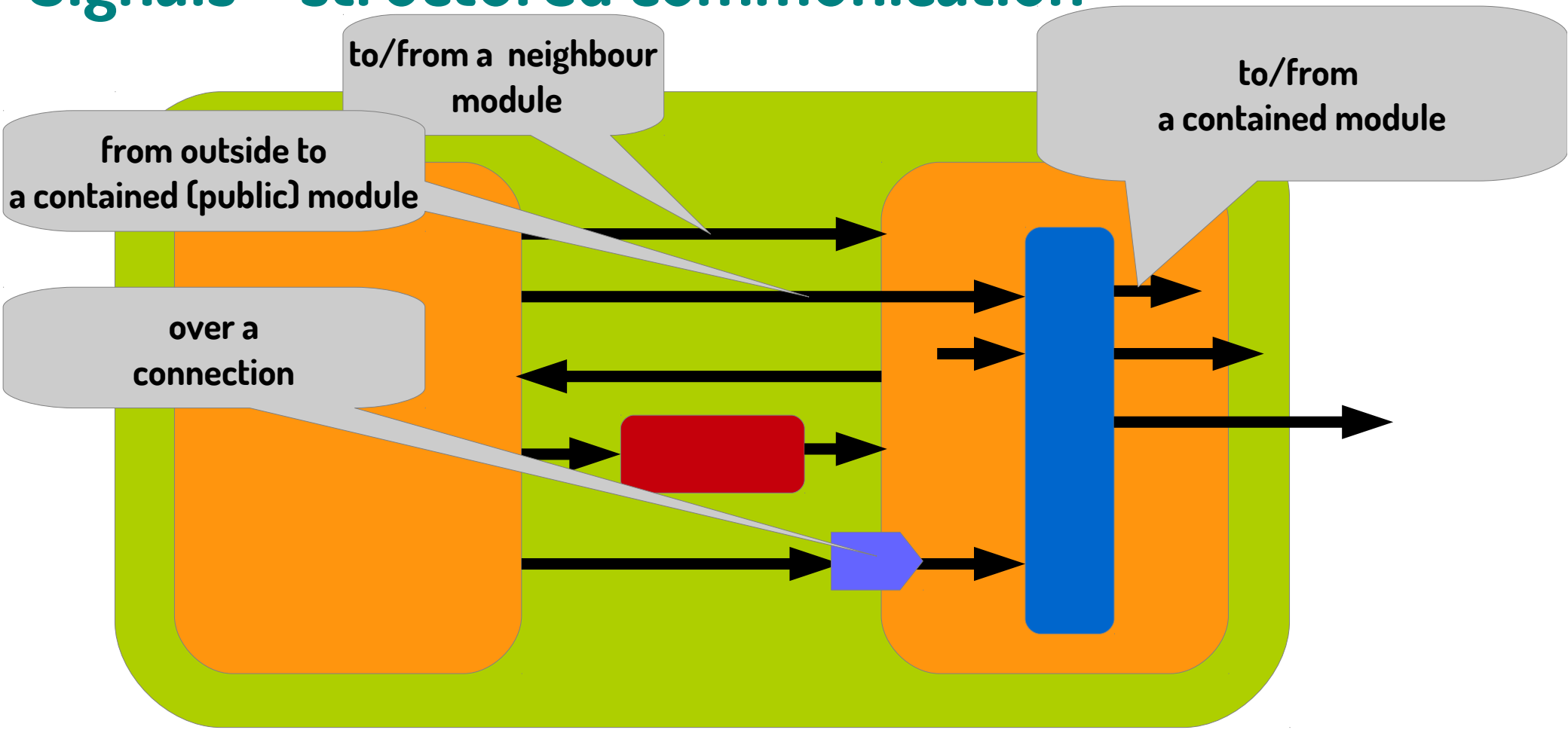
Are part of a hierarchy and uniquely referenceable

