



# Nordiasoft

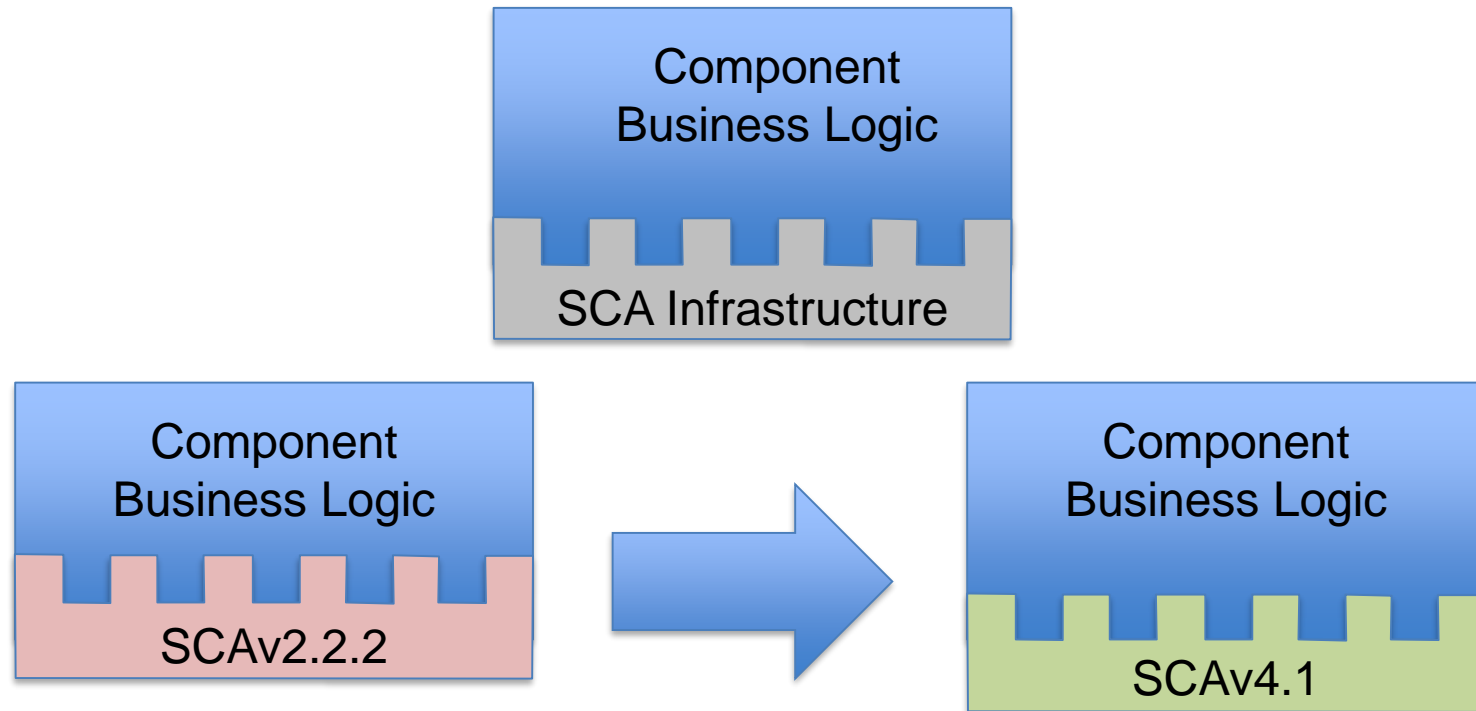
**SCA as infrastructure code: A  
seamless migration from  
SCAv2.2.2 to SCAv4.1**

**Juan Pablo Zamora Zapata**

Copyright © 2017 Nordiasoft.

All rights reserved. This presentation or any portion thereof may not be reproduced or used in any manner whatsoever without the express written permission of the publisher except for the use of brief quotations.

