SmartSoft MDSD Toolchain

15th November 2010 / SIMPAR Workshop, Darmstadt

Andreas Steck and Christian Schlegel

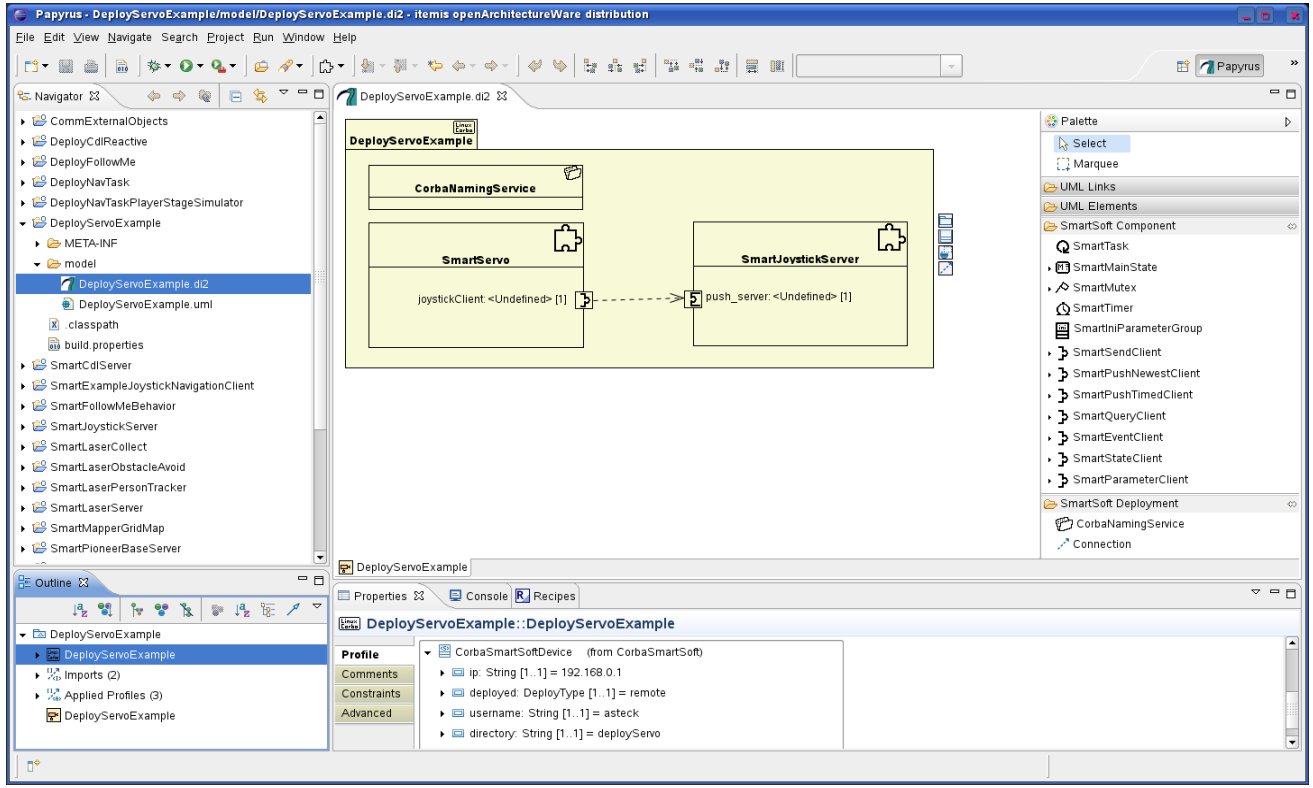
Computer Science Department
University of Applied Sciences Ulm

http://smart-robotics.sourceforge.net/
http://www.zafh-servicerobotik.de/ULM/index.php

SmartSoft MDSD Toolchain

- Example 1: Servo (RTAI-Linux)
- Example 2: Navigation Task
- Example 3: “Follow Me” - RoboCup@Home
- Example 4: Cleanup Table Scenario

SmartSoft MDSD Toolchain

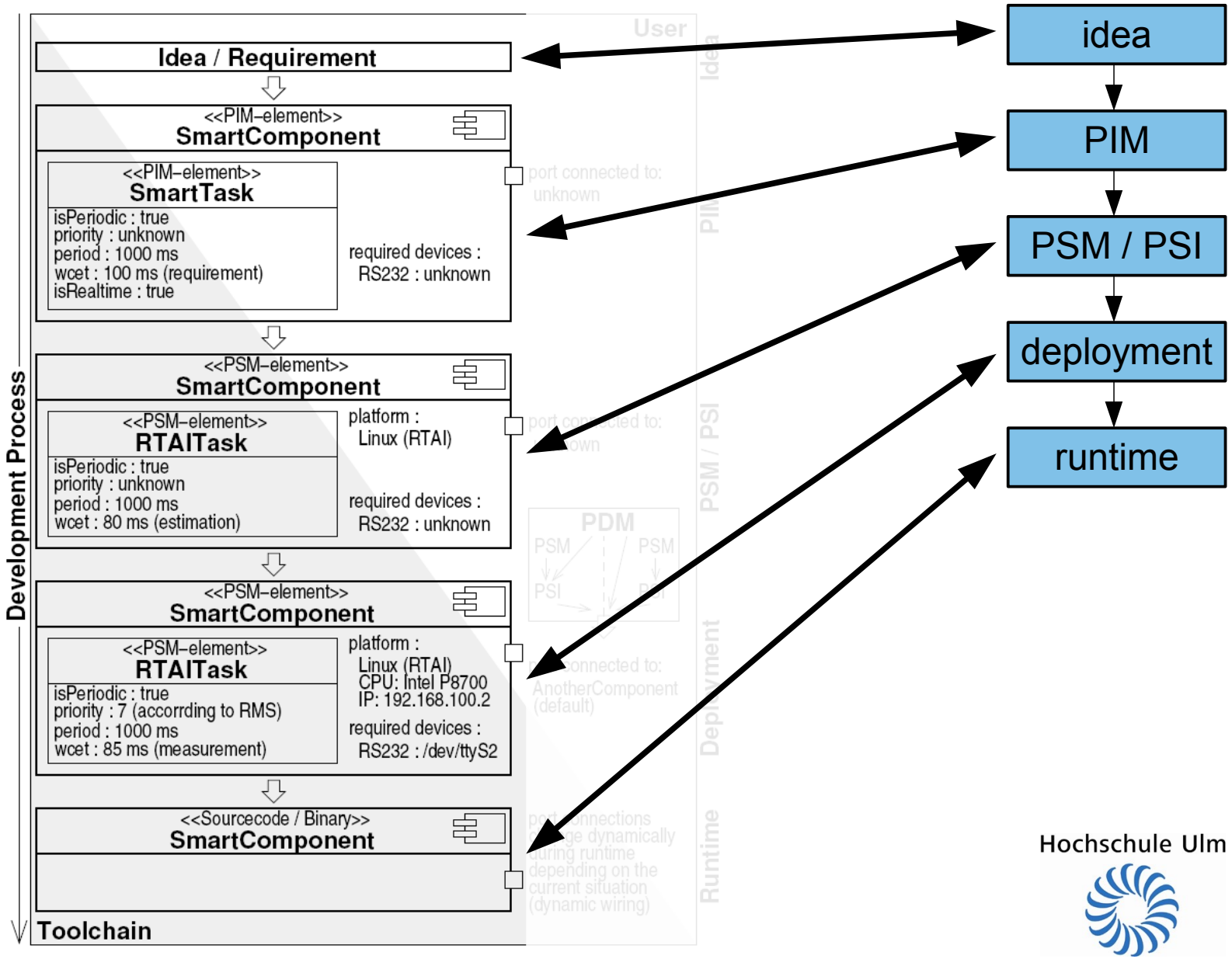


- Eclipse

- openArchitectureWare

- PapyrusUML

Hochschule Ulm

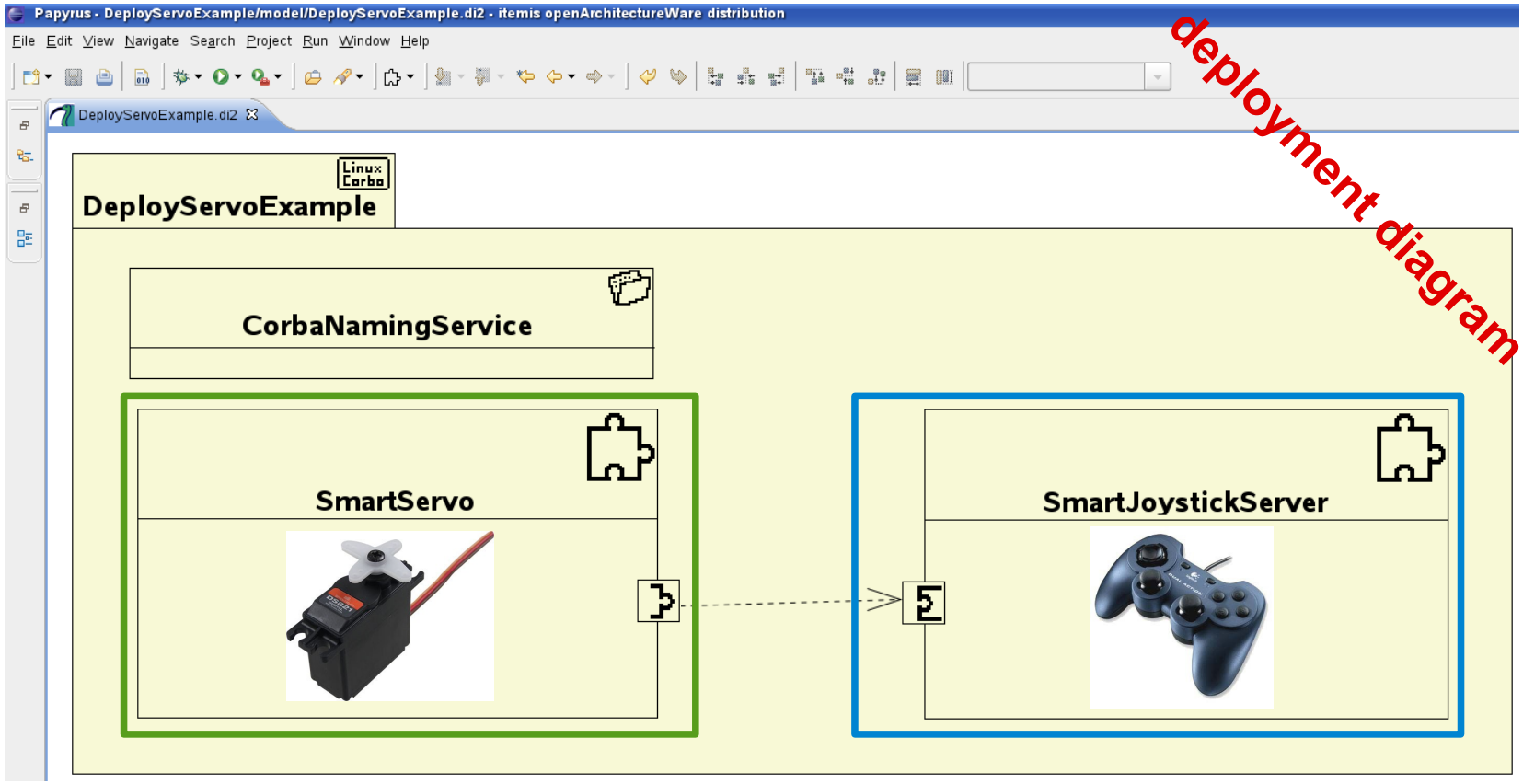

Development Process
Idea / Requirement
<<PIM-element>>
SmartComponent
<<PIM-element>>
SmartTask
isPeriodic : true
priority : unknown
period : 1000 ms
wcet : 100 ms (requirement)
isRealtime : true
required devices :
RS232 : unknown
<<PSM-element>>
SmartComponent
<<PSM-element>>
RTAITask
isPeriodic : true
priority : unknown
period : 1000 ms
wcet : 80 ms (estimation)
platform :
Linux (RTAI)
required devices :
RS232 : unknown
<<PSM-element>>
SmartComponent
<<PSM-element>>
RTAITask
isPeriodic : true
priority : 7 (accorrding to RMS)
period : 1000 ms
wcet : 85 ms (measurement)
platform :
Linux (RTAI)
CPU: Intel P8700
IP: 192.168.100.2
required devices :
RS232 : /dev/ttyS2
<<Sourcecode / Binary>>
SmartComponent
Toolchain
idea
PIM
PSM / PSI
deployment
runtime

SmartSoft MDSD Toolchain
Example 1: Servo (RTAI-Linux)

- introduction into overall MDSD process via simple example
- hard realtime in servo component

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)



deployment diagram

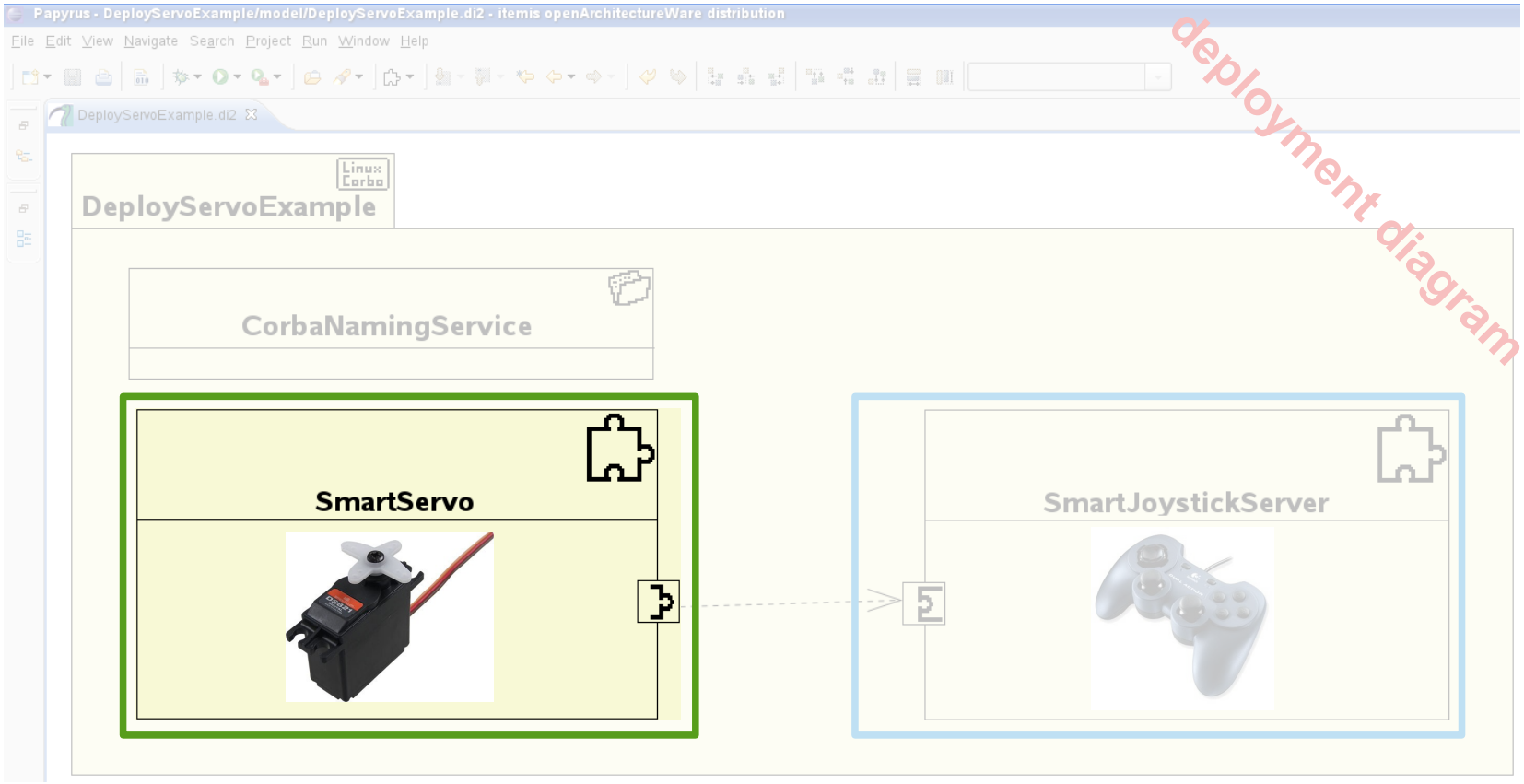
- idea
- PIM
- PSM / PSI
- deployment
- runtime