Can be implemented free of context

Are run synchronized to a common cycle clock

Are run only, when there is something to do

Signals – structured communication

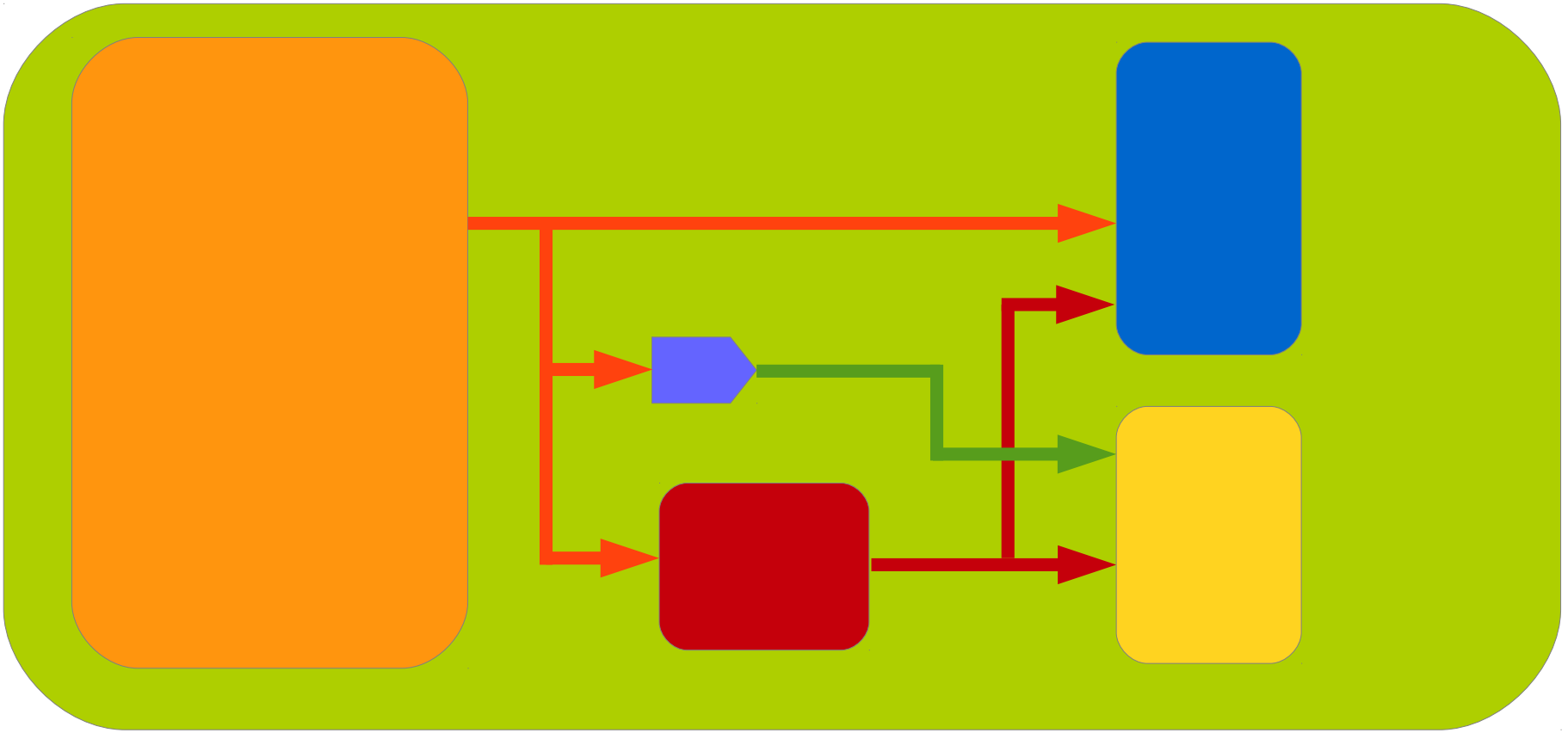


Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

ELBFISCH

Signals – exactly one source – multiple sinks



Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Signals – 4 standard types, more possible ...

Logical : boolean value (boolean)