WInnComm 2017 – San Diego, CA - November 2017



**SCA as infrastructure code: A seamless migration from SCAv2.2.2 to SCAv4.1**

# Outline

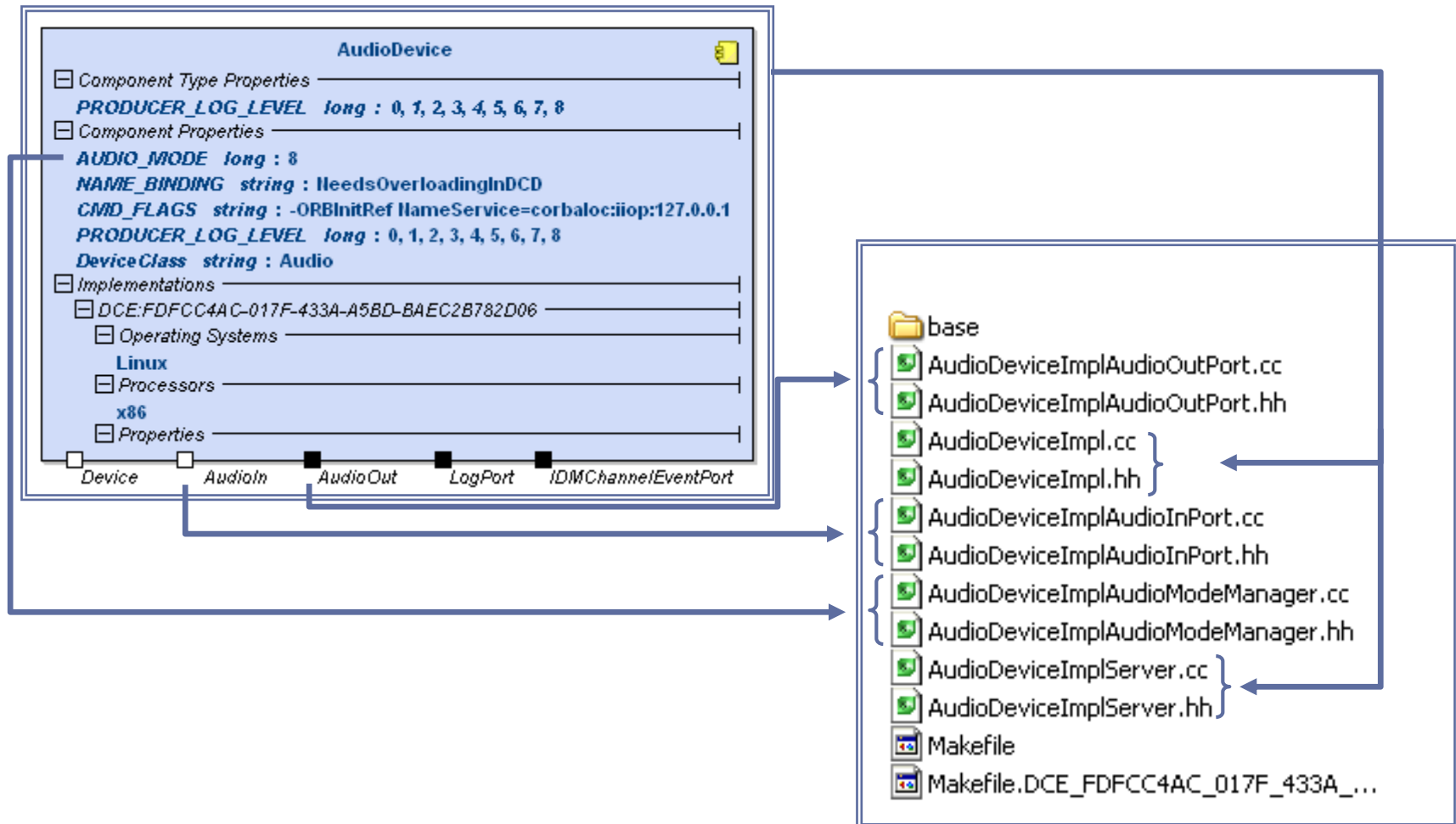
## Zero Merge Code Generation

**Placement of Signal Processing Business Logic**

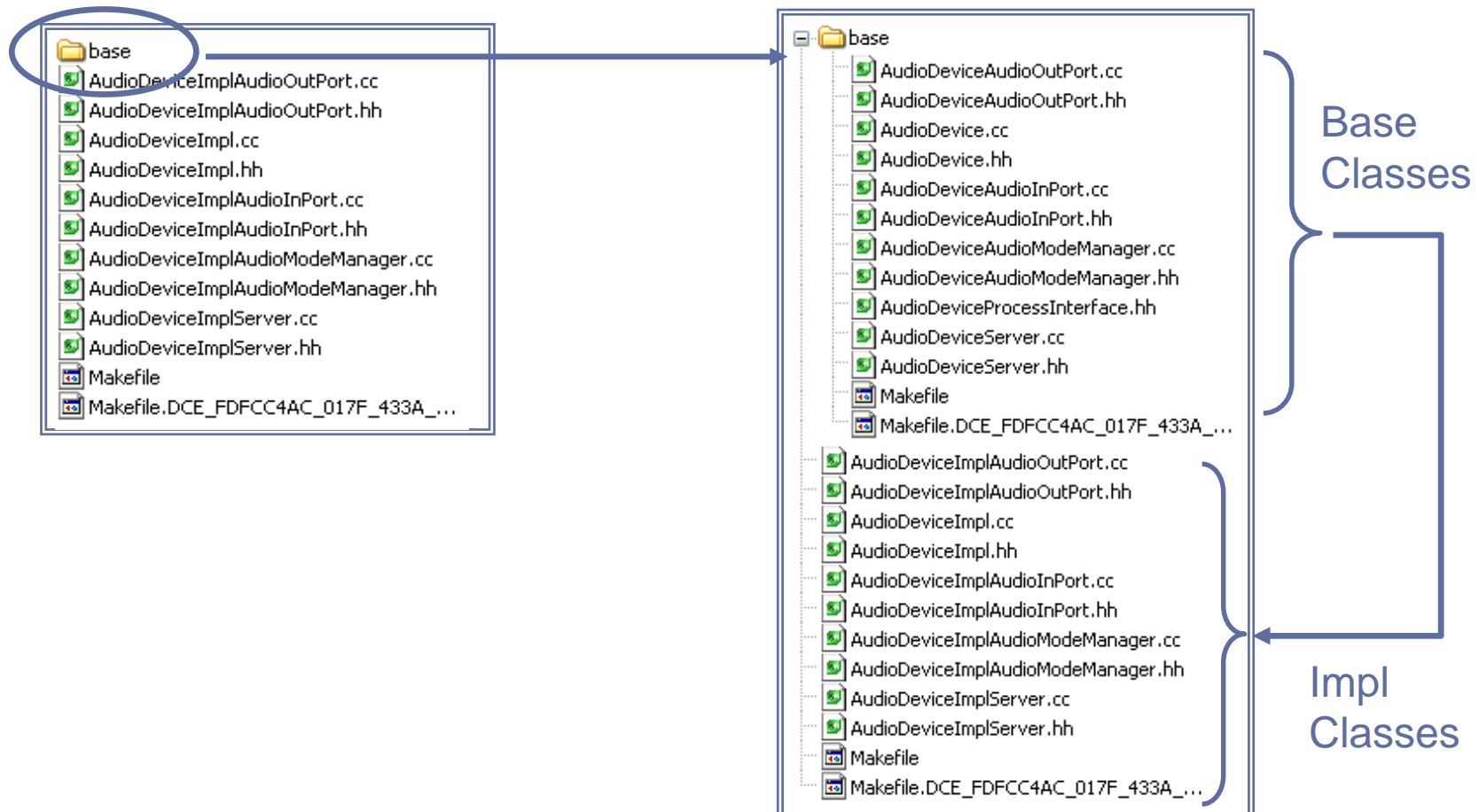
**SCAv2.2.2 vs SCAv4.1 differences**

**Full automatic migration from SCAv2.2.2 to SCAv4.1**

# Zero Merge Code Generation



# Zero Merge Code Generation



# Zero Merge Code Generation

```
#include "AudioDeviceAudioOutPort.hh"

/**
 * Implementation of a CF::Port servant for uses port "AudioOut".
 */
class AudioDeviceImplAudioOutPort :
    public AudioDeviceAudioOutPort
{
public:

    AudioDeviceImplAudioOutPort();

    ~AudioDeviceImplAudioOutPort();

protected:
}; //class AudioDeviceImplAudioOutPort

#endif //AUDIODEVICEIMPLAUDIOOUTPORT_HH
```