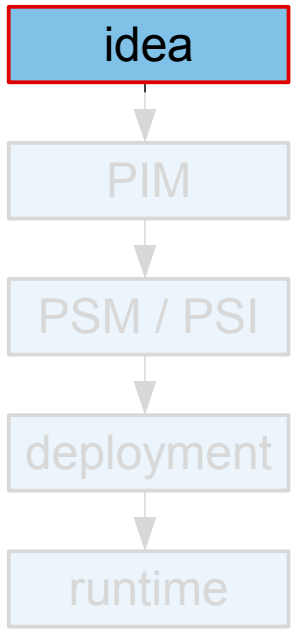
SmartServo component
- will be created in the following demo

SmartJoystickServer component
- preexisting component (COTS)

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)





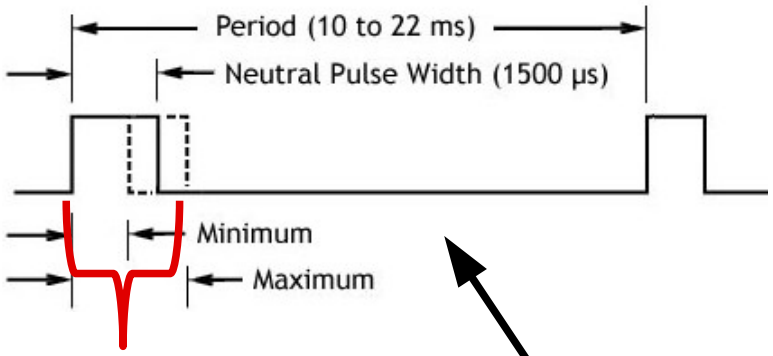
SmartServo component
- will be created in the following demo

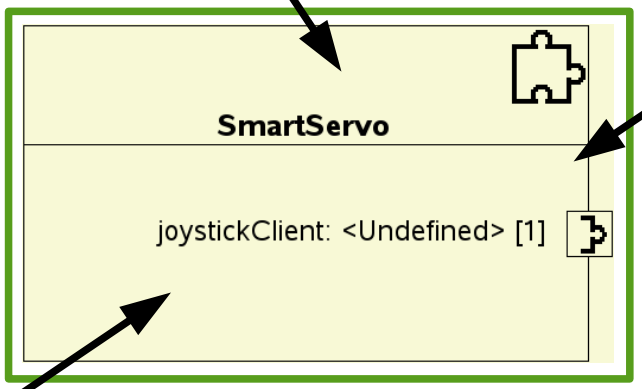
SmartJoystickServer component
- preexisting component (COTS)

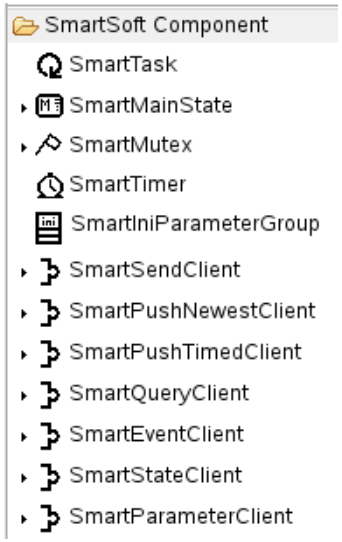
SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

creating a component idea



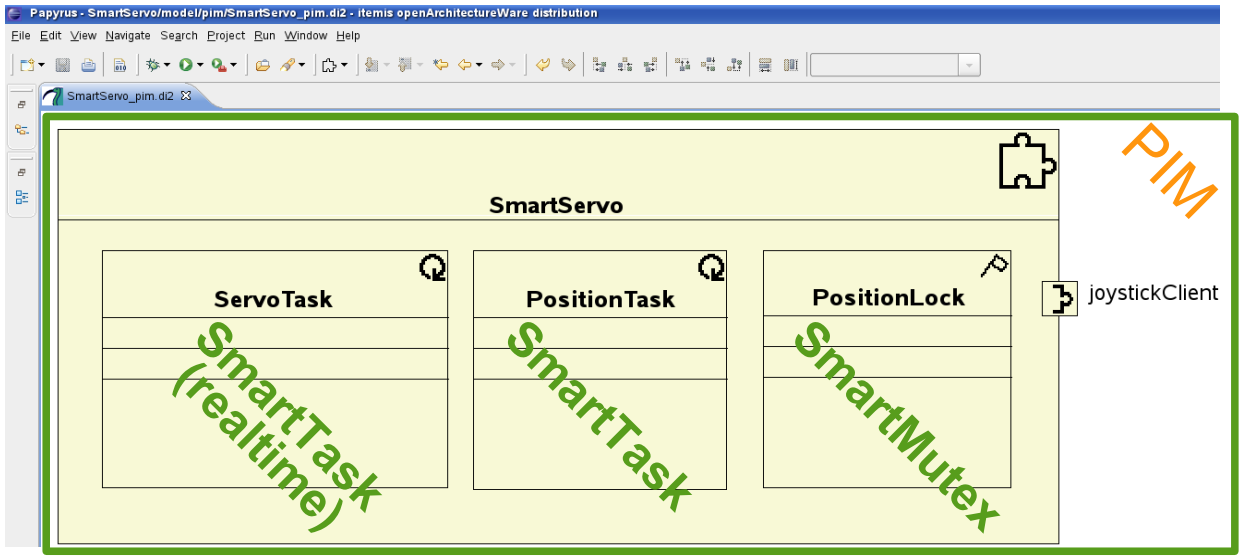


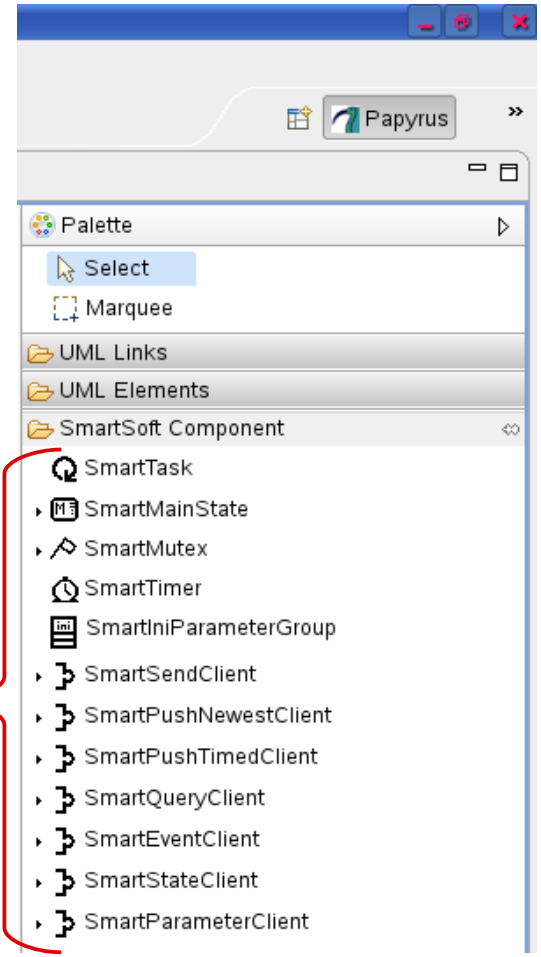


idea
PIM
PSM / PSI
deployment
runtime

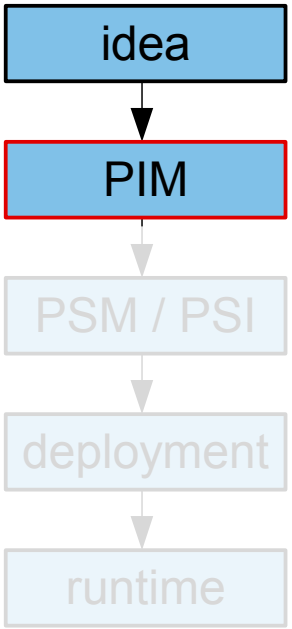
SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)



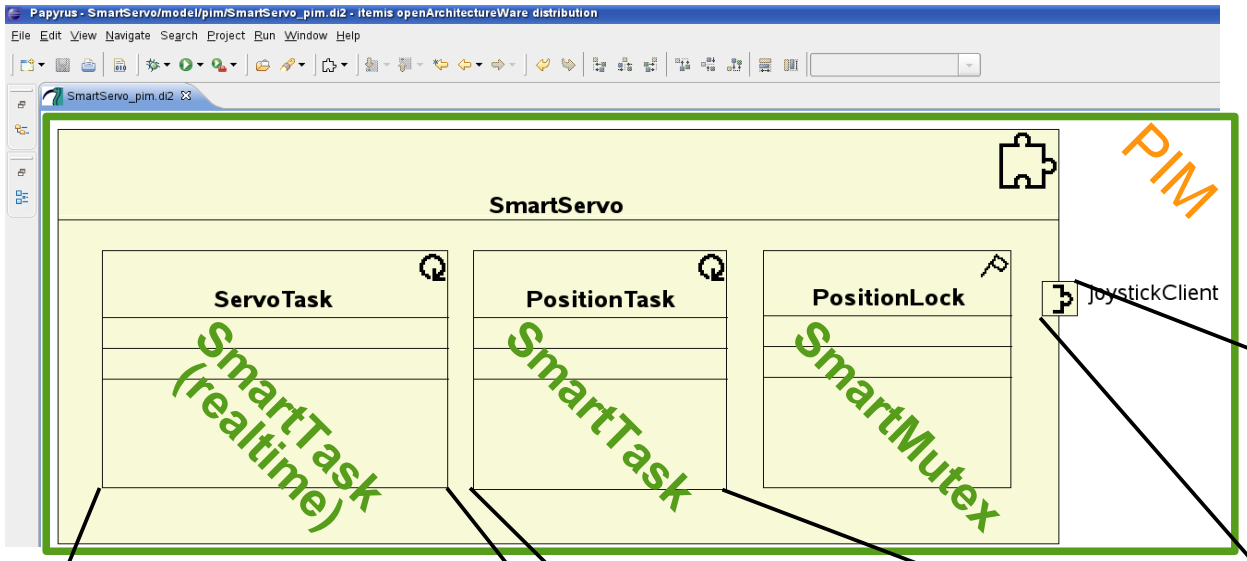


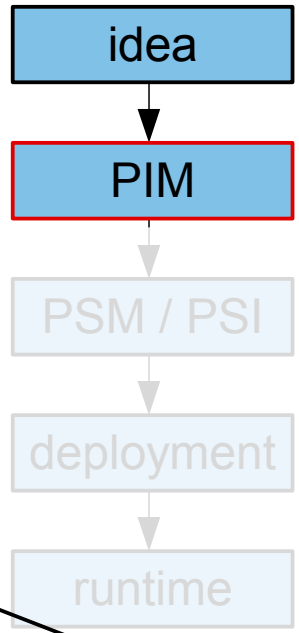
create PIM using the elements offered by the SmartSoft MDSD Toolchain palette



SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)





control servo

calculate servo position

port connected to smartJoystickServer

- SmartTask (from SmartMARS)
  - isRealtime: Boolean [1..1] = true
  - period: Integer [1..1] = 20
  - priority: Integer [1..1] = 1
  - isPeriodic: Boolean [1..1] = true
  - wcet: Integer [1..1] = 1
  - timeUnit: TimeUnitKind [1..1] = ms

- SmartTask (from SmartMARS)
  - isRealtime: Boolean [1..1] = false
  - period: Integer [1..1] = 0
  - priority: Integer [1..1] = 0
  - isPeriodic: Boolean [1..1] = false
  - wcet: Integer [1..1] = 0
  - timeUnit: TimeUnitKind [1..1] = s

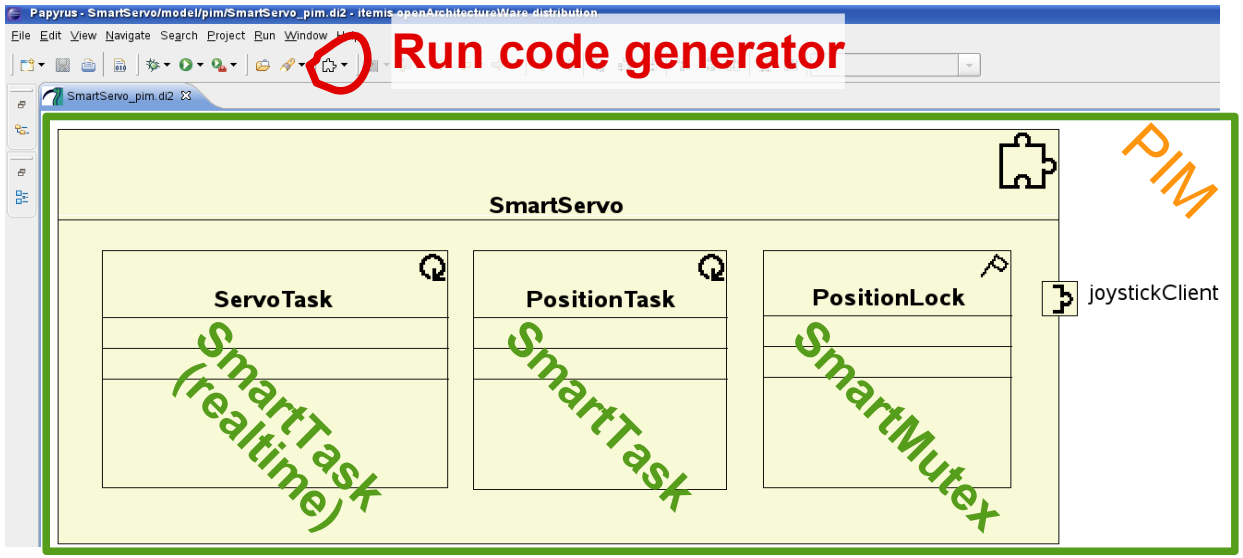
- SmartPushNewestClient (from SmartMARS)
  - serverName: String [1..1] = smartJoystickServer
  - serviceName: String [1..1] = joystick
  - commObject: Class [1..1] = CommJoystick

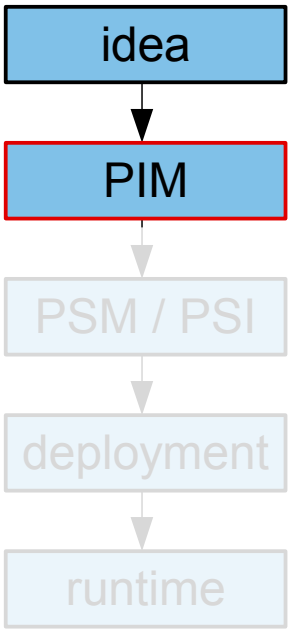
tagged values

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

creating a component generate PSI

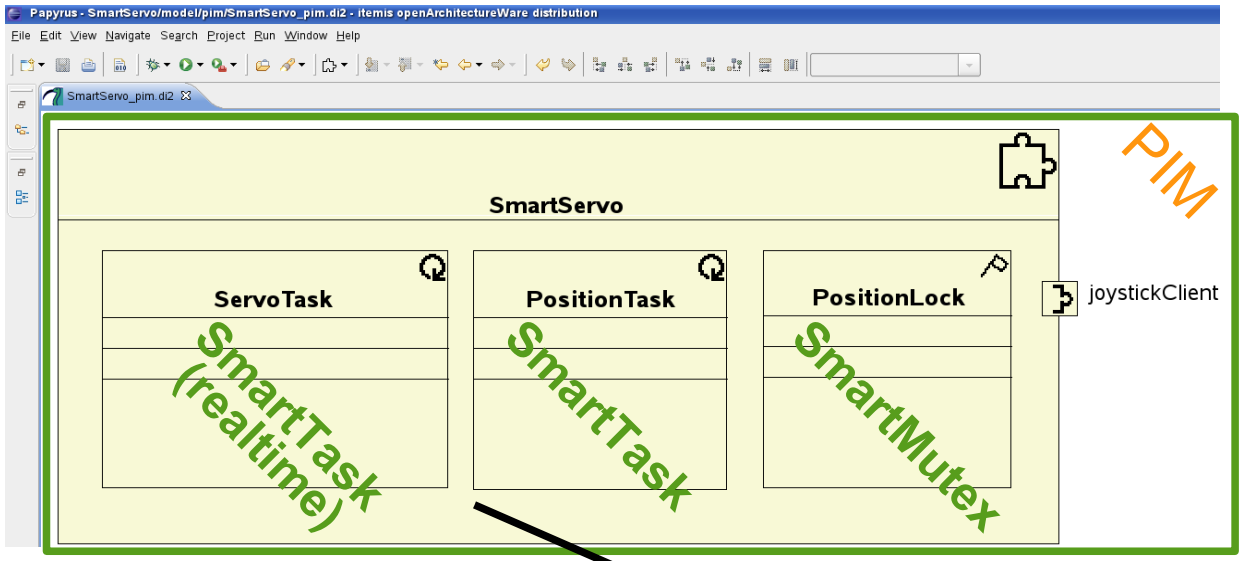


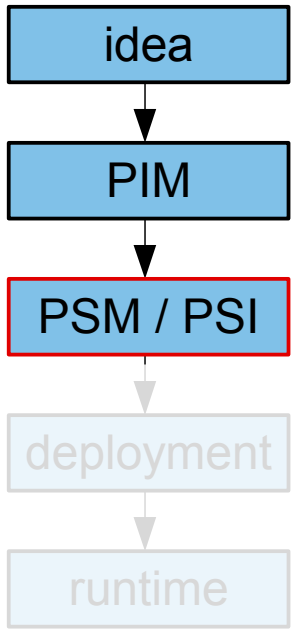


SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

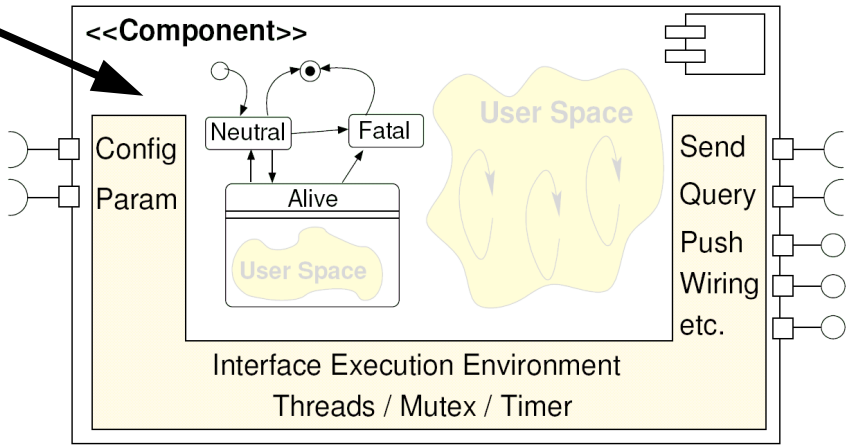
creating a component generate PSI





verification (e.g. QoS)+ transformation (is done by toolchain)

