

Software Architecture in Action

Flavio Oquendo, Jair C Leite, Thais Batista



Chapter 9

Designing Modifiability in Software Architectures

Learning outcomes of this chapter

- You will learn:
 - what is modifiability as an architectural quality
 - what are the architectural causes and effects of modifiability
 - tactics to improve modifiability
 - a taxonomy of modifiability primitives
 - a comparison technique to evaluate the modifiability in alternative architectures
- You will know how to:
 - express modifiability in software architecture
 - analyse an architecture to evaluate its modifiability
 - apply tactics to improve modifiability

The structure of this chapter

- Introduction
- Expressing modifiability using software architecture concepts
 - Modifiability causes and effects
 - Modifiability quality attributes
 - A classification of modifiability effects
 - Examples of the add primitive
- Modifiability Tactics
 - Using modifiability tactics
- Analysing Modifiability in the RTC System
 - RTC System requirements
 - RTC System – causes
 - Analysing the ripple effect in ARCH1
 - Design a new architecture: ARCH2
 - Analysing the ripple effects in ARCH2
 - Comparing modifiability in ARCH1 and ARCH2
- Summary



Introduction

Modifiability

Conceptual overview (1/2)

- Modifiability is a **quality to refer to the degree to which a product or system can be effectively or efficiently modified** without introducing defects or degrading existing product quality [ISO 25010:2010]
- **An example of modifiability** in the case of a room temperature controller system is to maintain its functionality when
 - **adding a temperature sensor**
 - to improve accuracy
 - **removing a temperature sensor**
 - to remove a damaged sensor
 - **replacing a temperature sensor**
 - to change a Fahrenheit temperature sensor to a Celsius sensor

Modifiability

Conceptual overview (2/2)

- **Extra-functional requirements** are expressed as quality attributes in the architecture
- The modification in an element can impact other elements (**ripple effect**) requiring new modifications in them.
- The ***degree of modifiability*** is related to the number of elements that need to be modified.
 - The best case is to have no additional modifications in the architecture.



Expressing modifiability using software architecture concepts

Modifiability causes and effects

- Modifiability can be defined as a **cause-effect relationship**
- We need to express the causes and the effects
 - **architectural causes** refer to identify the need of architectural modifications addressing the extra-functional requirements
 - **architectural effects** refer to the impact of each modification in other elements of the architecture
 - all causes of modification need to be identified as a taxonomy
 - for each cause, all effects need to be identified and quantified

Modifiability causes

- We need to consider causes that may require **modifications both in the use and in the definition of elements**
 - **modifications in the use** mean to add, remove, and replace an element in a defined architecture without modifying their types
 - **modifications in the definition** mean to add, remove, and replace, an element in its type definition
- In the RTC system, **examples of modification in use** are
 - to add a new Fahrenheit temperature sensor to increase measure precision
- In the RTC system, **examples of modification in definition** are
 - to add a new port to the sensor monitor component type to receive data from a new temperature sensor

Modifiability effects

- Effects are **the consequences of the causes** in the architecture
 - the **impact on the effectiveness and efficiency** of the system
 - **Effectiveness** refers to meeting the goals of the system
 - **Efficiency** refers to meeting these goals optimizing the use of resources
 - we use quality attributes to **quantify the effects** (see next slide)
- According to the requirements of the RTC System
 - Effectiveness is to **assure that the control-loop correctly adjust the room temperature**
 - according to the user desired temperature, the current average temperature and the presence of a person
 - Efficiency refers to **the resources used to adjust the temperature**

Modifiability quality attributes

- Modifiability quality attributes refers to a **quality used to quantify the effects**
 - The **ripple effect**
 - refers to a **spreading effect** caused by a modification in an architectural element
 - **as an example**
 - if we add a new sensor, we need to add a new port to the sensor monitor, to create a new connector, and to connect the sensor to the monitor. To add that new port we need to modify the sensor monitor definition.
 - The **cost of modifications**
 - refers to the **global cost implied by the modifications**
 - **as an example**
 - if we add a new sensor, the cost of this modification includes the cost of the sensor, the cost of modifying the architecture, the cost of refining the architecture towards the implementation, and the cost of deploying the system