## Implementation .hh

AudioDeviceImplAudioOutPort.hh

```
/**
 * Implementation of a CF::Port servant for uses port "AudioOut".
 */
class AudioDeviceAudioOutPort :
    public POA_CF::Port,
    public Synchronizable,
    public PortableServer::RefCountServantBase
{
public:
    PushPorts::DoubleSeqConsumer_var usedObject;

    AudioDeviceAudioOutPort();

    virtual ~AudioDeviceAudioOutPort();

    void connectPort(CORBA::Object_ptr connection, const char* connectionID)
        throw (CORBA::SystemException,
              CF::Port::InvalidPort,
              CF::Port::OccupiedPort);

    void disconnectPort(const char* connectionID)
        throw (CORBA::SystemException,
              CF::Port::InvalidPort);

protected:
    char* connectionID;
}; //class AudioDeviceAudioOutPort

#endif //AUDIODEVICEAUDIOOUTPORT_HH
```

## Base .hh

AudioDeviceAudioOutPort.hh

# Zero Merge Code Generation

```
#include "AudioDeviceImplAudioOutPort.hh"

/**
 * Complete constructor.
 */
AudioDeviceImplAudioOutPort::AudioDeviceImplAudioOutPort() :
    AudioDeviceAudioOutPort()
{
}

/**
 * Default Destructor
 */
AudioDeviceImplAudioOutPort::~AudioDeviceImplAudioOutPort()
{
}
```

## Implementation .cc

AudioDeviceImplAudioOutPort.cc

```
#include <sstream>
#include <string>
#include "AudioDeviceAudioOutPort.hh"

using namespace std;

AudioDeviceAudioOutPort::AudioDeviceAudioOutPort() :
    connectionID(0)
{
    usedObject = PushPorts::DoubleSeqConsumer::_nil();
}

AudioDeviceAudioOutPort::~AudioDeviceAudioOutPort()
{
}

void AudioDeviceAudioOutPort::connectPort(CORBA::Object_ptr connection,
                                           const char* connectionID)
{
    throw (CORBA::SystemException,
          CF::Port::InvalidPort,
          CF::Port::OccupiedPort)

{
    lock();

    if(!CORBA::is_nil(usedObject))
    {
        unlock();
        throw CF::Port::OccupiedPort();
    }

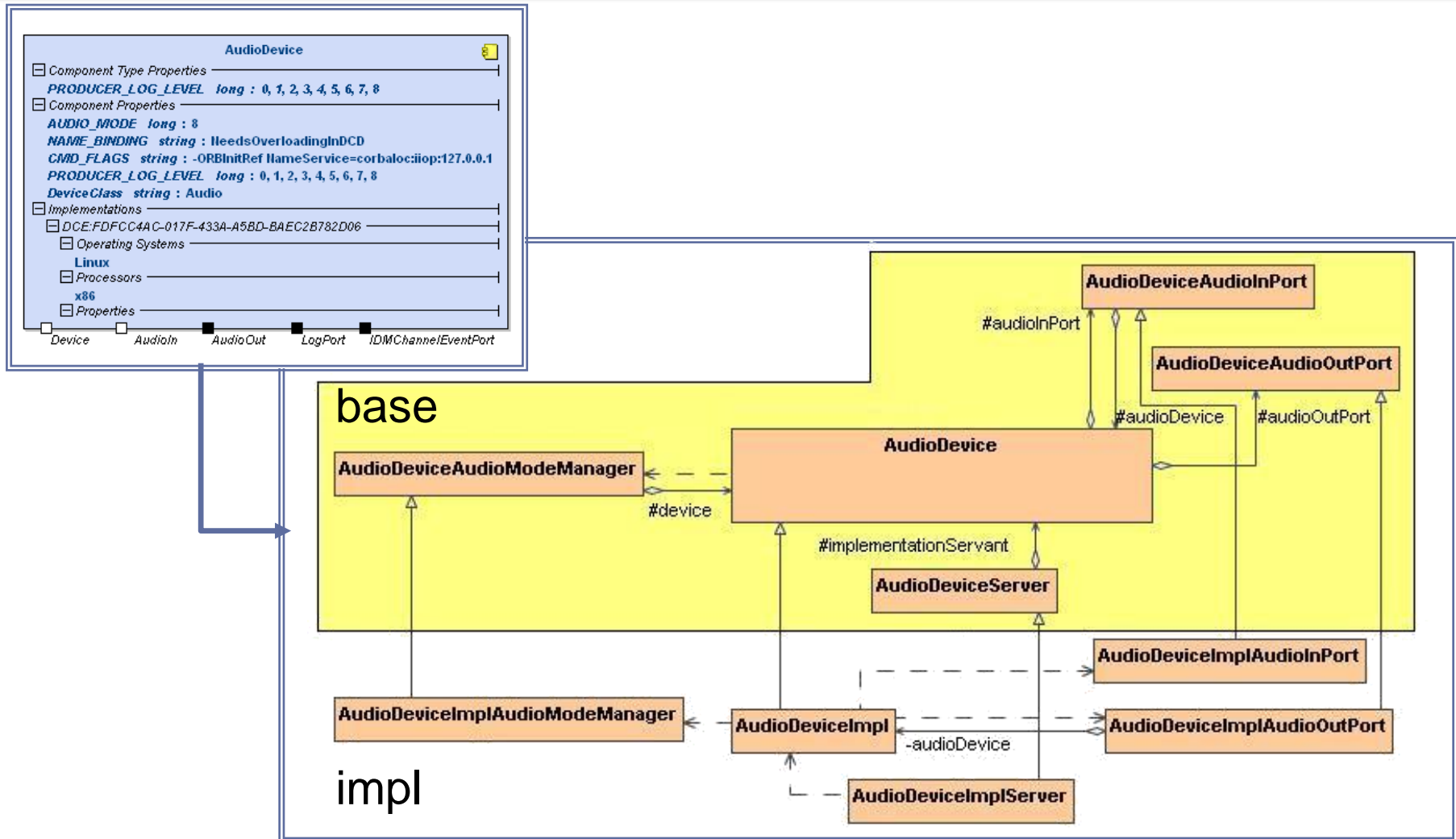
    if((CORBA::is_nil(connection)) ||
        (!connection->_is_a("IDL:PushPorts/DoubleSeqConsumer:1.0")))
    {
        string msg;
        msg.append("[ AudioDeviceAudioOutPort::connectPort] Destination ");
        msg.append("component for connection id= ").append(connectionID);
        msg.append(" must be of type IDL:PushPorts/DoubleSeqConsumer:1.0");
        unlock();
        throw CF::Port::InvalidPort(1, msg.c_str());
    }

    usedObject = PushPorts::DoubleSeqConsumer::_narrow(connection);
    this->connectionID = charDup(connectionID);
    unlock();
} //connectPort
```