executable component

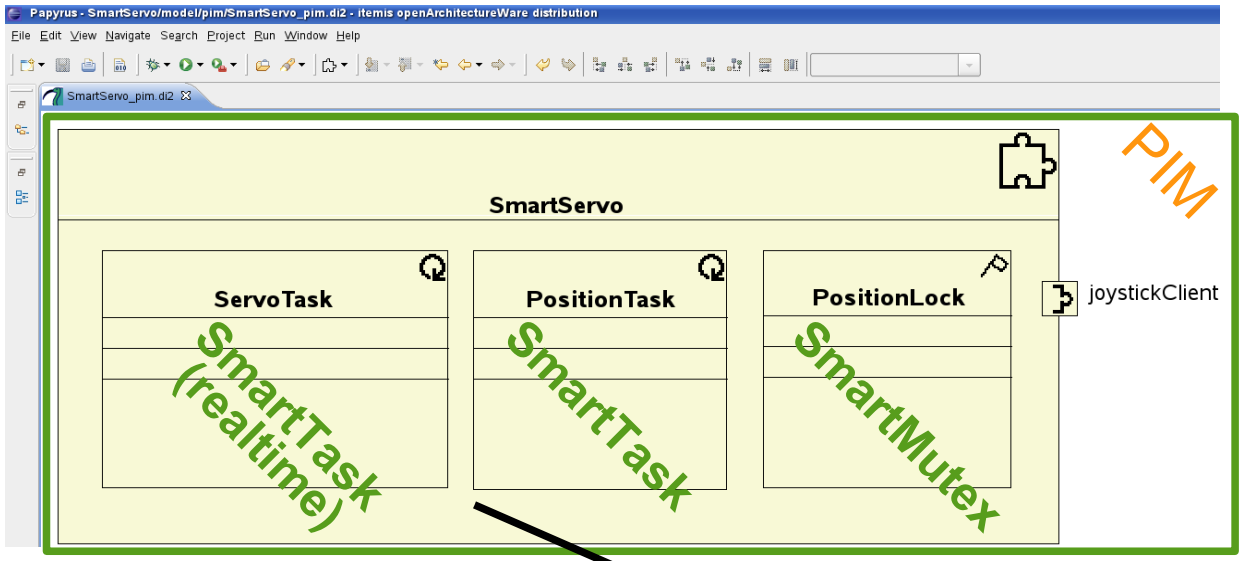


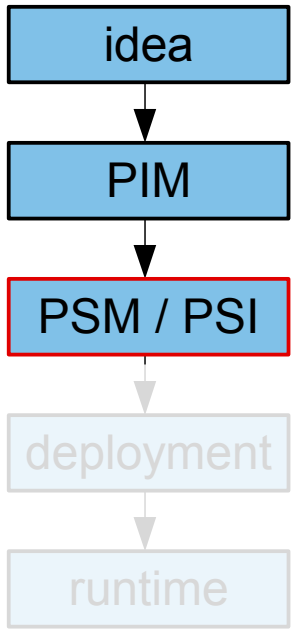
service robotik

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

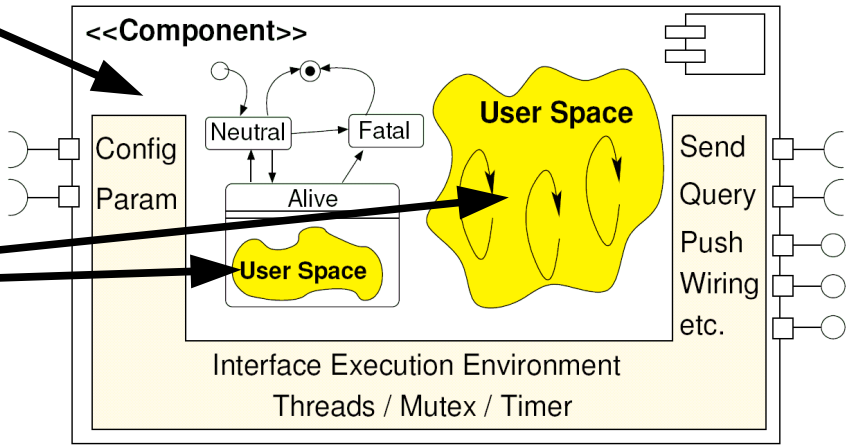
creating a component integrate user-code





verification (e.g. QoS)+ transformation
(is done by toolchain)

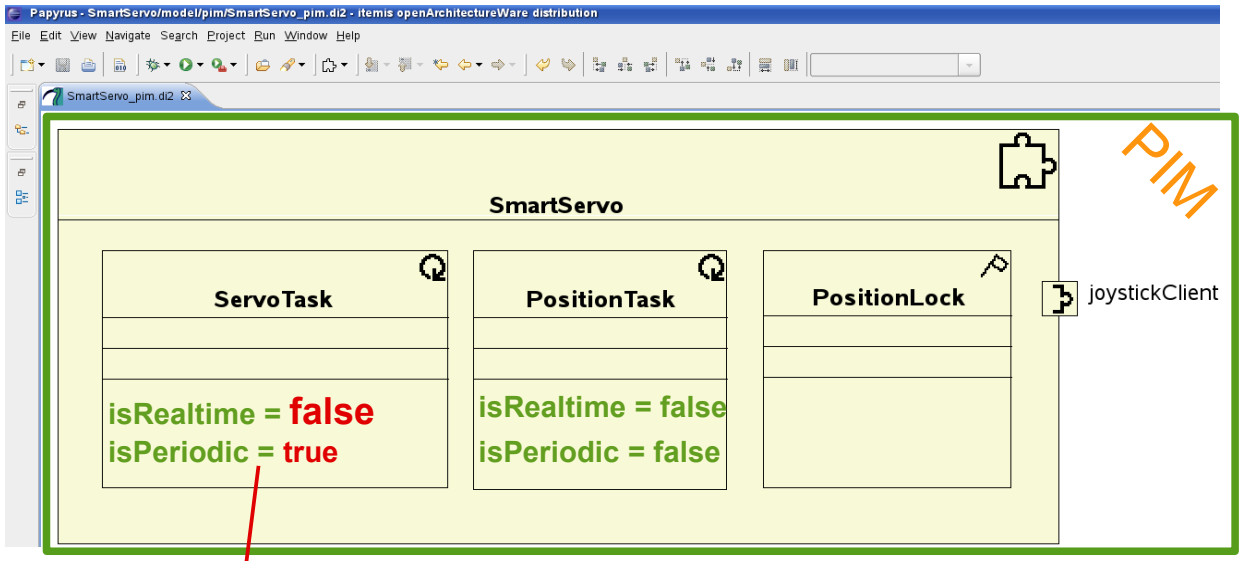
executable component



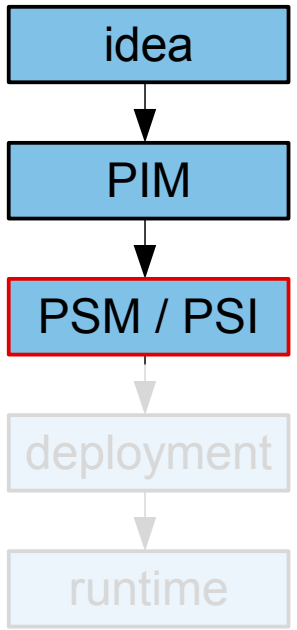
User Code
MATLAB / Simulink
RTAI-Lab
OpenCV / Qt / Kavraki-Lab

SmartSoft MDSD Toolchain
Example 1: Servo (RTAI-Linux)

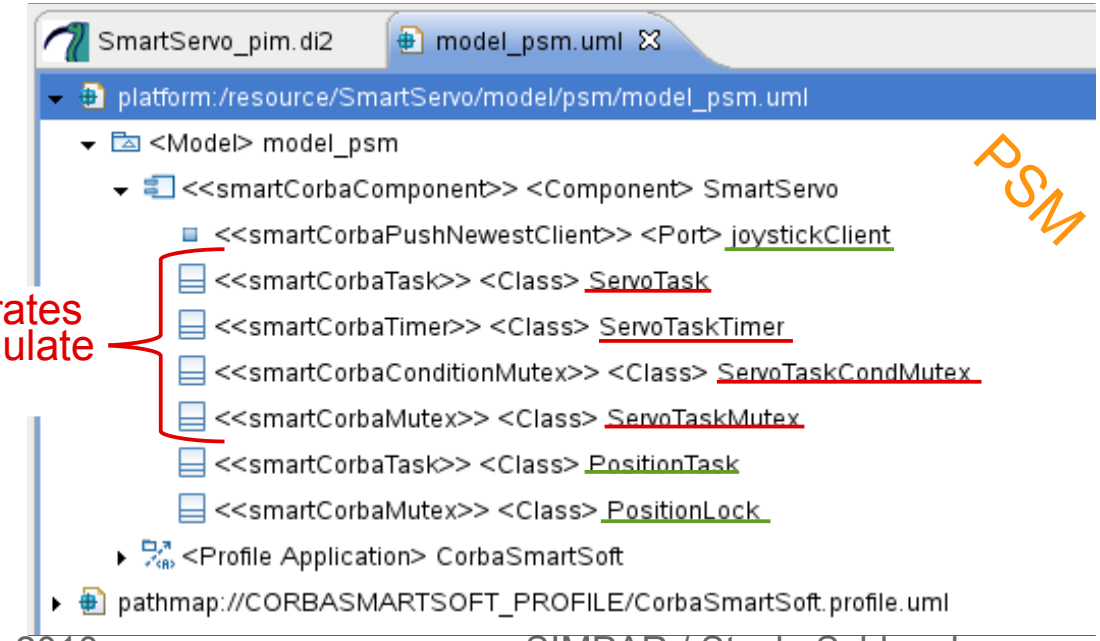
creating a component generate PSM

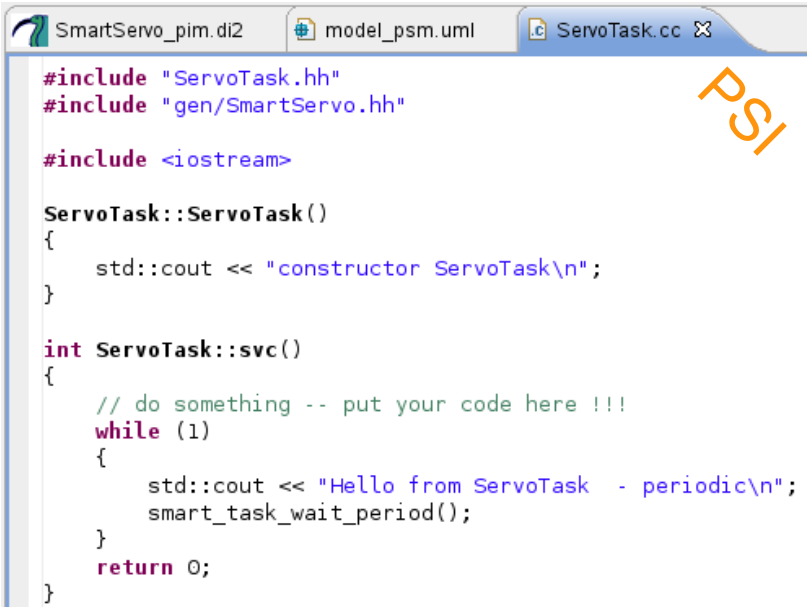


mapping
PIM elements
to PSM elements
(is done by toolchain)