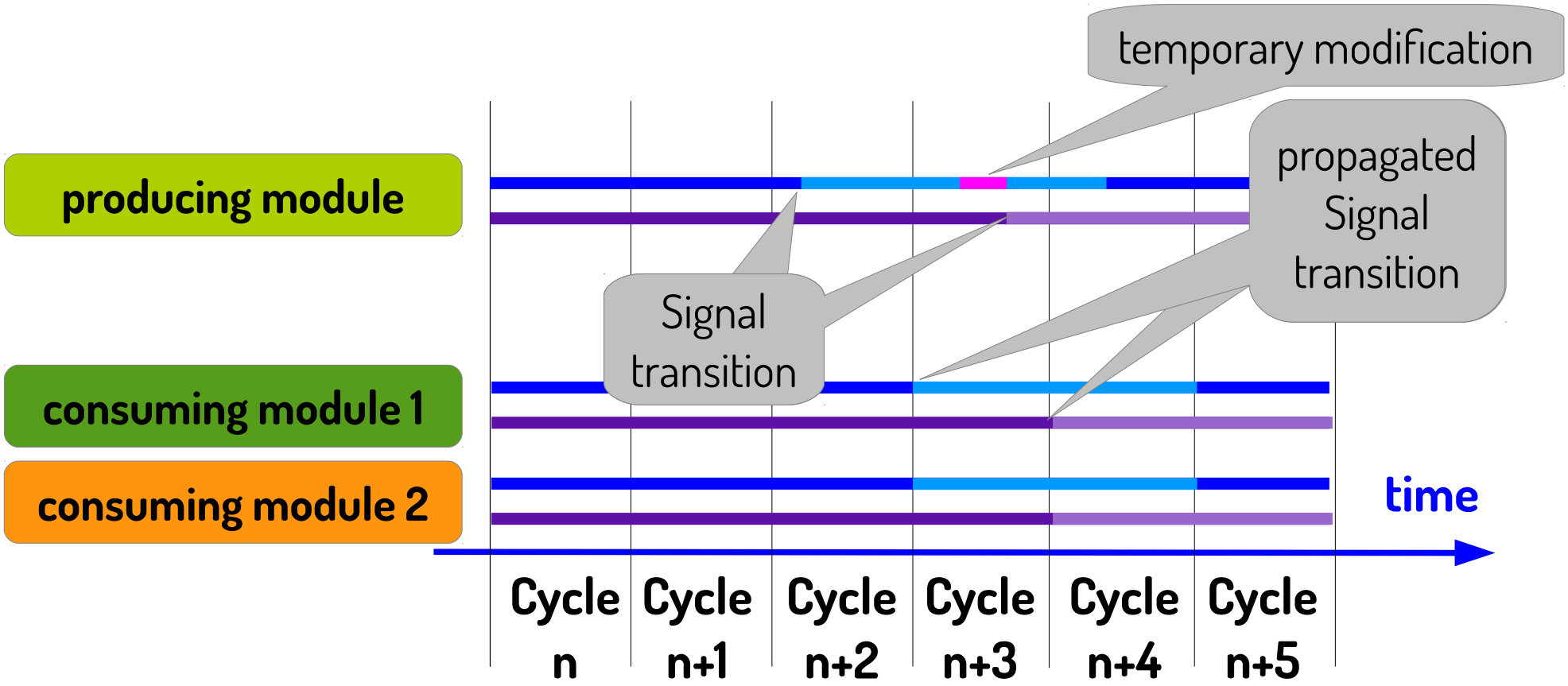
Decimal : floating point value (double)

SignedInteger : integer value (int)

CharString : string value (String)