## Base .cc

AudioDeviceAudioOutPort.cc

# Zero Merge Code Generation





# Outline

Zero Merge Code Generation

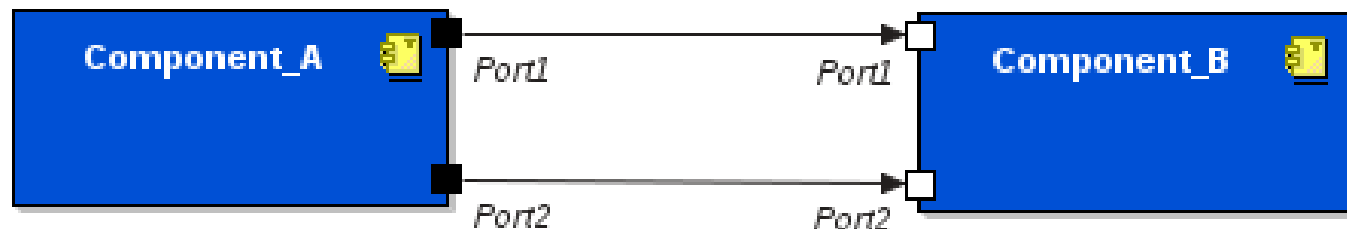
**Placement of Signal Processing Business Logic**

**SCAv2.2.2 vs SCAv4.1 differences**

**Full automatic migration from SCAv2.2.2 to SCAv4.1**

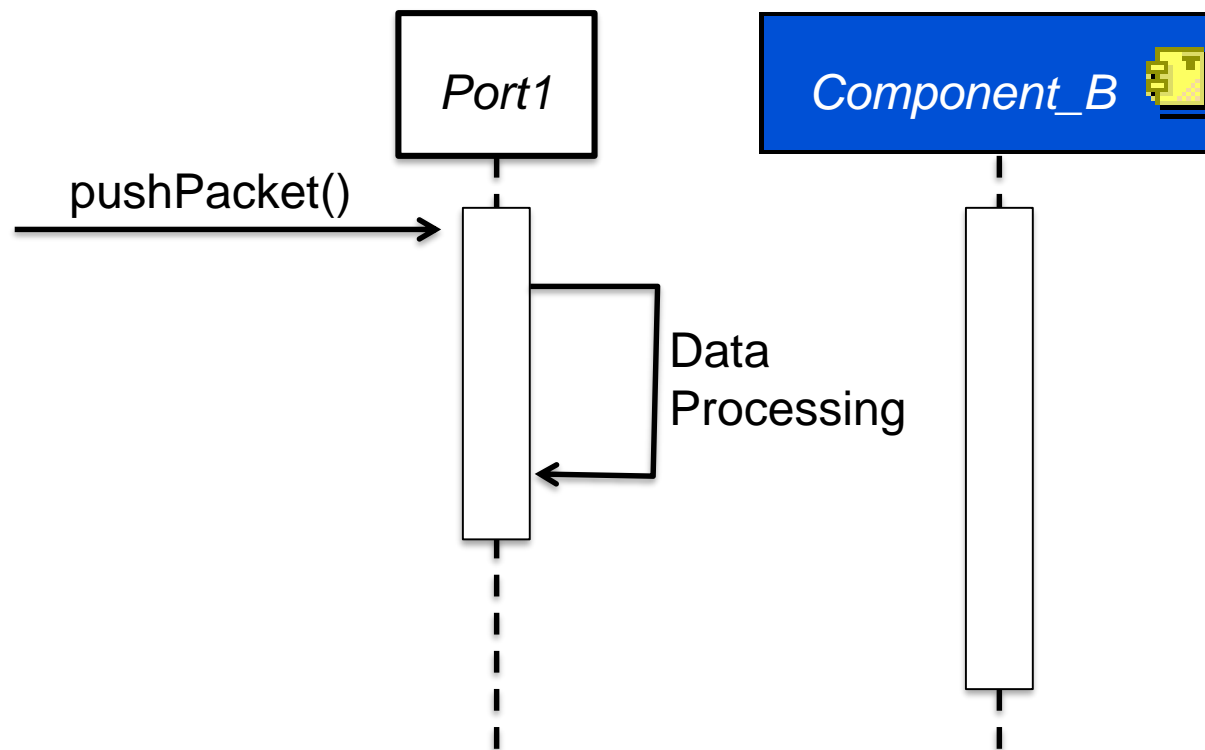
# Placement of Signal Processing Business Logic

- **Embedded in the Modeling Tool**
- **At Source code level**
  - In the port receiving the data
  - In a method of the main class
  - In an external Object