toolchain generates elements to emulate periodic tasks



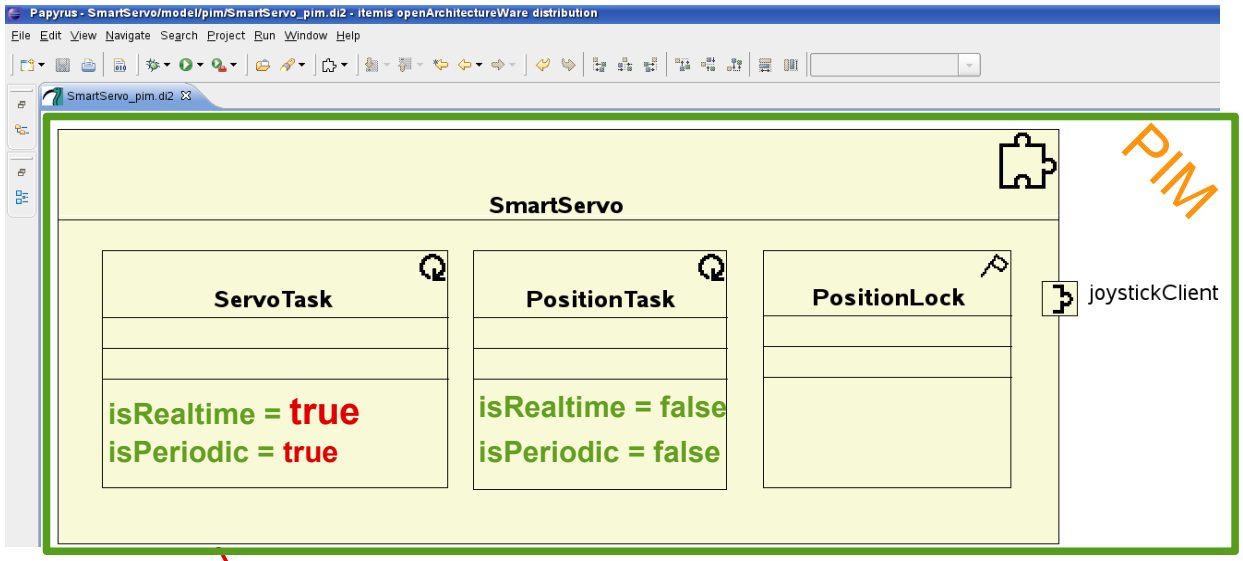


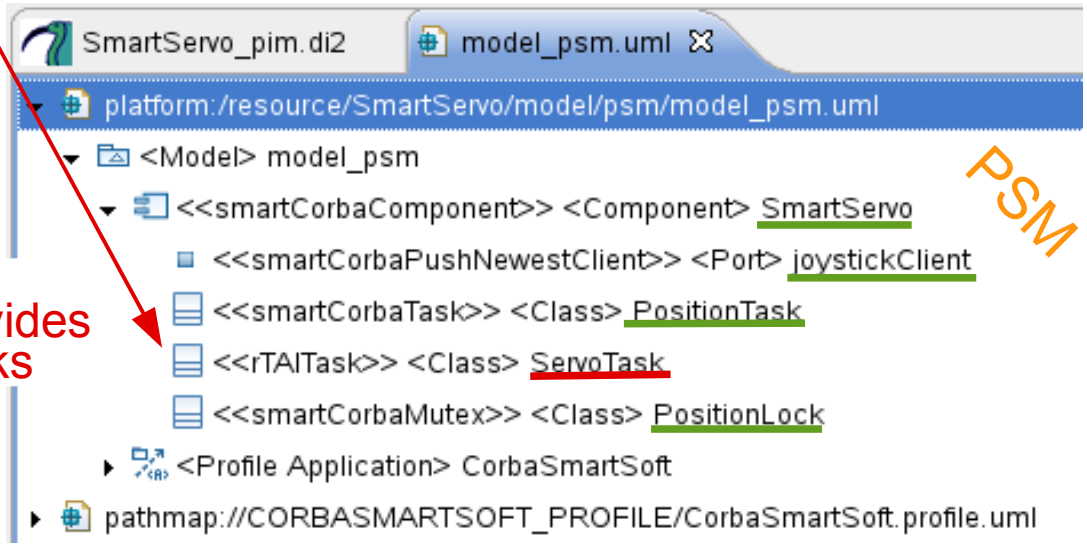
service robotik

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

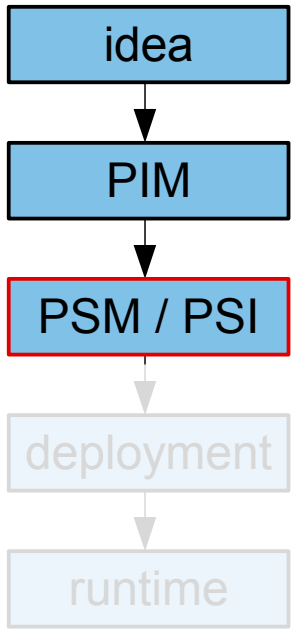
creating a component generate PSM

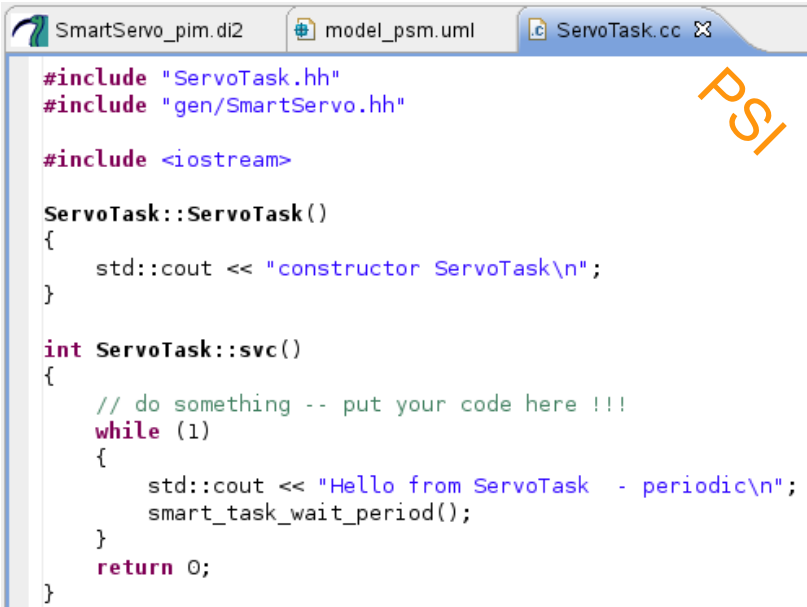




RTAI-Linux already provides periodic tasks

mapping PIM elements to PSM elements (is done by toolchain)



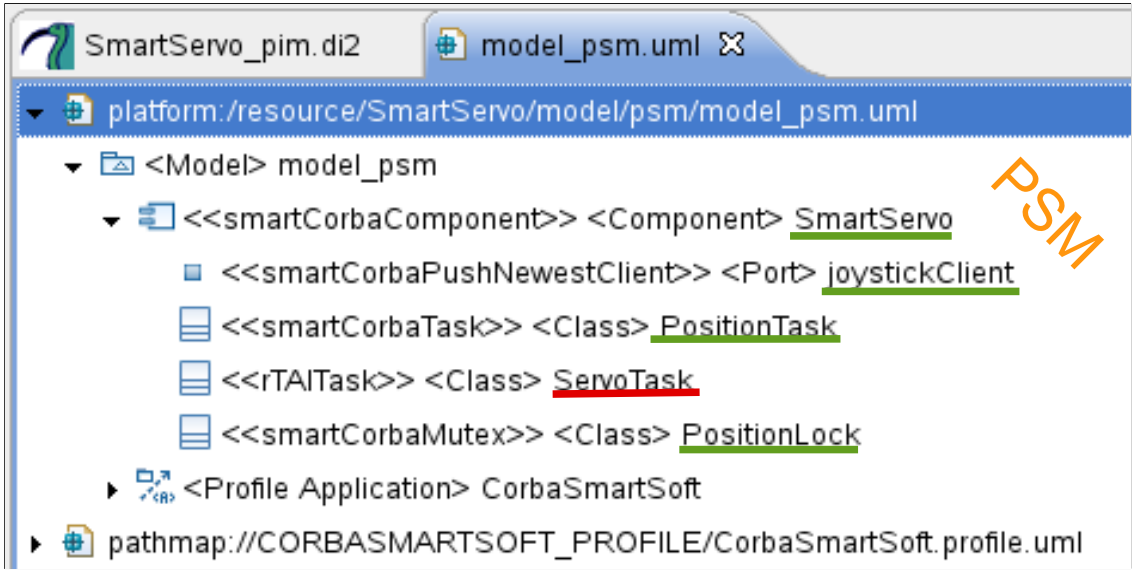


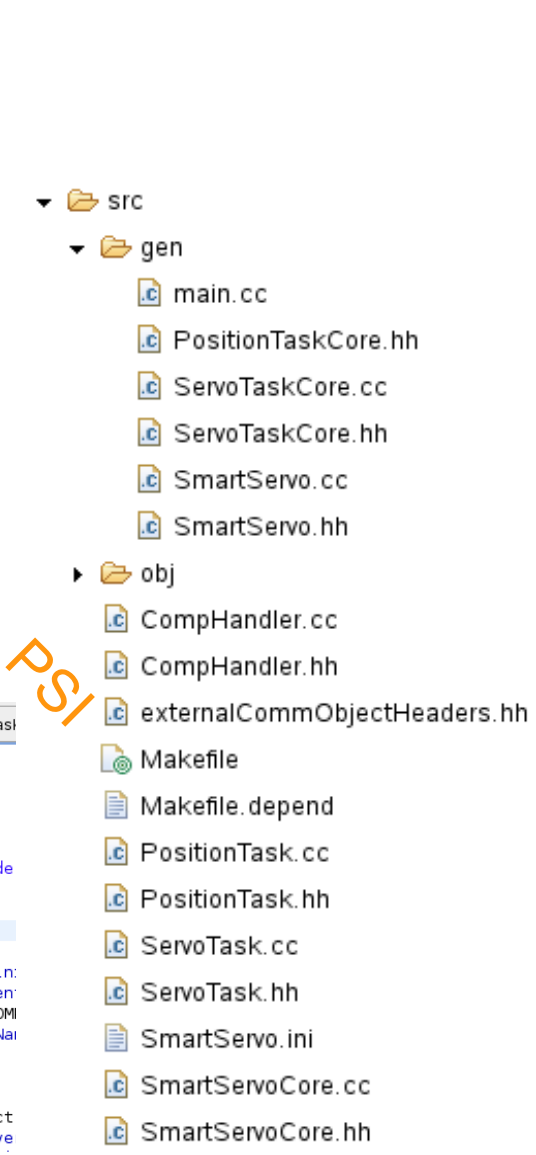

```
#include "ServoTask.hh"
#include "gen/SmartServo.hh"

#include <iostream>

ServoTask::ServoTask()
{
    std::cout << "constructor ServoTask\n";
}

int ServoTask::svc()
{
    // do something -- put your code here !!!
    while (1)
    {
        std::cout << "Hello from ServoTask  - periodic\n";
        smart_task_wait_period();
    }
    return 0;
}
```

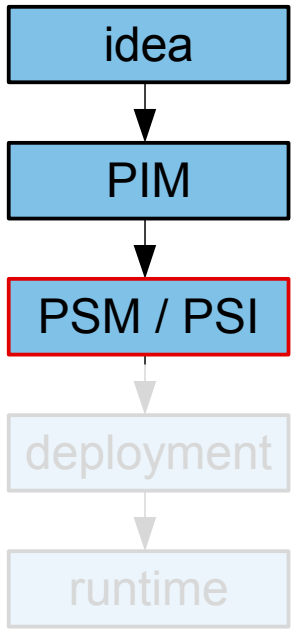
SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

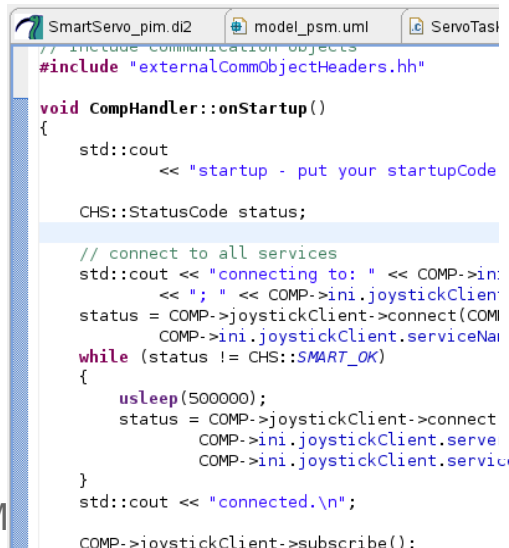
creating a component generate PSI







generate PSI
out of
PSM elements
(is done by toolchain)



Hochschule Ulm

SmartSoft MDSD Toolchain
Example 1: Servo (RTAI-Linux)

creating a component integrate user-code

SmartServo_pim.di2 | model_psm.uml | PositionTask.cc | ServoTask.cc | SmartServoCore.cc

```
#include "SmartServoCore.hh"

// constructor
SmartServoCore::SmartServoCore()
{
    std::cout << "constructor SmartServoCore\n";
    servoOnTime = 1500000;
}
```

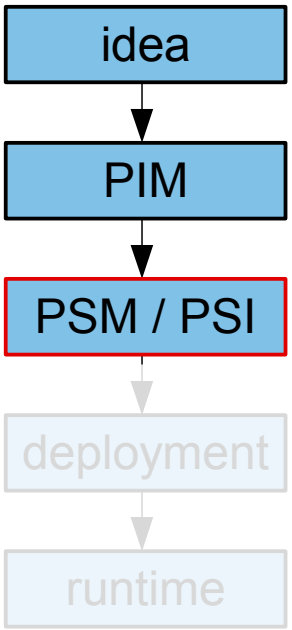
code added by developer

SmartServoCore.hh

```
#ifndef _SMARTSERVOCORE_HH
#define _SMARTSERVOCORE_HH

#include <iostream>

class SmartServoCore
{
private:

public:
    SmartServoCore();

    int servoOnTime;
};

#endif
```

code added by developer

PSI



Hochschule Ulm

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

creating a component integrate user-code

SmartServo_pim.di2 | model_psm.uml | PositionTask.cc | ServoTask.cc

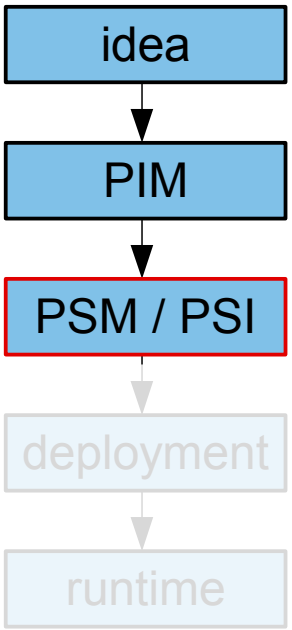
```
#include "PositionTask.hh"
#include "gen/SmartServo.hh"

#include <iostream>

PositionTask::PositionTask()
{
    std::cout << "constructor PositionTask\n";
}

int PositionTask::svc()
{
    // do something -- put your code here !!!
    while (1)
    {
        CHS::StatusCode status;
        Smart::CommJoystick cmd;
        status = COMP->joystickClient->getUpdateWait(cmd);
        if(status == CHS::SMART_OK)
        {
            double x = cmd.get_x();
            COMP->PositionLock.acquire();
            COMP->servoOnTime = 1500000 + (x * 500000);
            std::cout << "servoOnTime: " << COMP->servoOnTime << std::endl;
            COMP->PositionLock.release();

        }
    }
    return 0;
}
```

wait for position update (joystickServer)

calculate servo position (servoOnTime)

code added by developer

PSI



Hochschule Ulm

SmartSoft MDSD Toolchain
Example 1: Servo (RTAI-Linux)

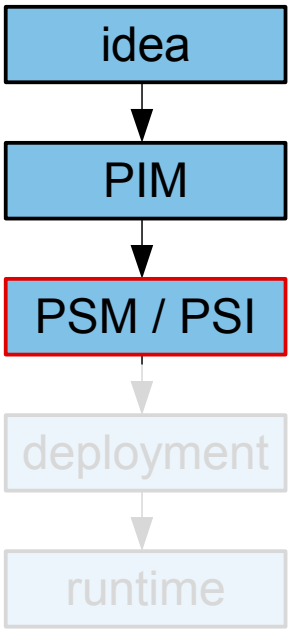
creating a component integrate user-code

SmartServo_pim.di2 | model_psm.uml | PositionTask.cc | ServoTask.cc

```
#include <iostream>
#include <sys/io.h>

#define LPT_PORT 0x378

ServoTask::ServoTask()
{
    std::cout << "constructor ServoTask\n";
}

int ServoTask::svc()
{

    RTIME on_time;
    // servo position
    on_time = nano2count(1500000);

    while (1)
    {
        // ontime
        COMP->PositionLock.acquire();
        on_time = nano2count(COMP->servoOnTime);
        COMP->PositionLock.release();

        // switch on
        outb(0xff, LPT_PORT);
        rt_sleep(on_time);

        // switch off
        outb(0x00, LPT_PORT);
        smart_task_wait_period();
    }
    return 0;
}
```

code added by developer

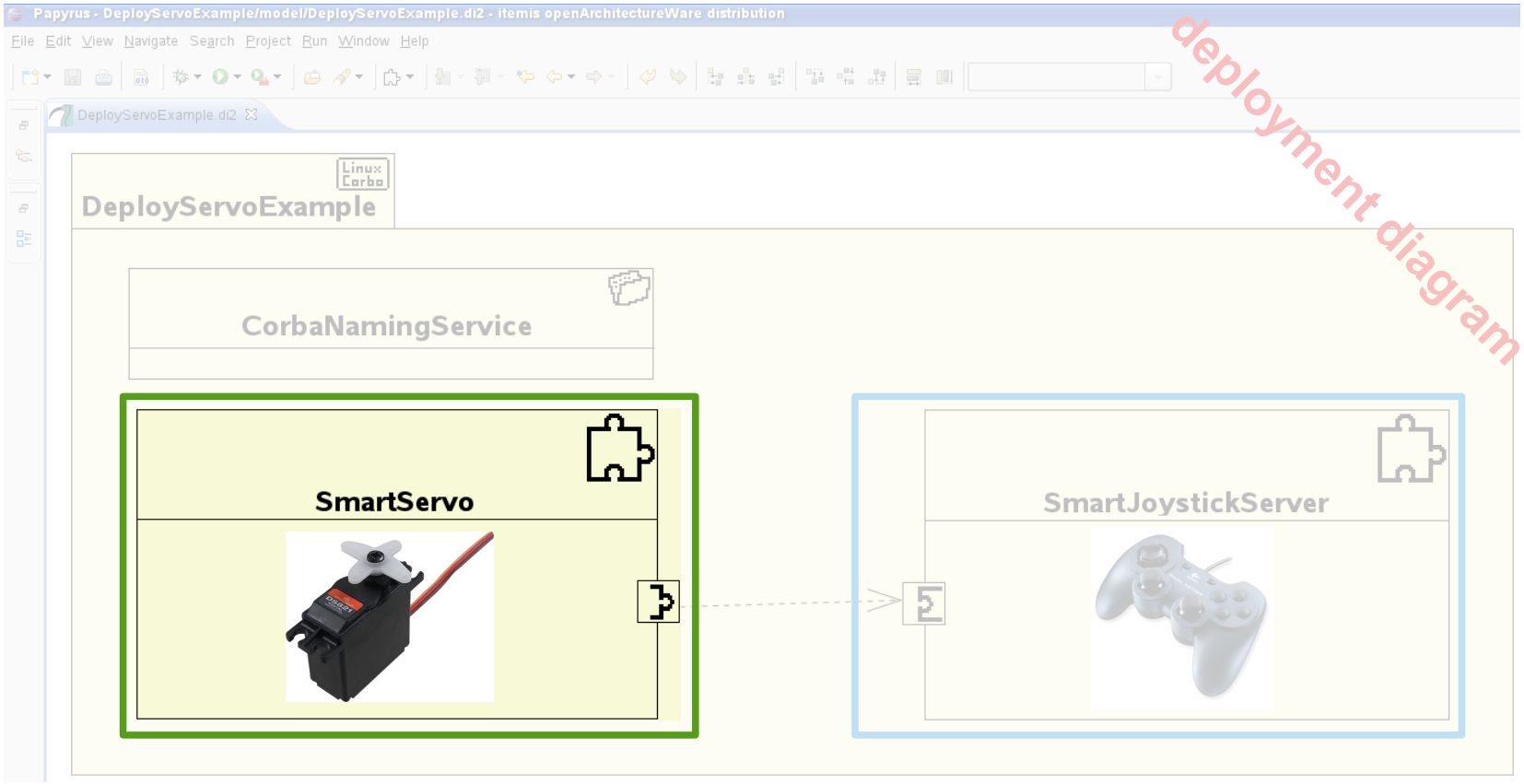
PSI

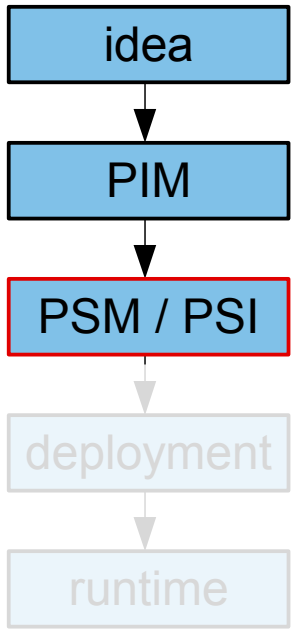


Hochschule Ulm

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)