Generic<ValueImpl> : generic type value

Signals – communication synchronous to a cycle

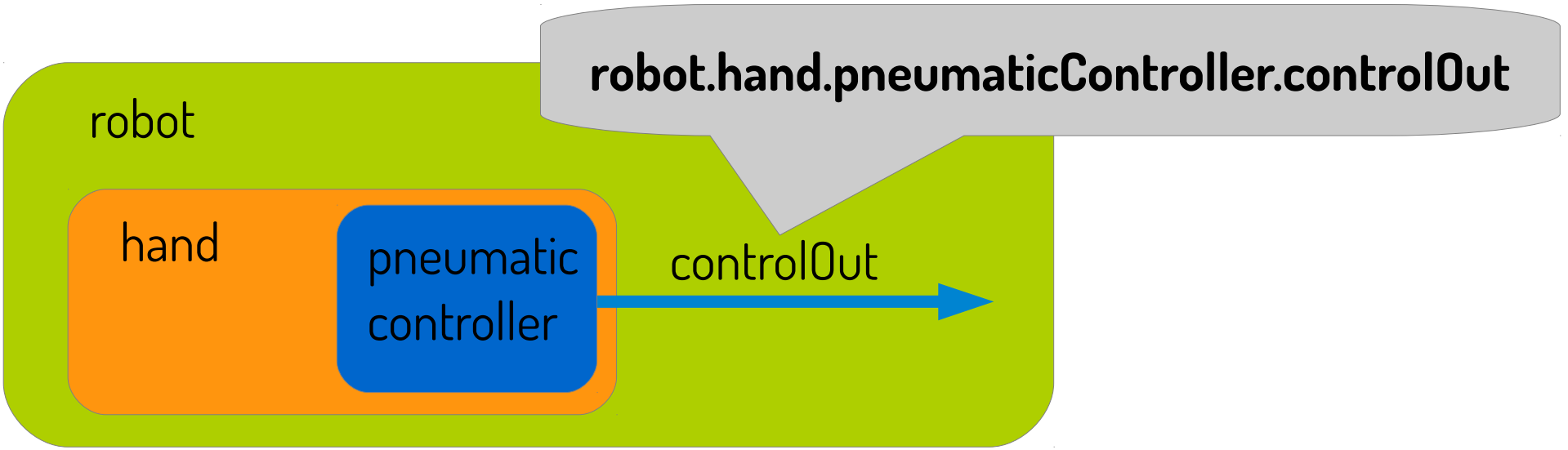


Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Signals – some additional properties

Signals can adopt the states „valid“ and „invalid“

Signals have an unique identifier



Signals – let's recap:

Modules communicate to each other using signals

Signale are of a certain type

Signals can be written by exactly one module,

Signals can be read by many modules

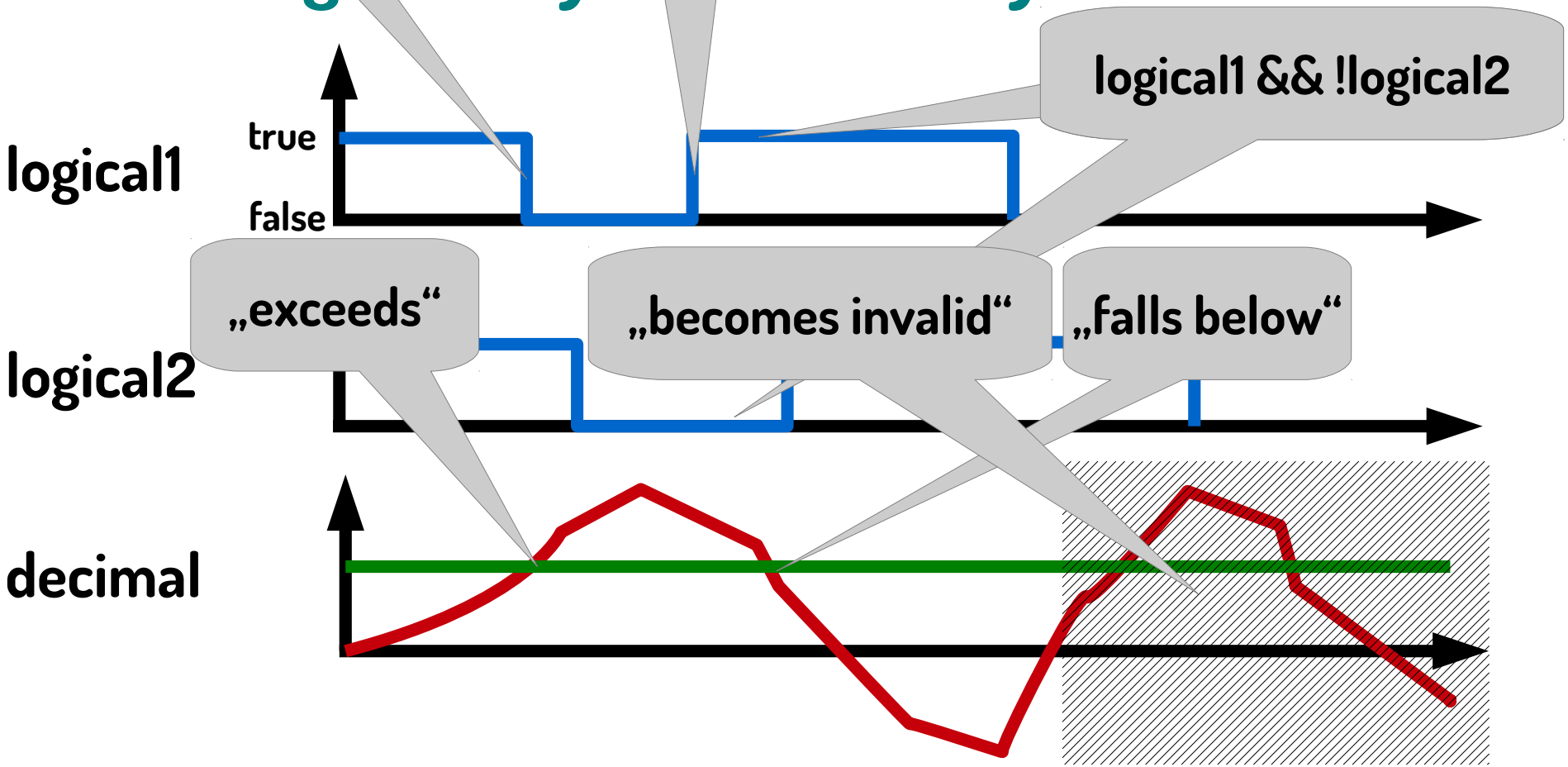
Signals can be connected to each other (if compatible)