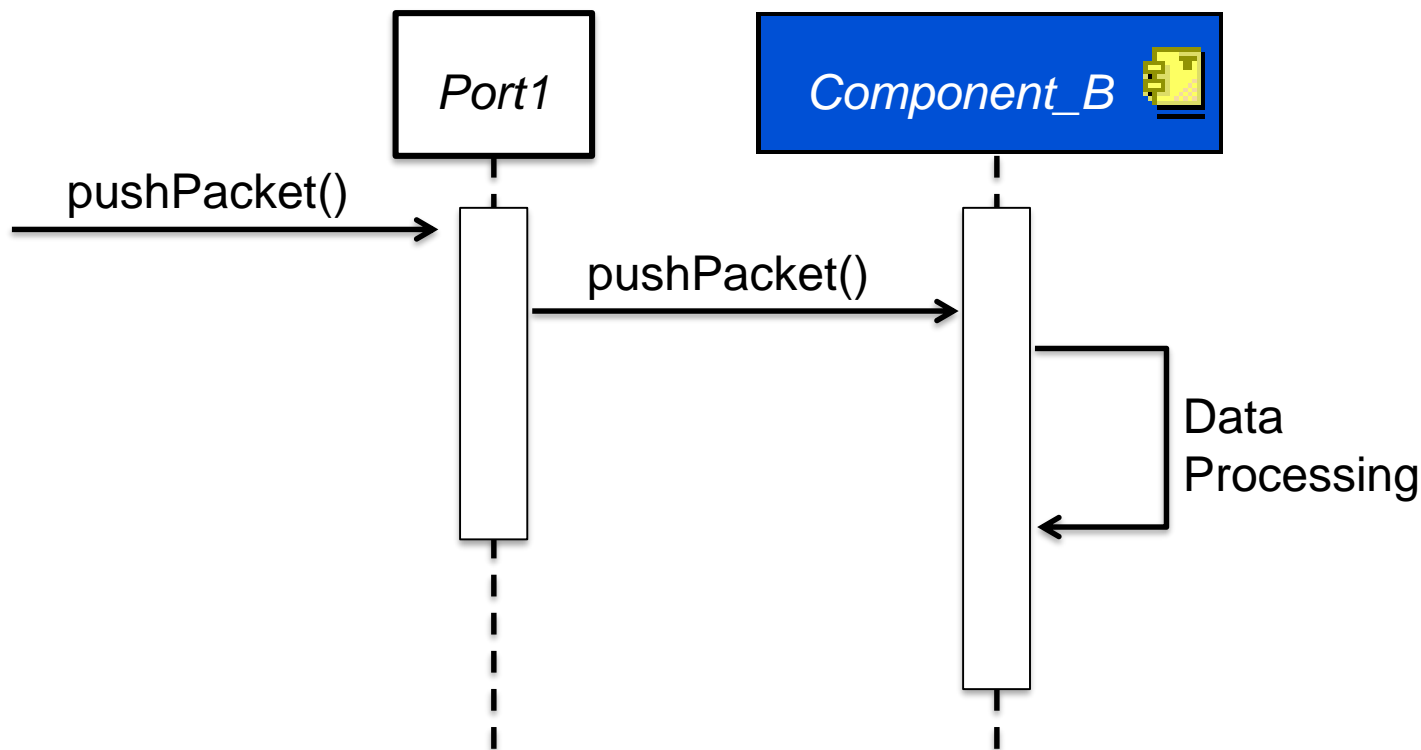
# Placement of Signal Processing Business Logic

- **At Source code level**
  - In the port receiving the data



# Placement of Signal Processing Business Logic

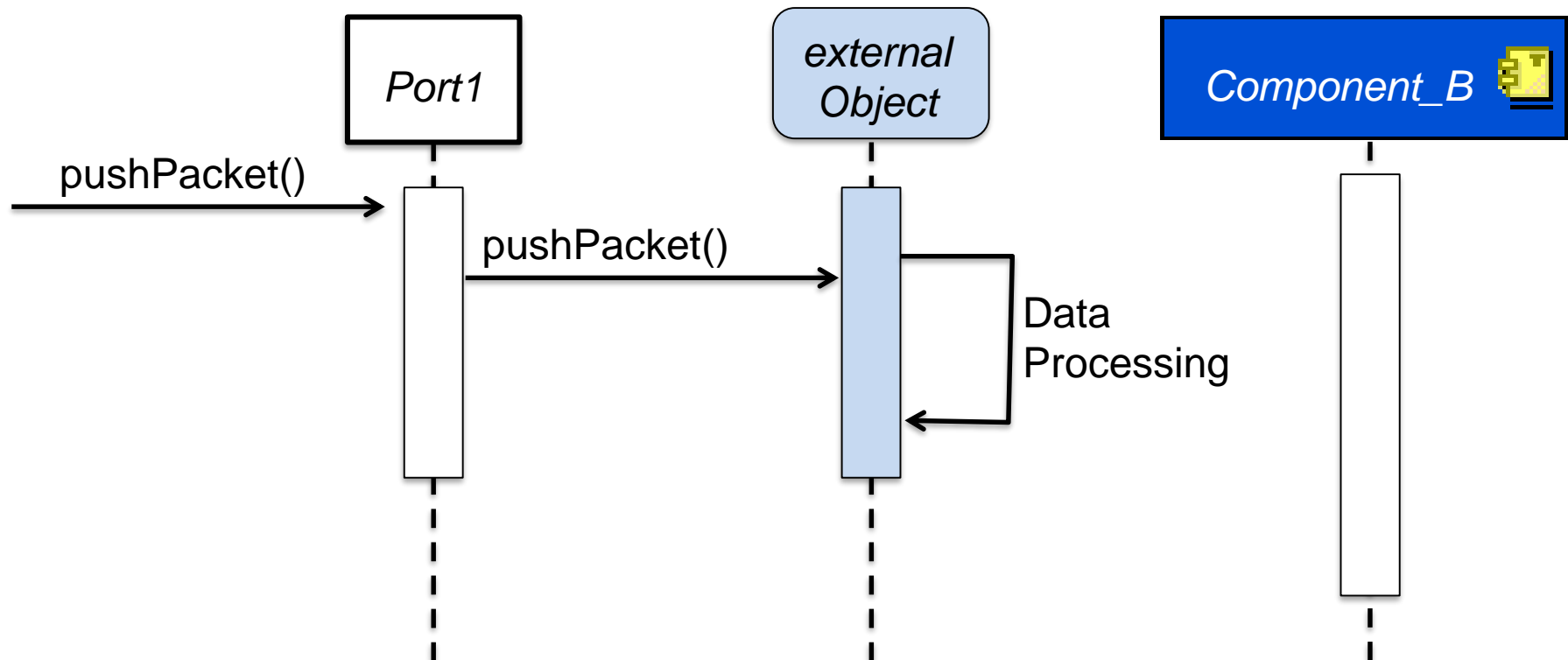
- **At Source code level**
  - In a method of the main class