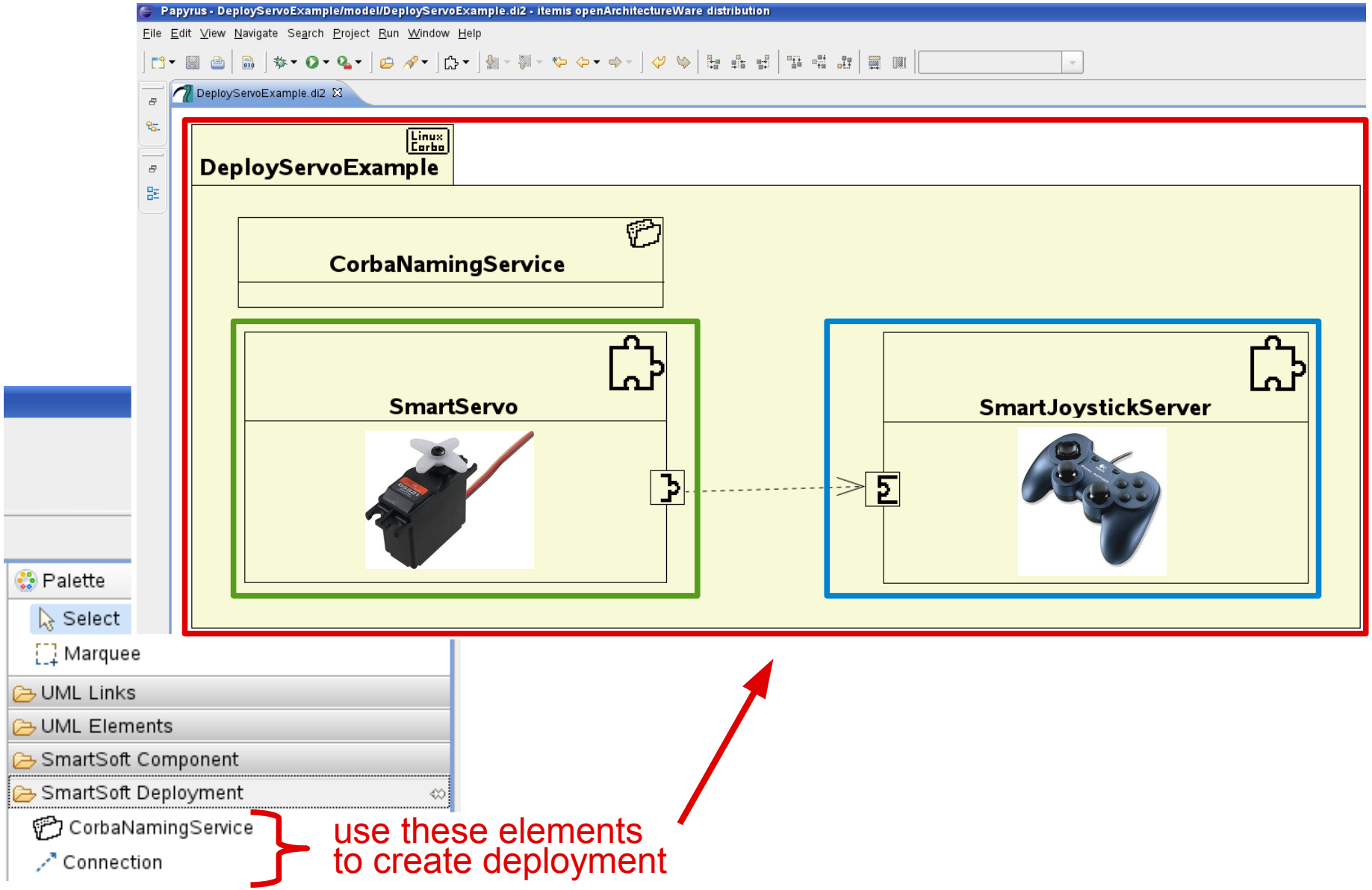
SmartServo component
- created

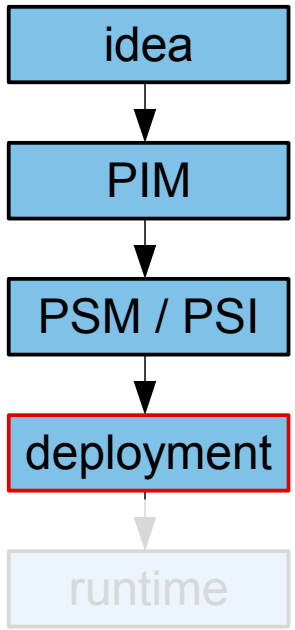
SmartJoystickServer component
- preexisting component (COTS)

SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

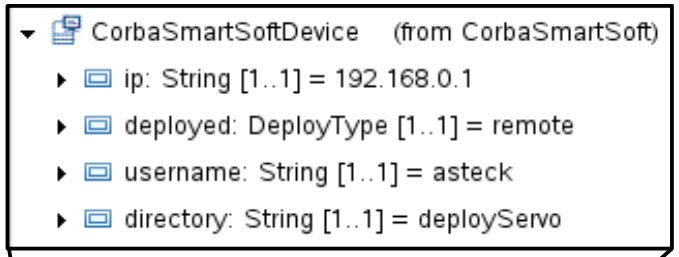
create deployment

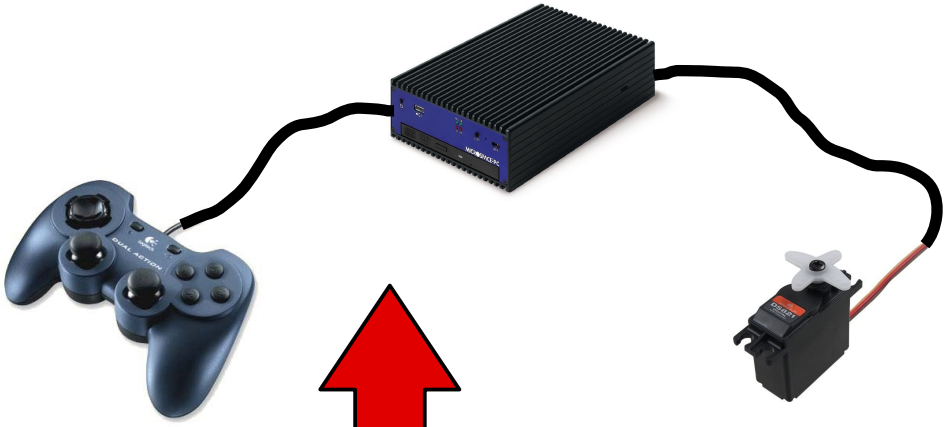


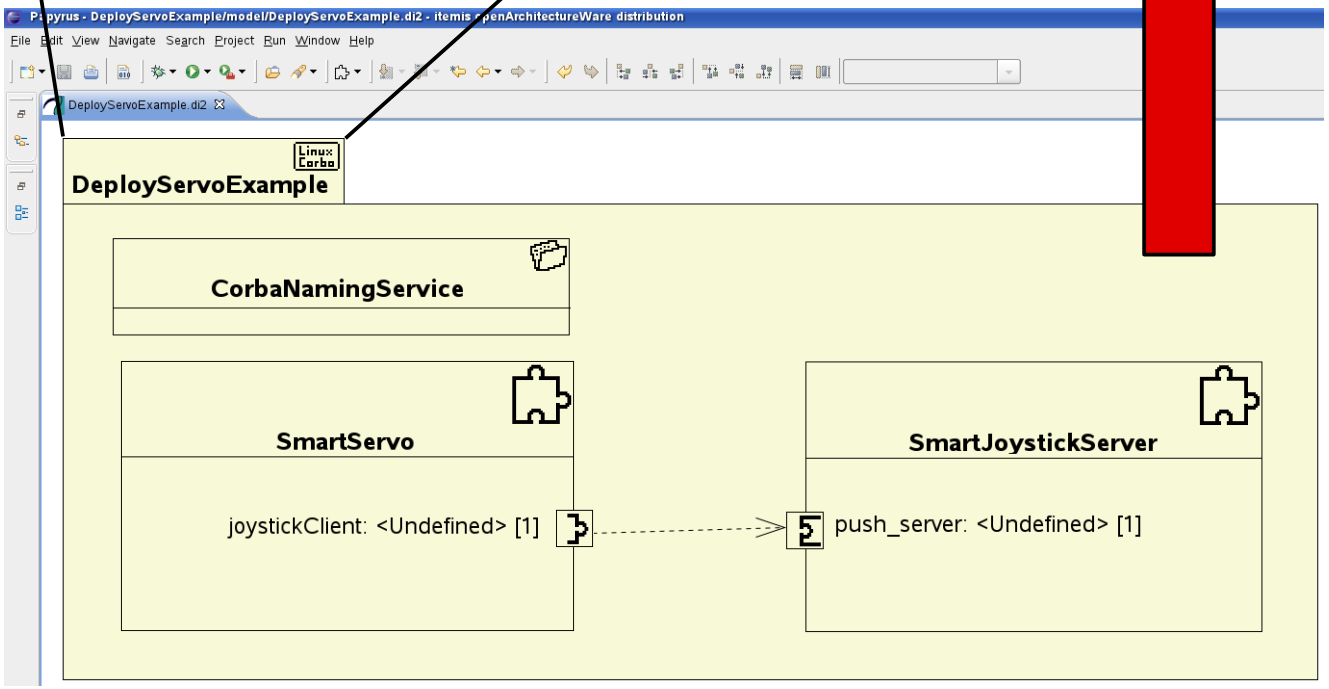


SmartSoft MDSD Toolchain
Example 1: Servo (RTAI-Linux)

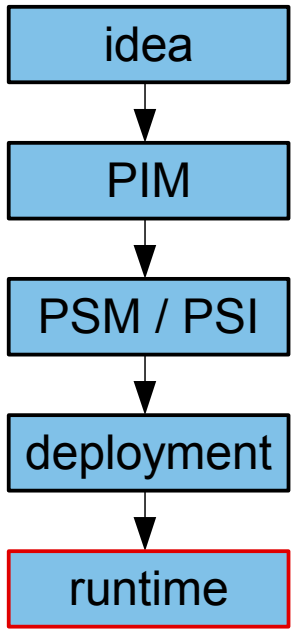
create deployment







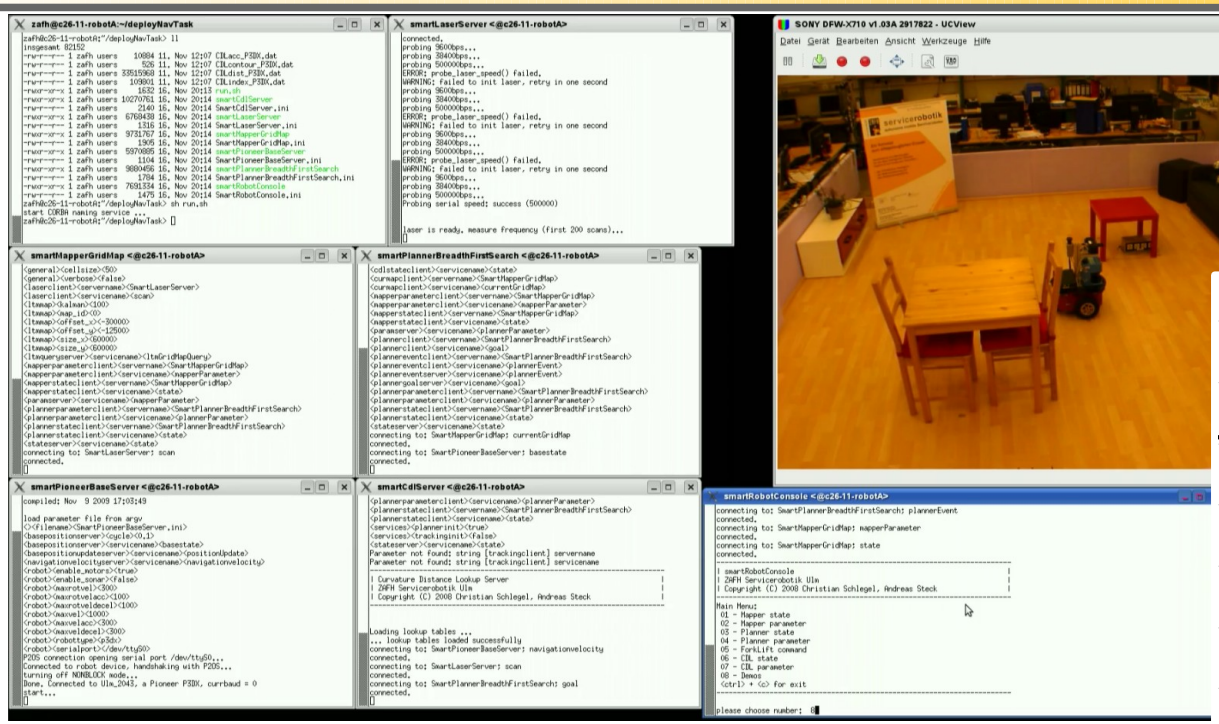
deploy to target system and run startscript

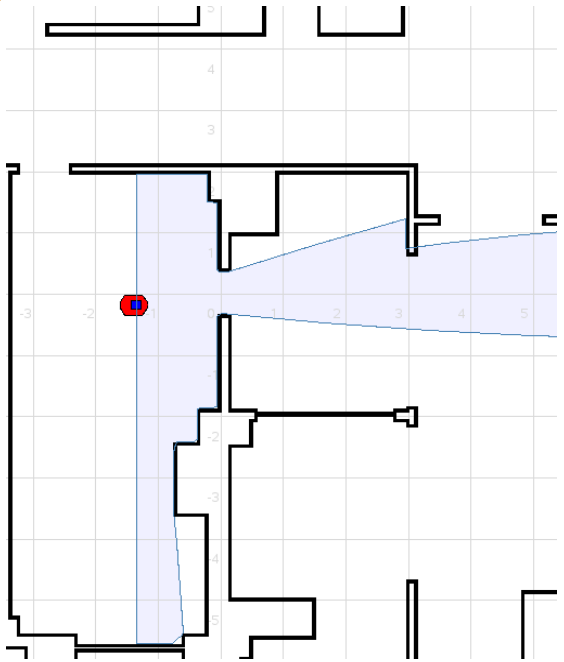


Hochschule Ulm

SmartSoft MDSD Toolchain

Example 2: Navigation Task





- deployment of COTS components
- simple replacement of components (P3DX/SICK ↔ Player/Stage)
- robot drives to positions given by operator
- collision avoidance, map building, path planning, etc.