Signals are synchronized

Signals can adopt the states „valid“ and „invalid“

Signals are identified by a unique name

Events – imag the dyna ics of a system



Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

ELBFISCH

Events – awaited and processed by modules:

```
do{
    analogInput.becomesValid().await();
    done = false;
    do{
        try{
            do{
                if (analogInput.get() > upperThreshold){
                    switchOn.set(false);
                    analogInput.fallsBelow(lowerThreshold).await();
                }
                if (analogInput.get() < lowerThreshold){
                    switchOn.set(true);
                    analogInput.exceeds(upperThreshold).await();
                }
            }
            while(!finished);
        }
        catch(SignalInvalidException ex){
            switchOn.invalidate();
            done = true;
        }
      }
    while(!done && !finished);
}
while(!finished);
```

Events – easy to implement:

```
private class TeaReadyToServe extends ProcessEvent{
    @Override
    public boolean fire() throws ProcessException{
        boolean ready = false;
        switch(tea){
            case GREENTEA:
                ready = actualBrewTime > 150;
                break;
            case BLACKTEA:
                ready = actualBrewTime > 180;
                break;
            case HERBTEA:
                ready = actualBrewTime > 300;
                break;
        }
        return ready;
    }
}
```

Events – let's recap:

Events image the dynamics of a system

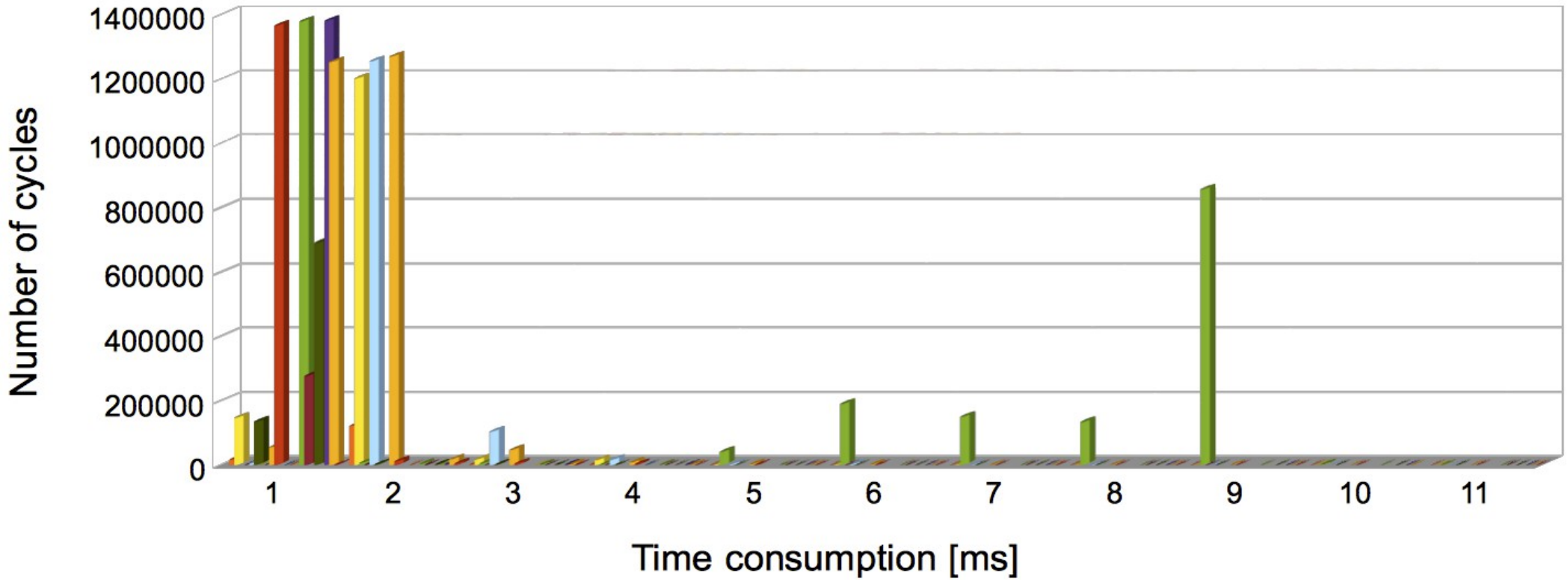
Events can be explicitly awaited by modules