# Placement of Signal Processing Business Logic

- **At Source code level**

- In an external Object



# Outline

Zero Merge Code Generation

Placement of Signal Processing Business Logic

**SCAv2.2.2 vs SCAv4.1 differences**

**Full automatic migration from SCAv2.2.2 to SCAv4.1**

# SCAv2.2.2 vs SCAv4.1 differences

## ▪ Control

- start(), stop() – Exception namespace change

## ▪ State Machines

- Namespace change

## ▪ Device Registration

- API change - Push & pull vs push only

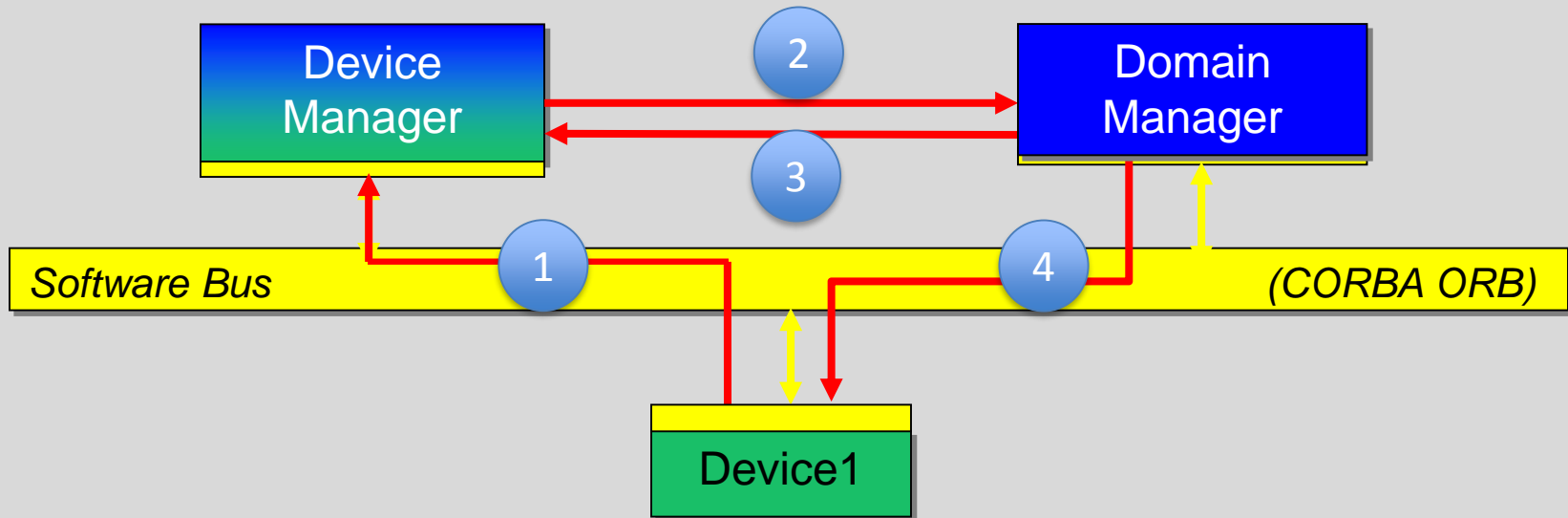
## ▪ Port Connections

- API change - mass connections vs single connections

## ▪ Metadata

- XML Domain Profile changes

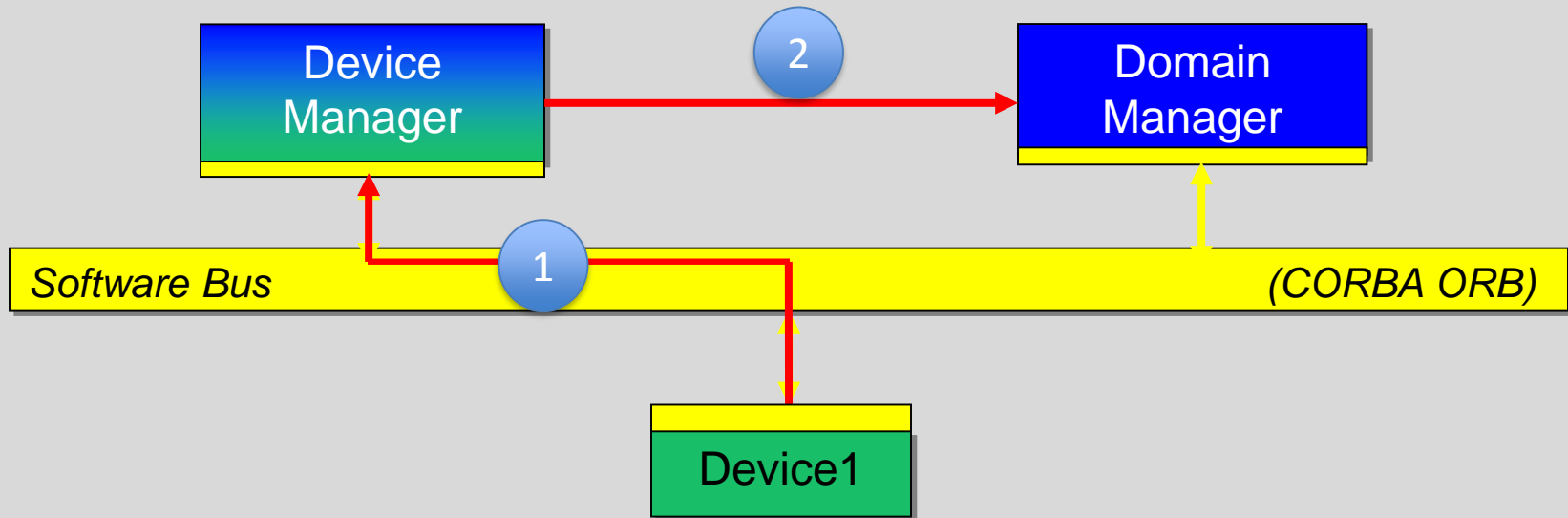
# Device Registration SCAv2.2.2



1. *Device* registers with its *DeviceManager*
2. *DeviceManager* registers *Device* with *DomainManager*
3. *DomainManager* requests *Device* info from *DeviceManager*
4. *DomainManager* requests from *Device* Software Profile (SPD/PRF) to extract advertised capabilities