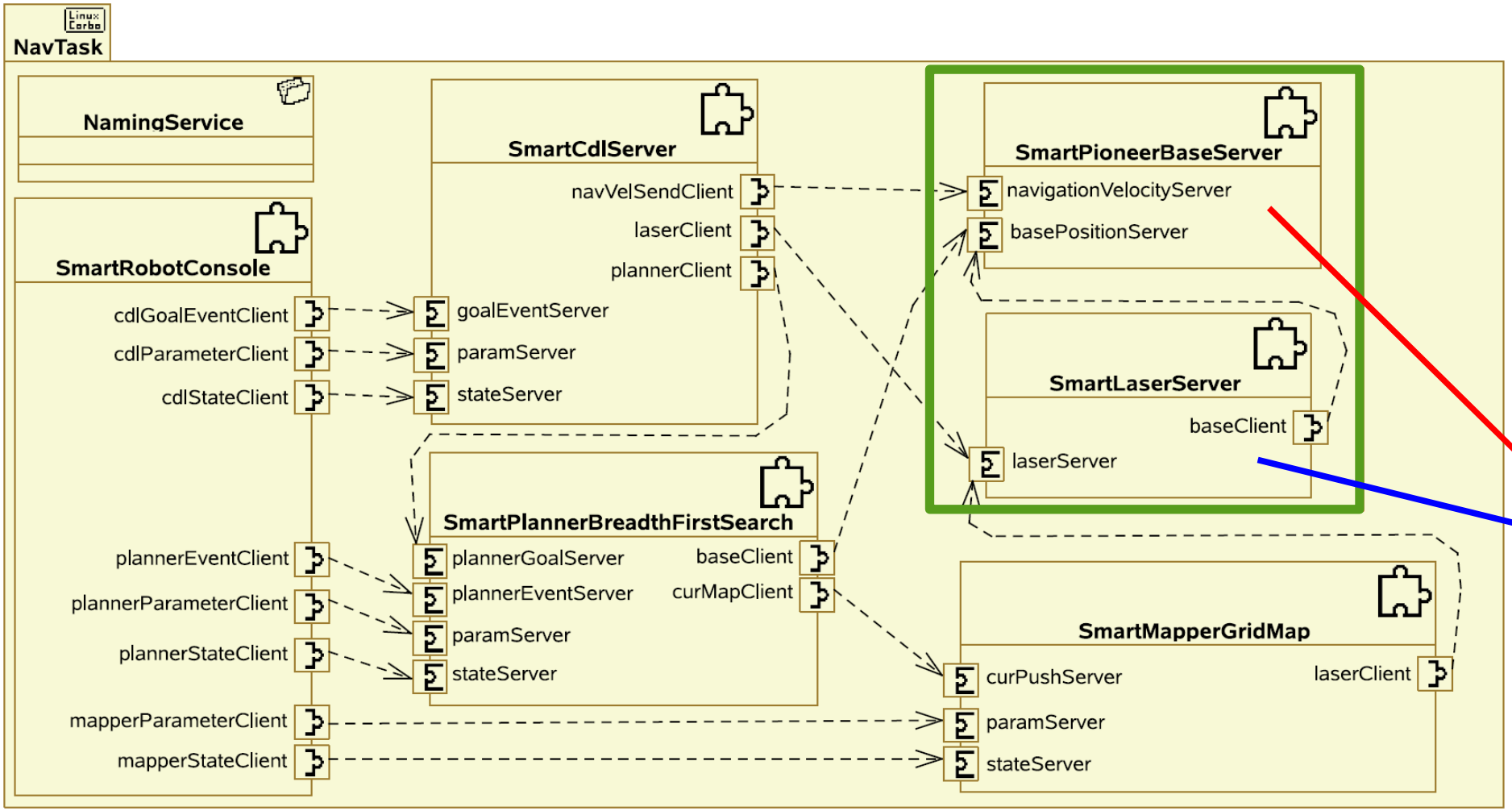
Hochschule Ulm

service robotik

SmartSoft MDSD Toolchain

Example 2: Navigation Task

deployment navigation task P3DX/SICK

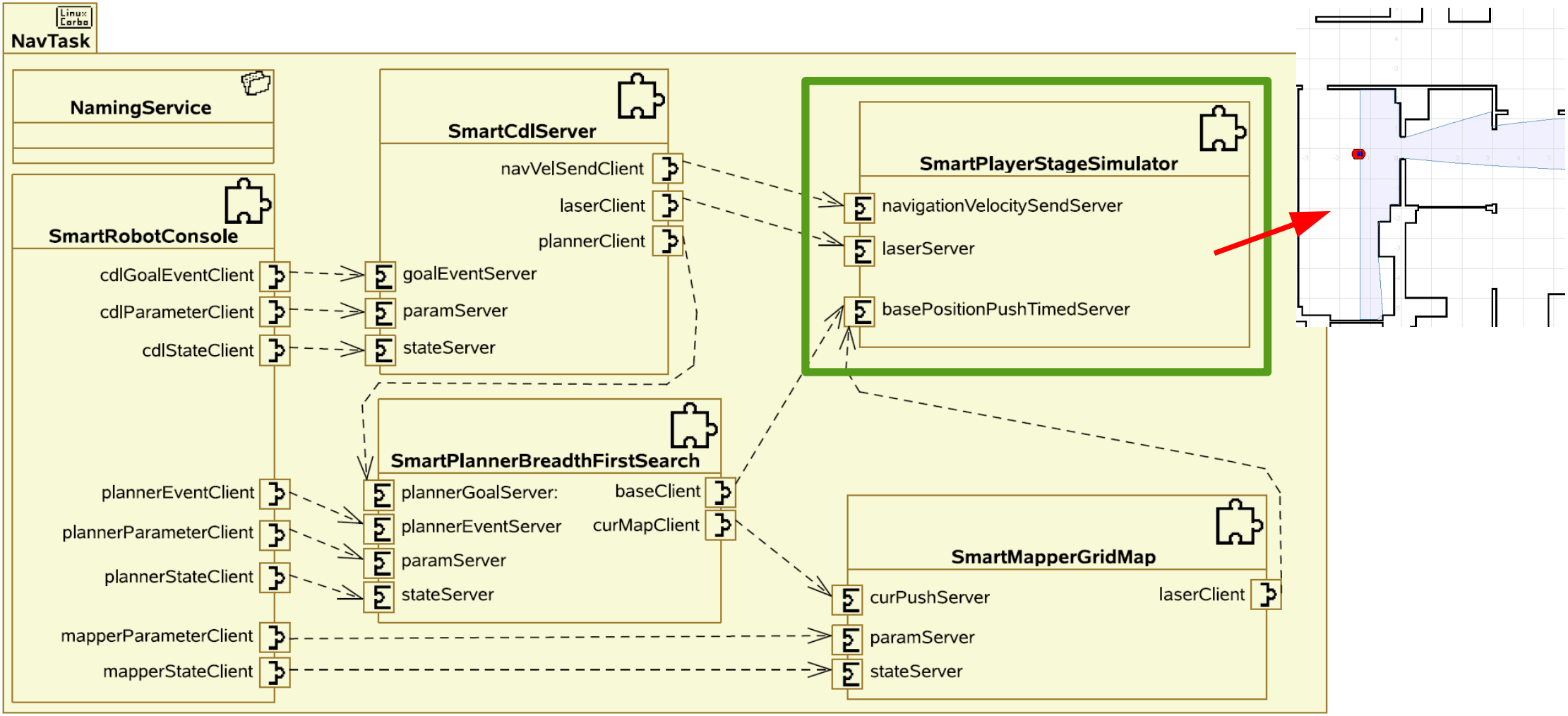


Hochschule Ulm

SmartSoft MDSD Toolchain

Example 2: Navigation Task

deployment navigation task Player/Stage

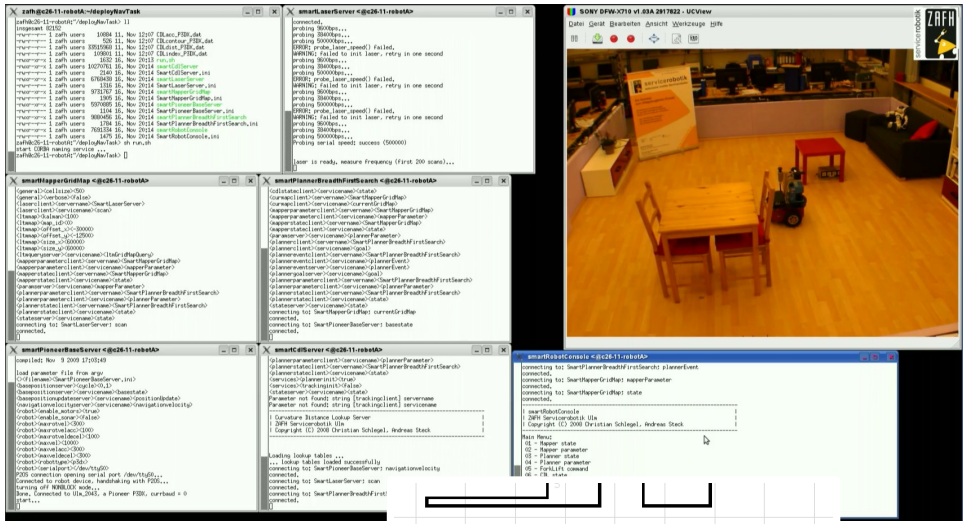


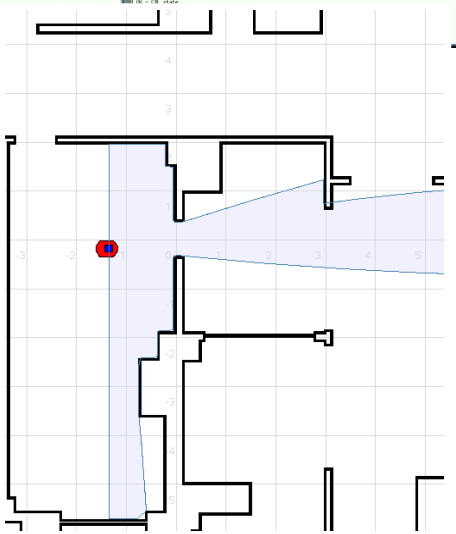
SmartSoft MDSD Toolchain

Example 2: Navigation Task

Live Demo

- how to create a deployment in SmartSoft MDSD Toolchain
- define the initial wiring between the components
- add platform specific details
- add target platform settings
- deploy scenario to target
- run scenario in Player/Stage simulator





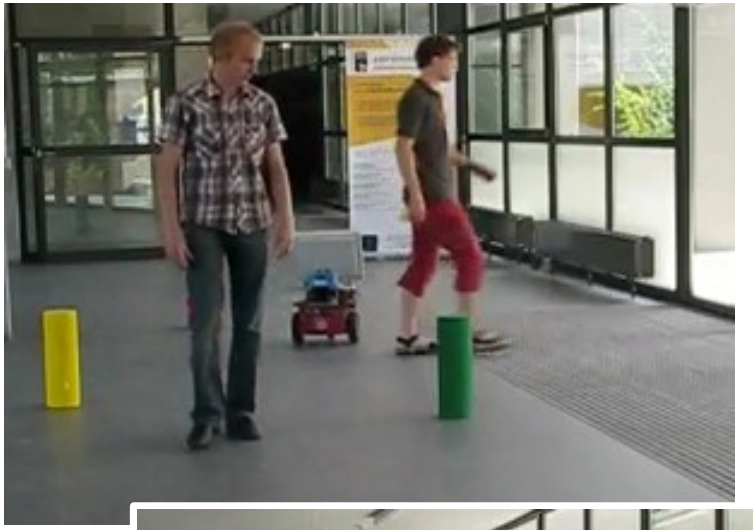
watch on youtube:
http://www.youtube.com/watch?v=4dZnDPgw3fA

attached file:
2009-12-02-deploy-toolchain.mp4

Hochschule Ulm

SmartSoft MDSD Toolchain

Example 3: “Follow Me” - RoboCup@Home

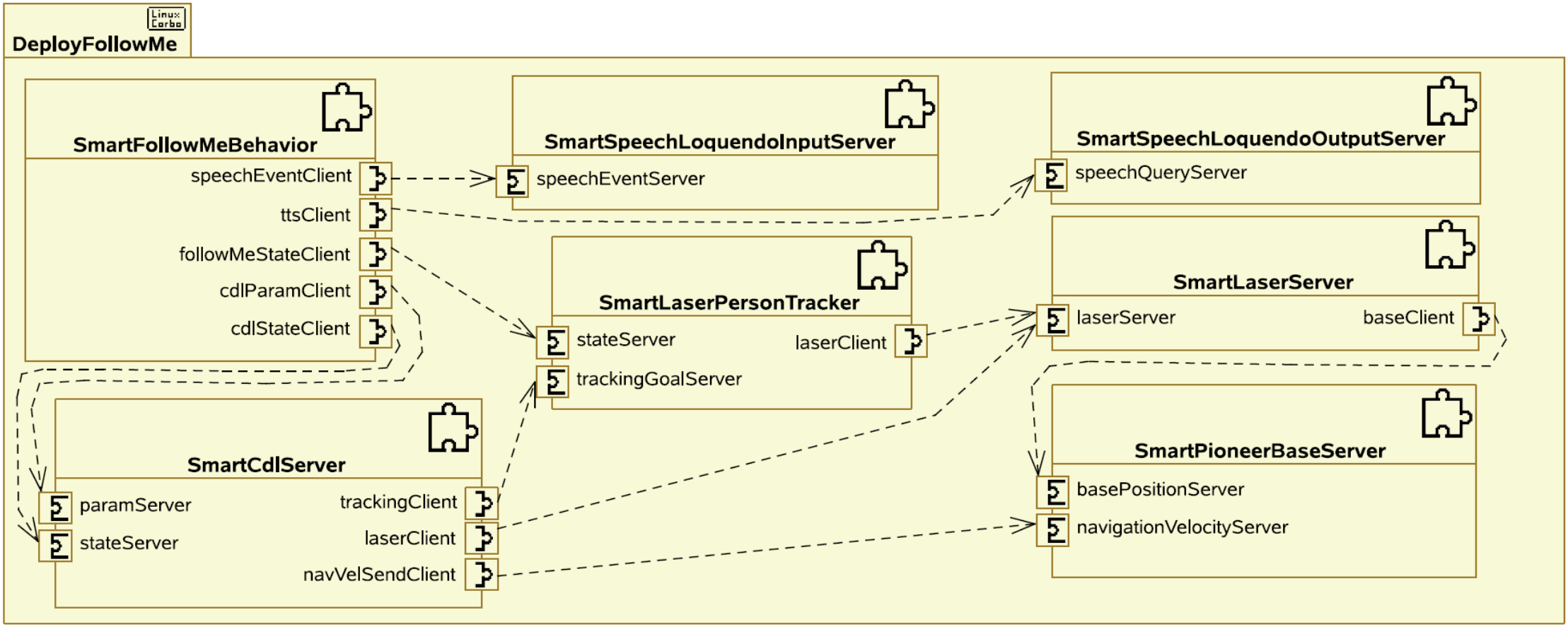


- deployment of COTS components
- reuse for RoboCup@Home
- robot follows a person through an unstructured environment

Hochschule Ulm

SmartSoft MDSD Toolchain

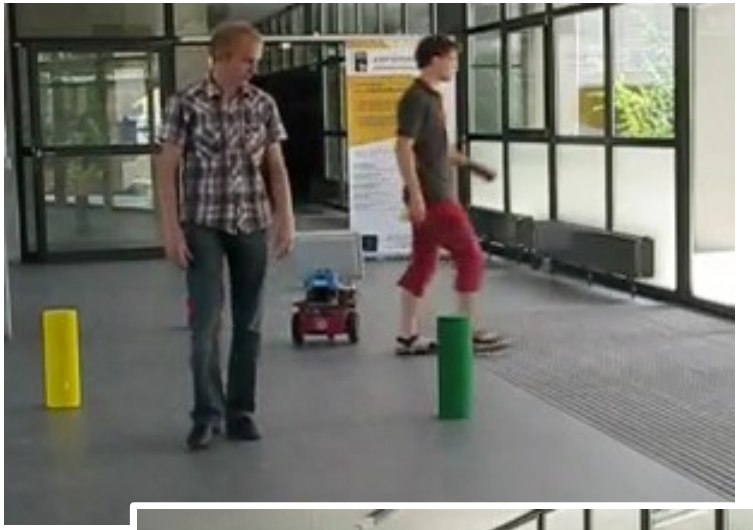
Example 3: “Follow Me” - RoboCup@Home



SmartSoft MDSD Toolchain

Example 3: “Follow Me” - RoboCup@Home

overview



watch on youtube:

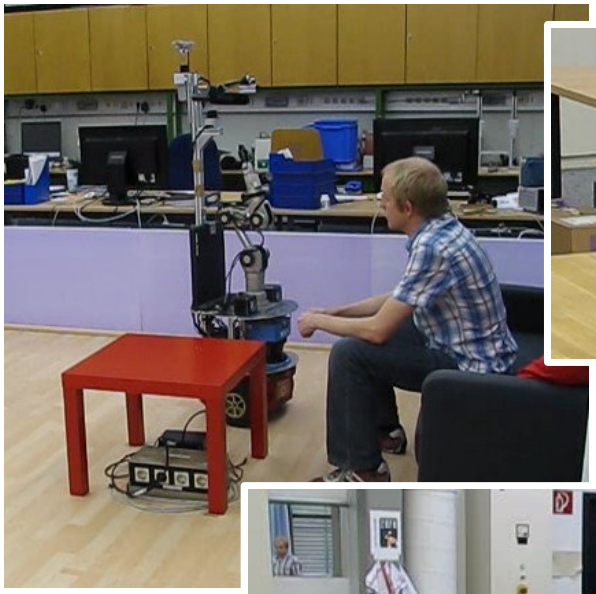
http://www.youtube.com/watch?v=-xvO_bbrb-I
http://www.youtube.com/watch?v=grf-_-32oY0

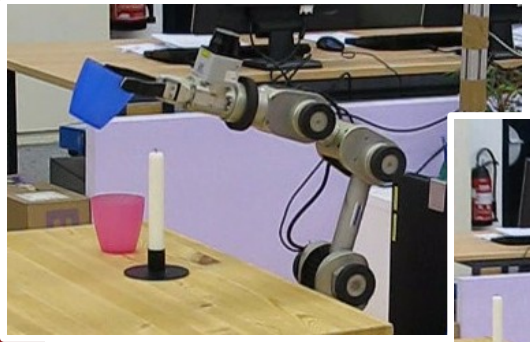
attached files:

2009-09-10-followMe.mp4
2010-02-25-followMe.mp4

SmartSoft MDSD Toolchain

Example 4: Cleanup Table Scenario



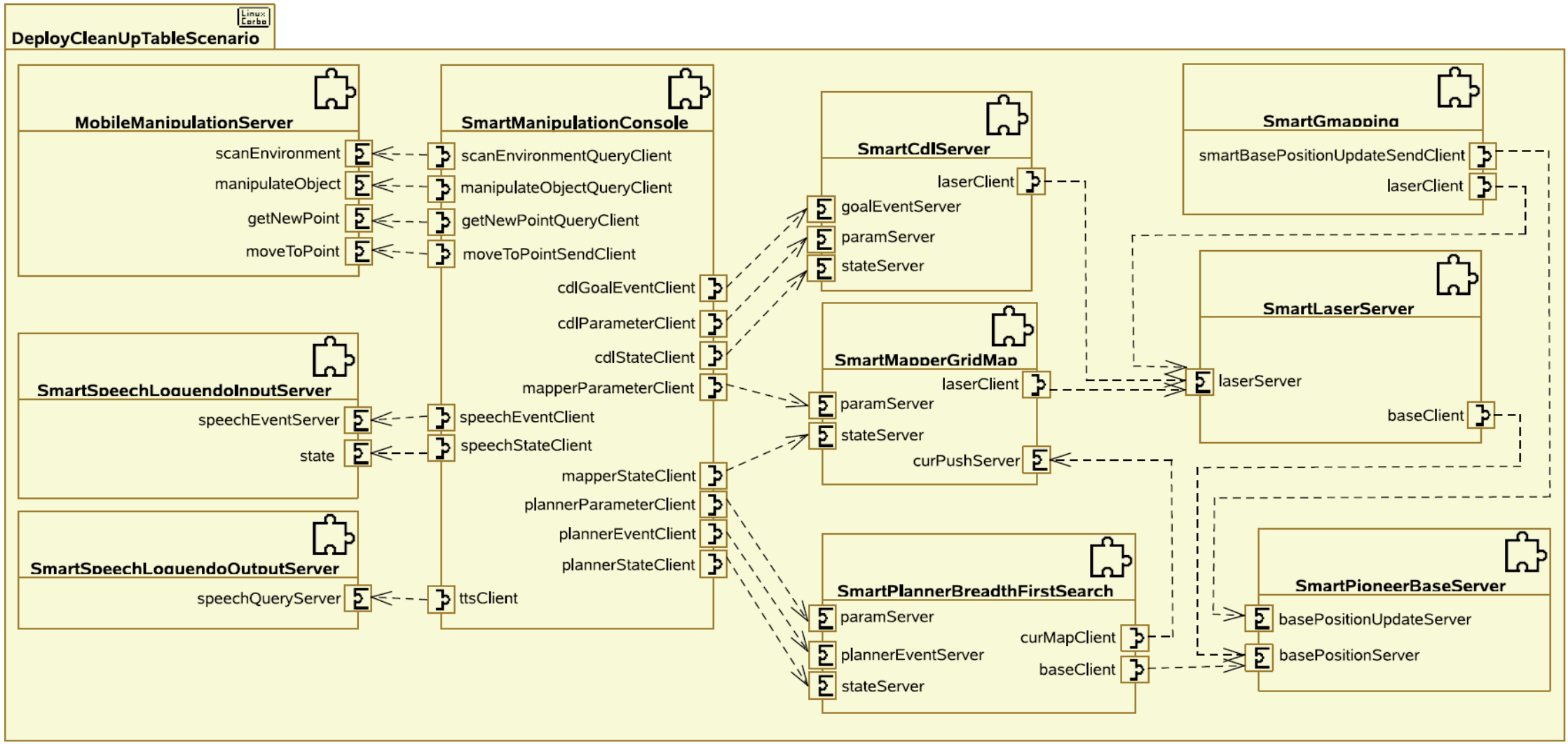


- deployment of COTS components
- complex real world scenario in everyday environment
- robot cleans up the table

Hochschule Ulm

SmartSoft MDSD Toolchain

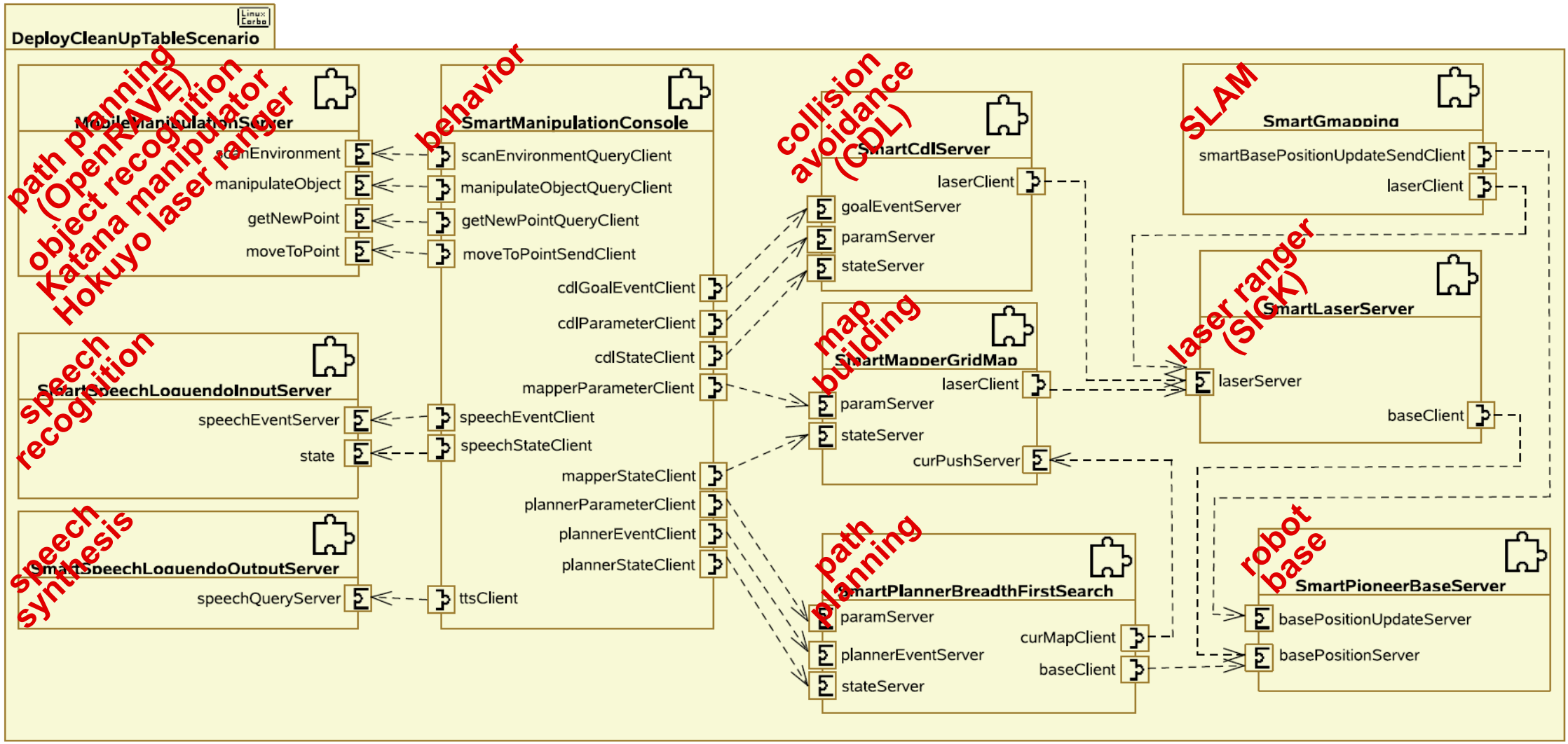
Example 4: Cleanup Table Scenario



SmartSoft MDSD Toolchain

Example 4: Cleanup Table Scenario

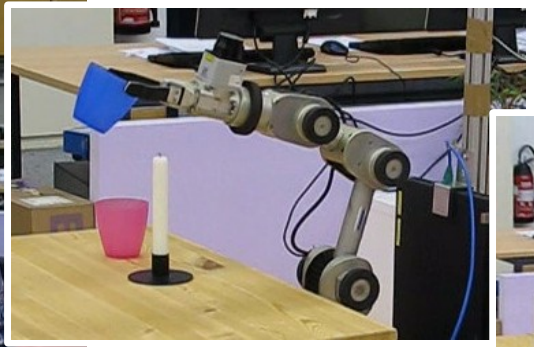
deployment



Hochschule Ulm

SmartSoft MDSD Toolchain

Example 4: Cleanup Table Scenario

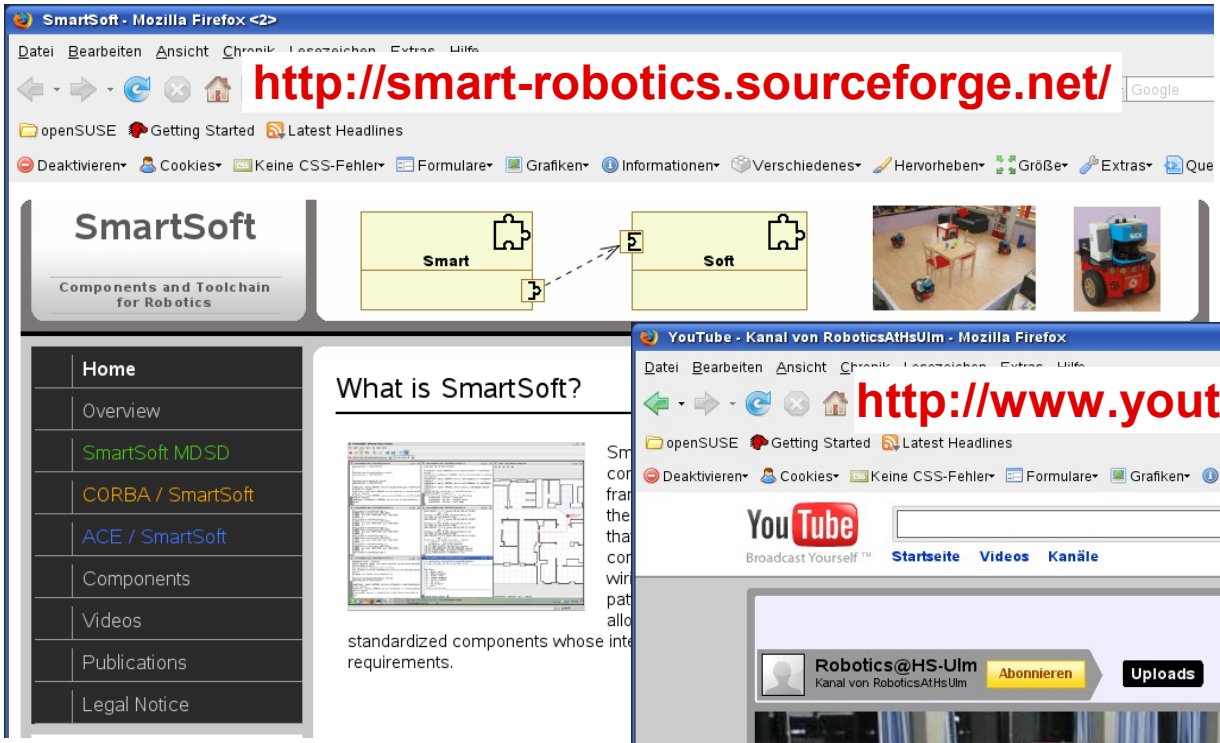


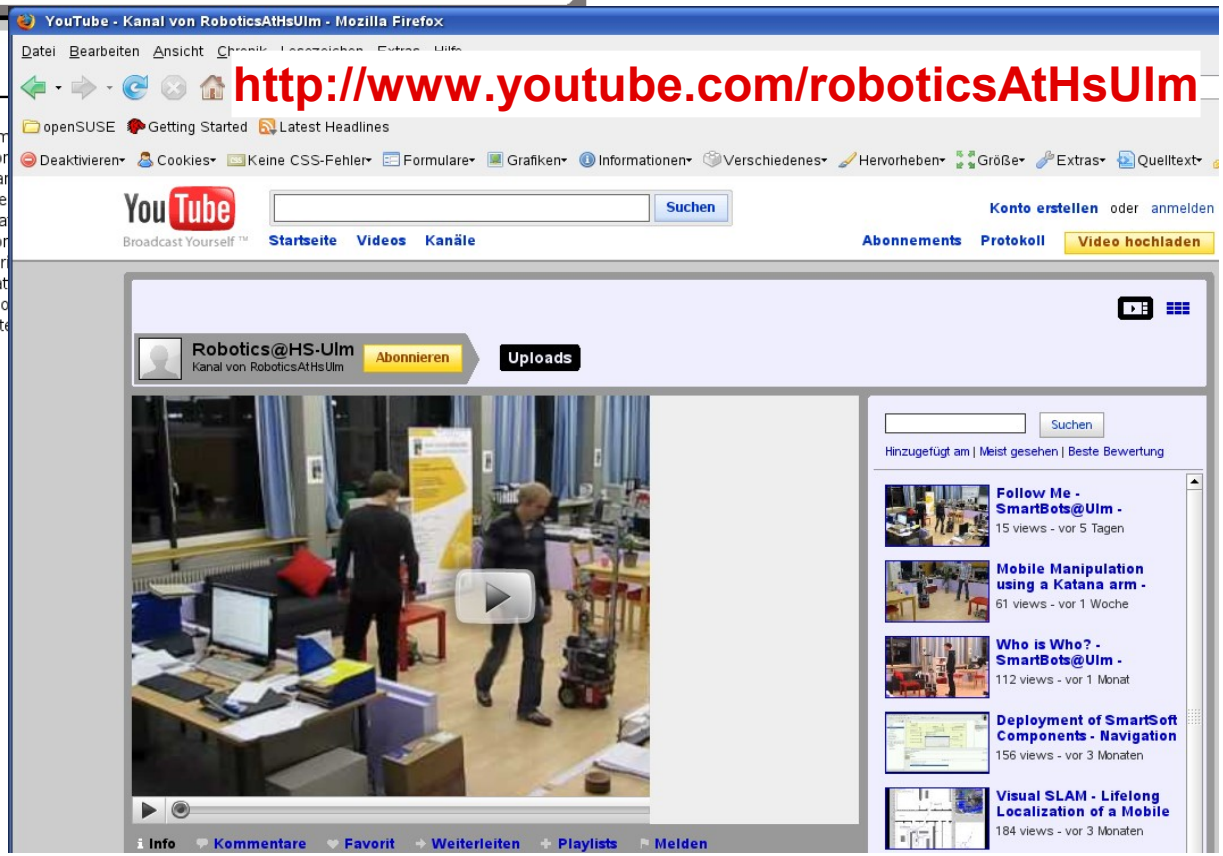
watch on youtube:
http://www.youtube.com/watch?v=40d4DIk5LCQ