Application specific events can be easily implemented

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

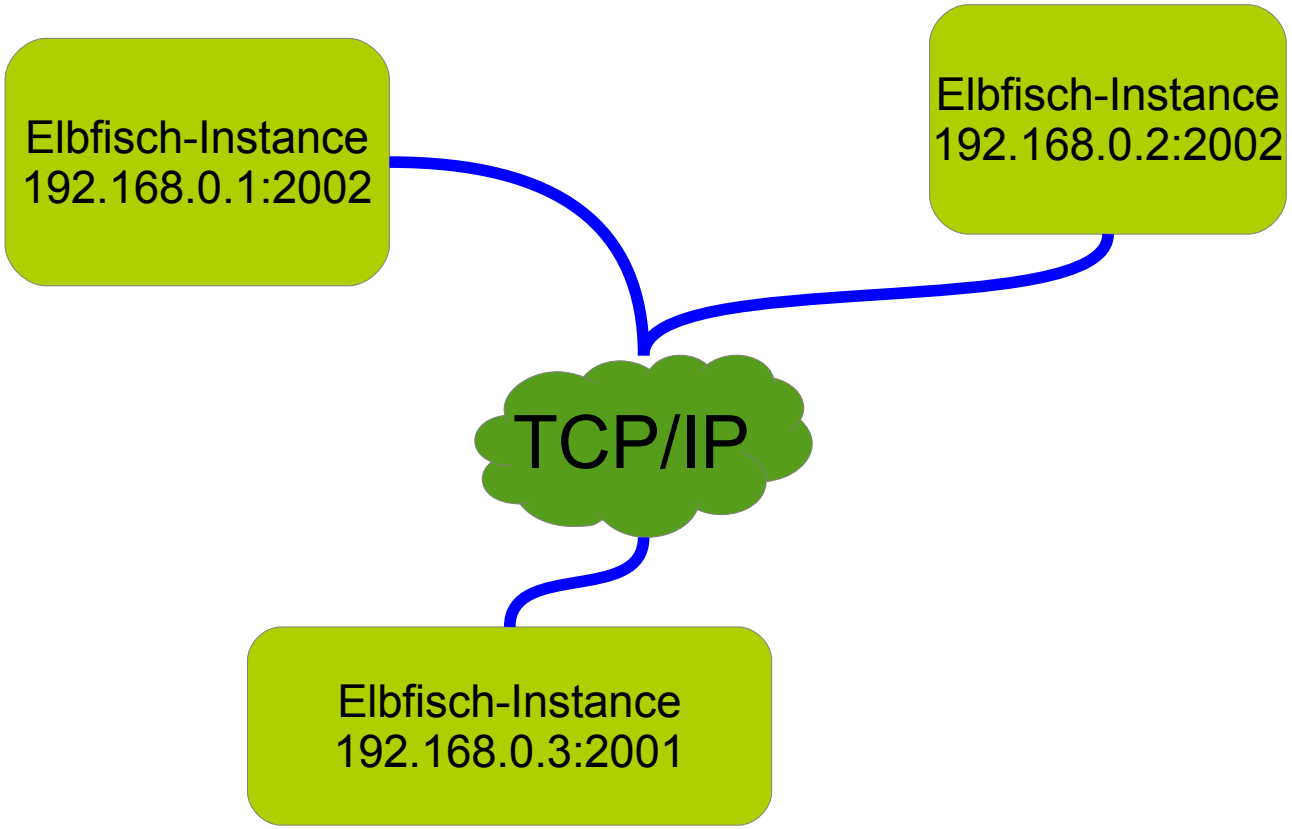
ELBFISCH

Module – Dynamics of a concrete application:



Bernd Schuster, MSK Gesellschaft für Automatisierung mbH,

Multiple Instances – Distributed applications:

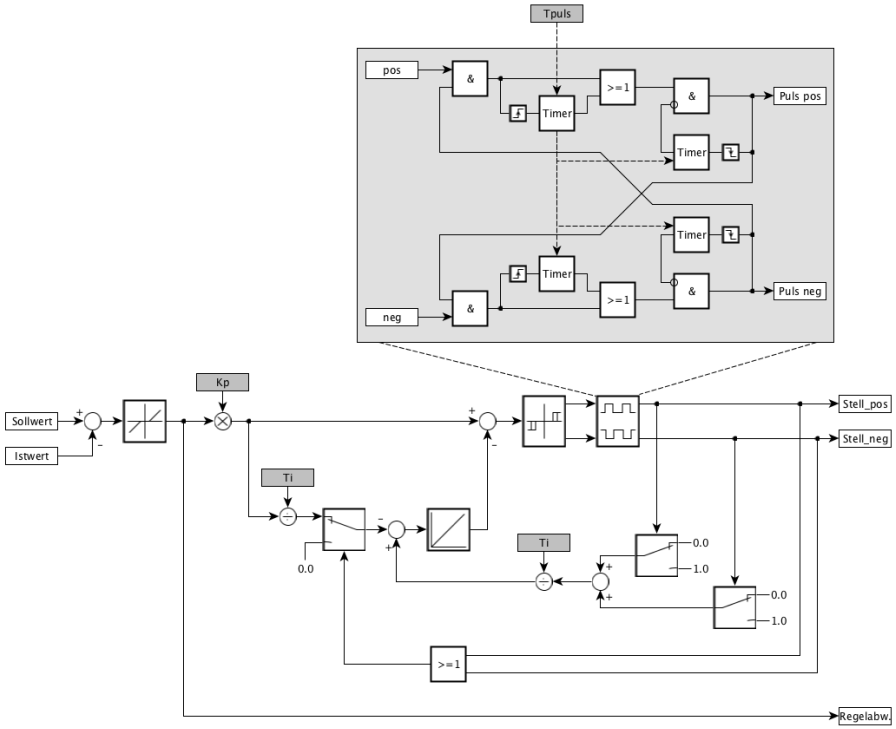


Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

ELBFISCH

Some concrete applications: industrial oven



Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

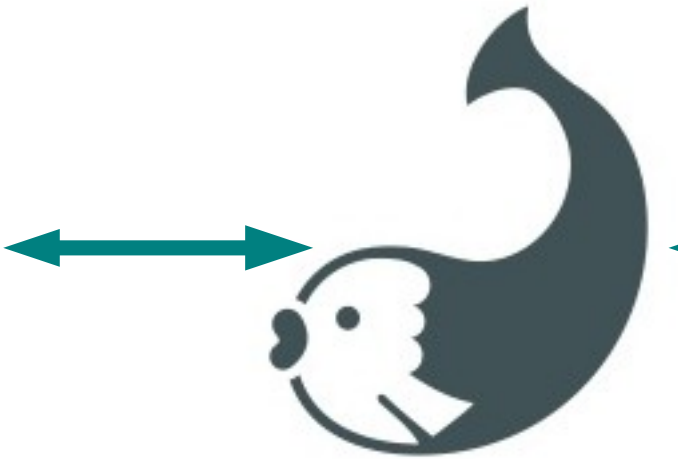
Some concrete applications: industrial oven

Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

ELBFISCH

Some concrete applications: Aerospace QS



Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld

Opensource- Laufzeitsystem für die
komponentenbasierte Implementierung
von Automatisierungslösungen mit Java

ELBFISCH

www.elbfisch.org

Bernd Schuster, MSK Gesellschaft für Automatisierung mbH, Schenefeld