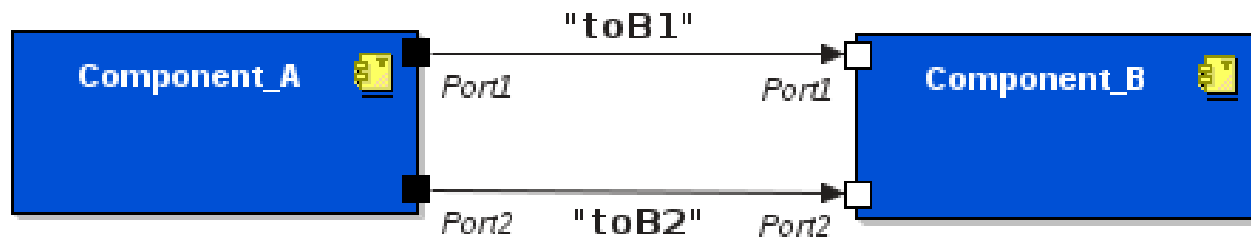
# Device Registration SCAv4.1



1. *Device* registers with its *DeviceManager*
2. *DeviceManager* registers *Device* with *DomainManager*

# Mass Connections

- **SCAv2.2.2**

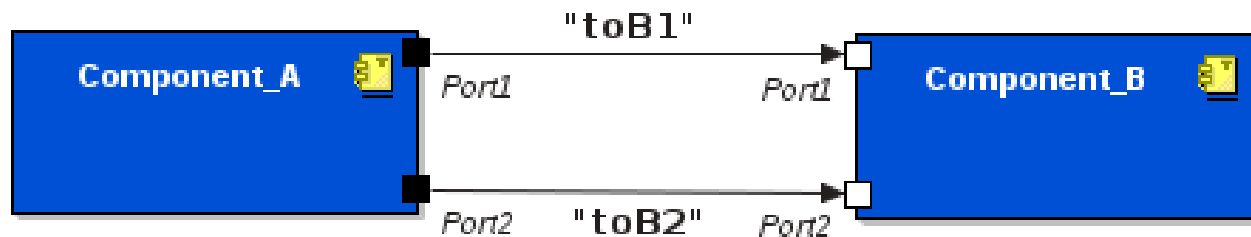


- Connections

```
portA1 = component_A.getPort("Port1")
portB1 = component_B.getPort("Port1")
portA1.connectPort(portB1, "toB1")
portA2 = component_A.getPort("Port2")
portB2 = component_B.getPort("Port2")
portA2.connectPort(portB2, "toB2")
```