attached file:
2010-02-23-cleanUpTable.mp4

Hochschule Ulm

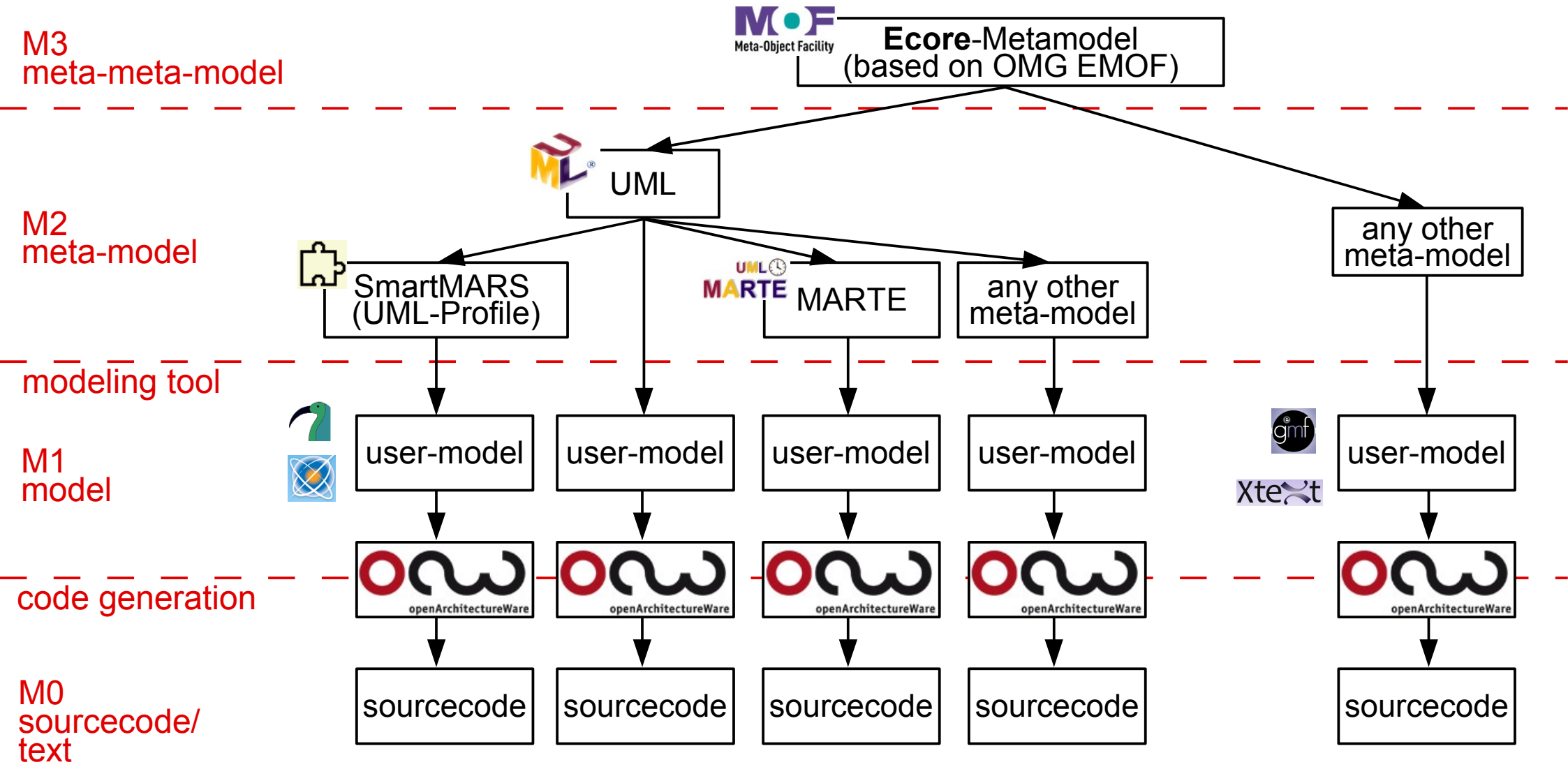
SmartSoft MDSD Toolchain Links



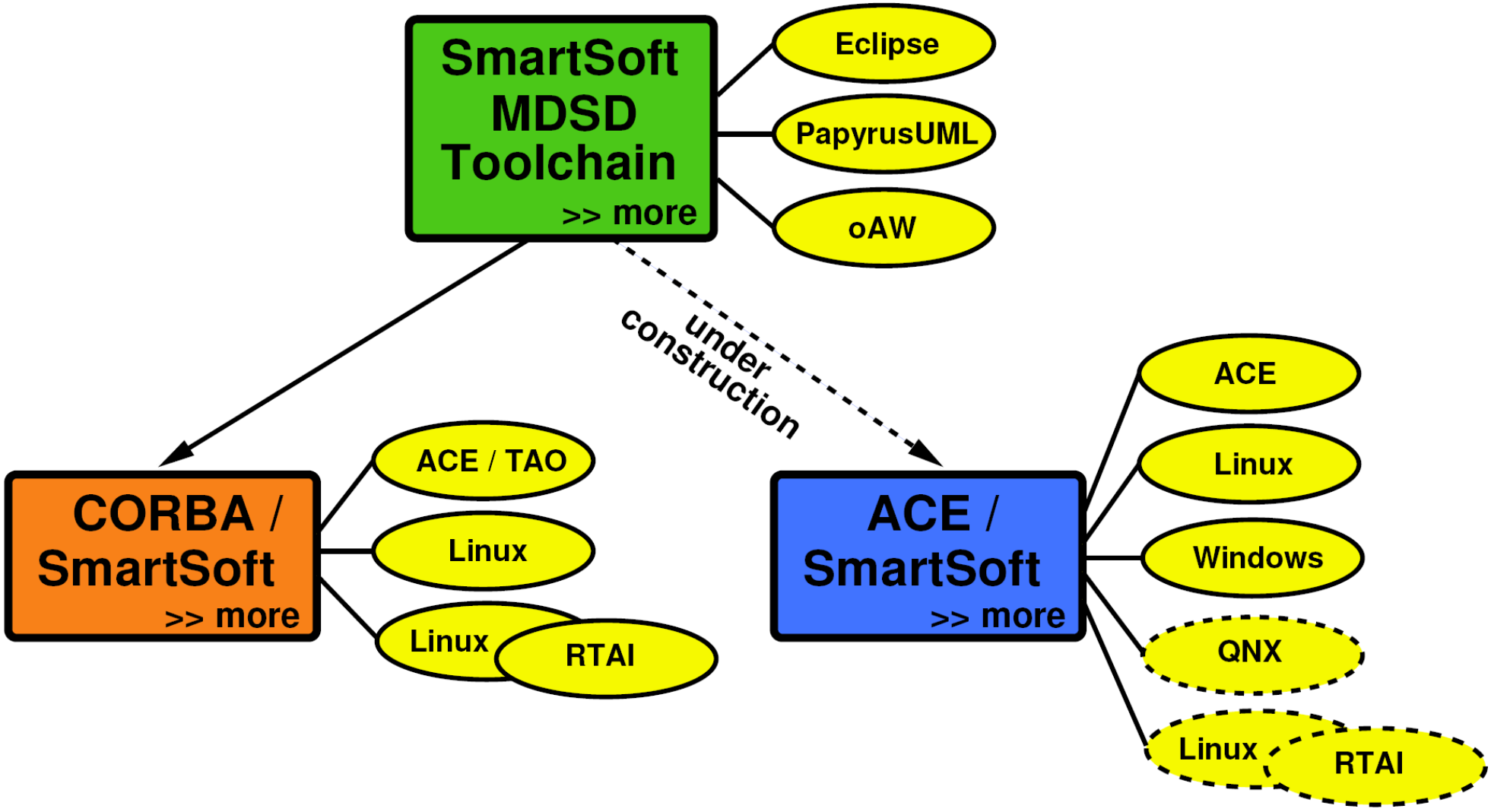


Addendum

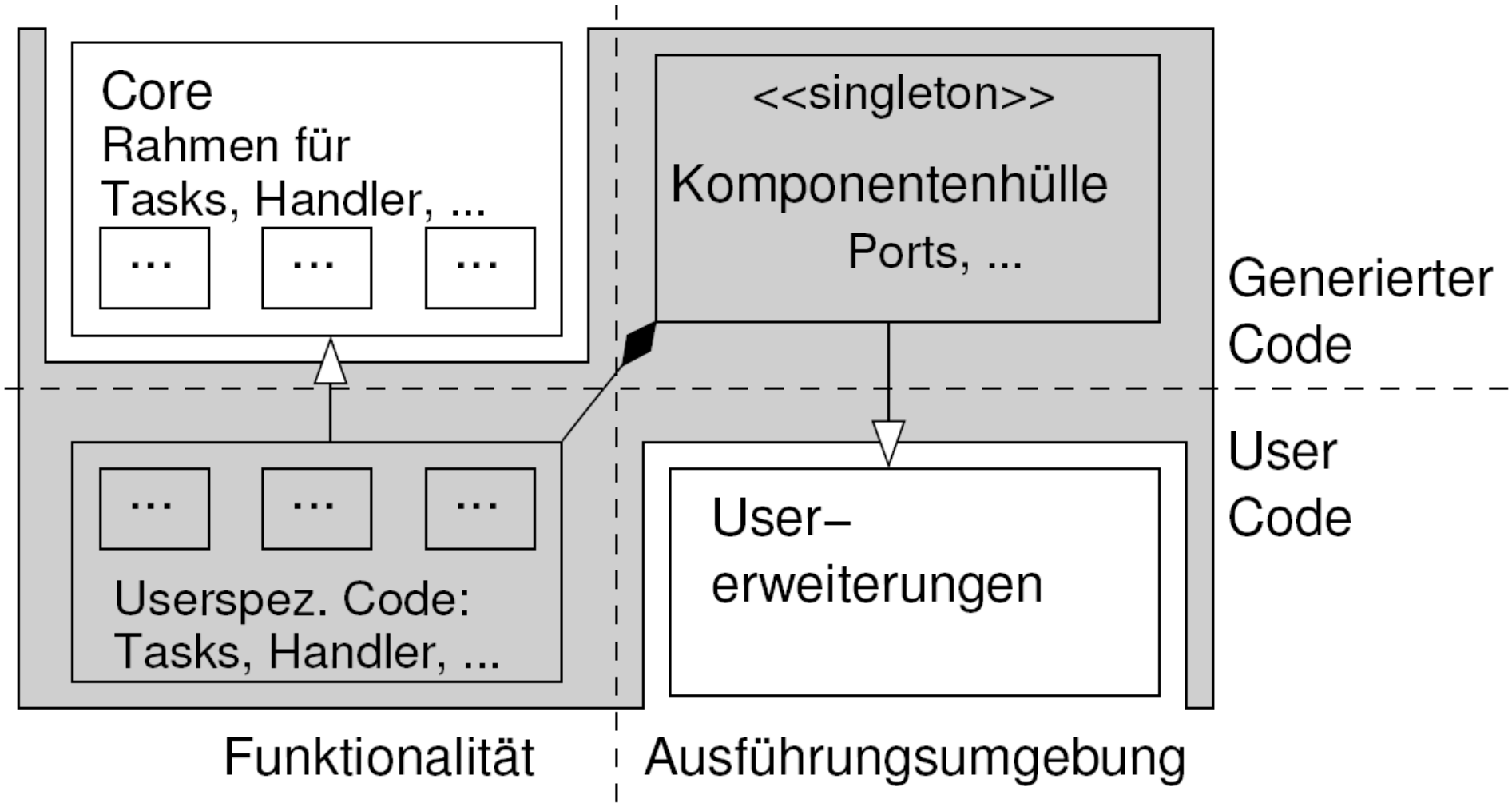
Eclipse Modeling Framework
Meta-Modeling



SmartSoft MDSD Toolchain



SmartSoft MDSD Toolchain
Generation Gap Pattern



SmartSoft MDSD Toolchain

Example 1: Servo (RTAI-Linux)

creating a component workflow

PIM

SmartMARS – Metamodel

(**M**odeling and **A**nalysis of **R**obotics **S**ystems)

- UML2-Profile
- platform independent stereotypes
  - SmartComponent
  - SmartTask
  - SmartMutex
  - SmartQueryServer
  - SmartEventClient
  - ...

M2M
oAW
xTend
check

PSM

CorbaSmartSoft
CORBA based implementation of SmartSoft

AceSmartSoft
ACE based implementation of SmartSoft

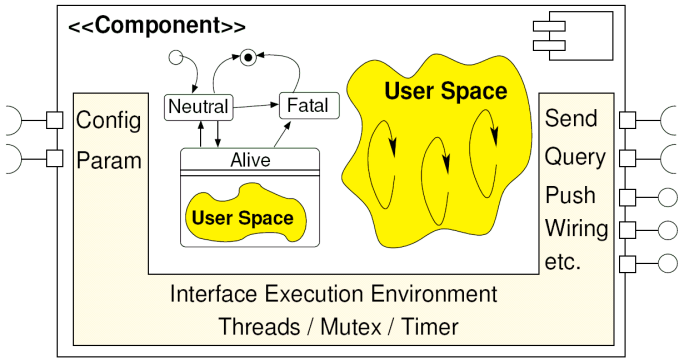
...
any other middleware

- UML2-Profile
- platform specific stereotypes

has to be created by a middleware expert

M2T
oAW
xPand
check

PSI



The User Space can contain arbitrary code and libraries

The User Space stays the same independent of the different platform specific models

Just the component hull will be created

Hochschule Ulm

SmartSoft MDSD Toolchain

creating a component workflow

PIM

SmartMARS – Metamodel

(**M**odeling and **A**nalysis of **R**obotics **S**ystems)

- UML2-Profile
- platform independent stereotypes
  - SmartComponent
  - SmartTask
  - SmartMutex
  - SmartQueryServer
  - SmartEventClient
  - ...

M2M
oAW
xTend
check

PSM

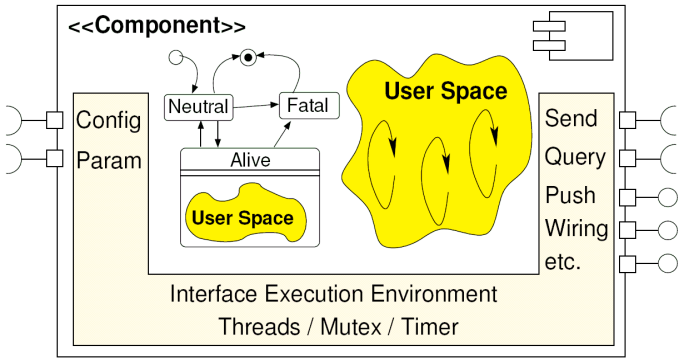
CorbaSmartSoft
CORBA based implementation of SmartSoft

- UML2-Profile
- platform specific stereotypes

has to be created by a middleware expert

M2T
oAW
xPand
check

PSI



The User Space can contain arbitrary code and libraries

The User Space stays the same independent of the different platform specific models

Just the component hull will be created

Hochschule Ulm

SmartSoft MDSD Toolchain

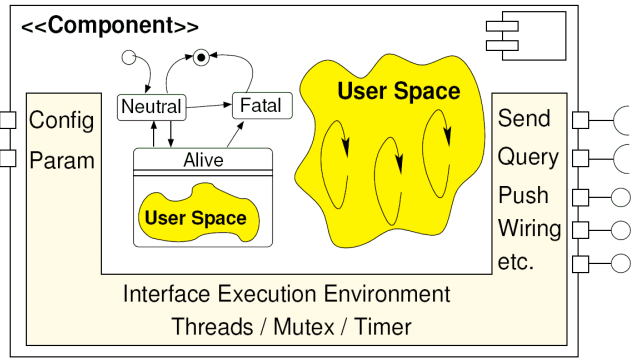
creating a component workflow

PIM	PSM	PSI

SmartMARS – Metamodel

(**M**odeling and **A**nalysis of **R**obotics **S**ystems)

model checking + M2M + M2T



- UML2-Profile
- platform independent stereotypes
  - SmartComponent
  - SmartTask
  - SmartMutex
  - SmartQueryServer
  - SmartEventClient
  - ...

- UML2-Profile
- platform specific stereotypes

has to be created by a middleware expert

The User Space can contain arbitrary code and libraries

The User Space stays the same independent of the different platform specific models

Just the component hull will be created

Hochschule Ulm