# Mass Connections

- SCAv4.1



- Connection

All provides port references are obtained at registration

`component_A.connectUsesPorts("Port1", "toB1", "Port2", "toB2"...)`

# Outline

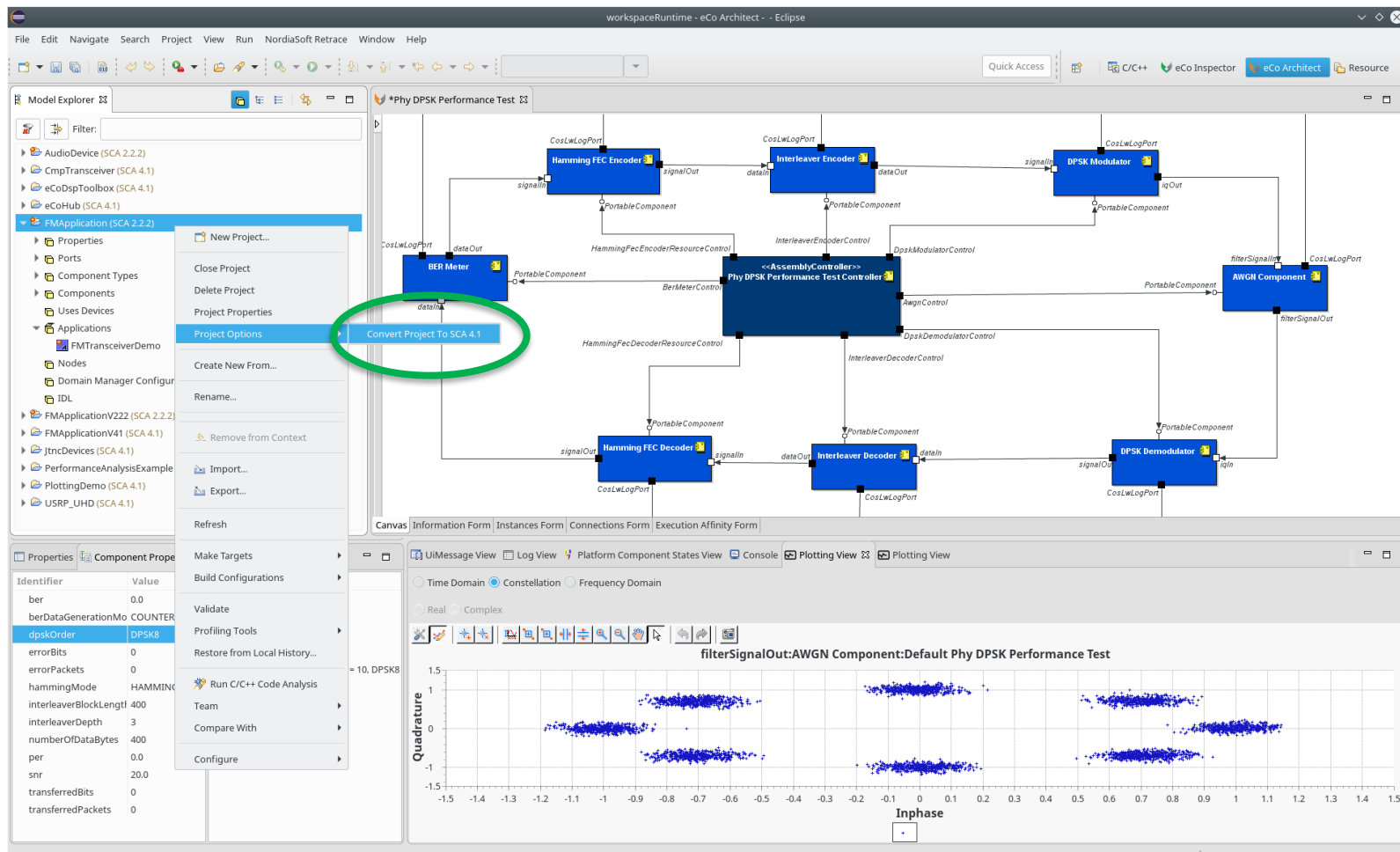
Zero Merge Code Generation

Placement of Signal Processing Business Logic

SCAv2.2.2 vs SCAv4.1 differences

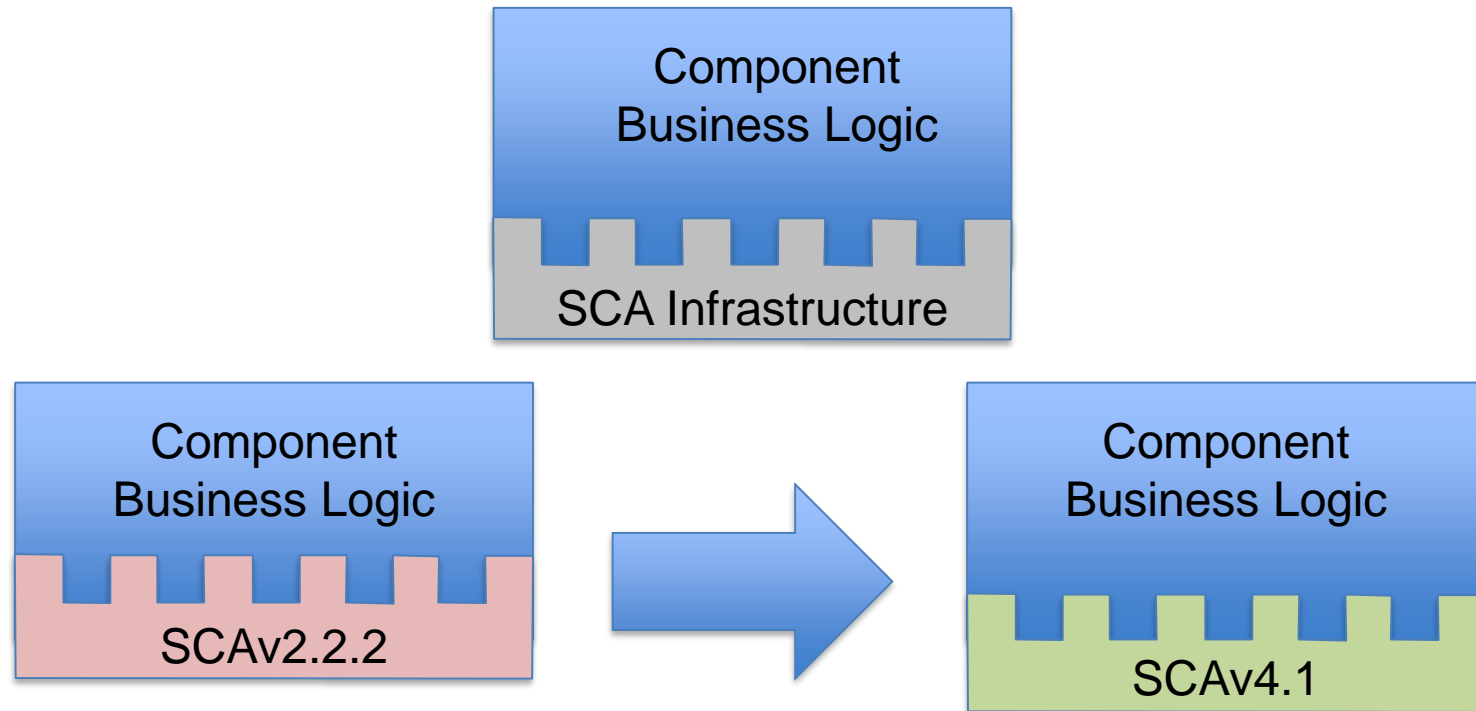
**Full automatic migration from SCAv2.2.2 to SCAv4.1**

# Full automatic migration from SCAv2.2.2 to SCAv4.1



The screenshot displays the NordiaSoft eCo Architect workspace. On the left, the Model Explorer shows a project structure with a context menu open for 'FMApplication (SCA 2.2.2)'. The 'Convert Project To SCA 4.1' option is highlighted with a green circle. The main canvas shows a block diagram of a 'Phy DPSK Performance Test' system. This system includes components like 'Hamming FEC Encoder', 'Interleaver Encoder', 'DPSK Modulator', 'BER Meter', 'Hamming FEC Decoder', 'Interleaver Decoder', and 'DPSK Demodulator', all interconnected with a central 'Phy DPSK Performance Test Controller'. The bottom right panel shows a 'Plotting View' with a constellation plot titled 'filterSignalOut:AWGN Component:Default Phy DPSK Performance Test'. The plot shows data points in the Inphase vs. Quadrature plane, with axes ranging from -1.5 to 1.5. The plot displays four distinct clusters of points, representing the four DPSK symbols.

# Full automatic migration from SCAv2.2.2 to SCAv4.1



**SCA as infrastructure code: A seamless migration from SCAv2.2.2 to SCAv4.1**

# The End