A classification of modifiability effects

Action primitives

- We can express modifiability effects using **the following action primitives**

- **Types**

- add, remove, or replace a component type, a port type, a connector type, a value type and a configuration type
- configuration types
 - add, remove, or replace a component
 - add, remove, or replace a connector
 - add, remove, or replace a port in a component
- component types
 - add, remove, or replace a port

To modify a connector type we can replace it

- **Instances**

- add, remove, or replace a component in a configuration
- add, remove, or replace a port in a component
- add, remove, or replace a connector between the ports of components in a configuration

A classification of modifiability effects

Examples of the add primitive

- We can add existing instances using the ***add* primitive**
 - **Types**
 - Component types
 - **add** a port in a component type
 - **add** localTemp3:CTemperatureIPT in RoomTemperatureControllerCP type
 - **Instances**
 - **add** a component in a configuration
 - **add** s3:TemperatureSensorCP in RTCSysARCH2 configuration
 - **add** a connector in a configuration
 - **add** c3:FahrenheitToCelsiusCN between localTemp3:CTemperatureIPT of rtc:RoomTemperatureControllerCP and current:FTemperatureOPT of s3:TemperatureCP



Modifiability Tactics

Modifiability Tactics

- **Tactics** are design decisions that influence quality attributes
 - **Tactics** are used to minimize effects
- Example of modifiability tactics:
 - **T1 - localize modifications**
 - to assign responsibilities to elements during design such that anticipated changes will be limited in scope
 - as an example, in the RTC System, we can design a component to each responsibility: a component to monitor sensor, another to monitor presence and a third component to control the heater and cooler
 - **T2 - minimize ripple effects**
 - to avoid that a modification in an architectural element causes a series of other modifications
 - as an example, in the RTC System, we can design a component using a multiplex port: a monitor sensor temperature having a multiplex port does not need to be modified when adding a new sensor.



Analysing Modifiability in the RTC System

Modifiability example

- In this example, **we present two modifiability requirements**. For each requirement, we enumerate the causes and then we depict the effects. Our analysis is not complete. We explain the analysis for one cause in two different architectures of the RTC system.
- First, we analyse the *ARCH1* architecture.
 - This architecture **has two simple ports in the controller to receive data from two sensors**.
- Then, we design and analyse the *ARCH2* architecture.
 - This architecture **has one port that allows multiple connectors**.

Modifiability example

RTC System – modifiability requirements

- We use an AND/OR tree to depict the requirements (R), causes (C) and effects (E)
- RTC modifiability requirements (AND)
 - R1 – The system must allow the modification of the temperature sensors to improve temperature measures
 - R2 – The system must allow the modification of the user interface to display the current average temperature

Modifiability example

RTC System – causes

- RTC modifiability requirements (AND)
 - R1 – The system must allow the modification of temperature sensors to improve temperature measures (OR)
 - C1 – Add a new temperature sensor of a defined type to improve accuracy
 - C2 – Add a new temperature sensor of a new type to include sensors supporting other temperature metric systems
 - C3 – Replace a temperature sensor of the same type
 - C4 – Replace a temperature sensor for a new type to improve accuracy or to support other temperature units
 - C5 – Remove a temperature sensor when it is not needed anymore
 - R2 – The system must allow the modification of the user interface to display the current average temperature (OR)
 - C6 – Replace the user interface

Modifiability example

Analysing the ripple effect in ARCH1

- We analyse the ripple effect on **two different architectures of the RTC system** considering the same cause of modification
 - the modification is to improve temperature measures (**Requirement R1**) by adding a new sensor to the RTC system (**Cause C1**)

Modifiability example

RTC System ARCH1 – effects (1/2)

- Here we depict the effects to cause C1
- C1 – add a new temperature sensor component of a defined type to improve accuracy implies to (AND)
 - E1 – add a NewRoomTemperatureControllerCP component type, which is to (AND)
 - add NewRoomTemperatureControllerCP as a subtype of RoomTemperatureController
 - add the localTemp3:CTemperatureIPT port in NewRoomTemperatureControllerCP
 - E2 – add a NewSensorMonitorCP component type, which is to (AND)
 - add a NewSensorMonitorCP as a subtype of SensorMonitorCP
 - add the s3:CTemperatureIPT port to the SensorsMonitorCP component type

Modifiability example

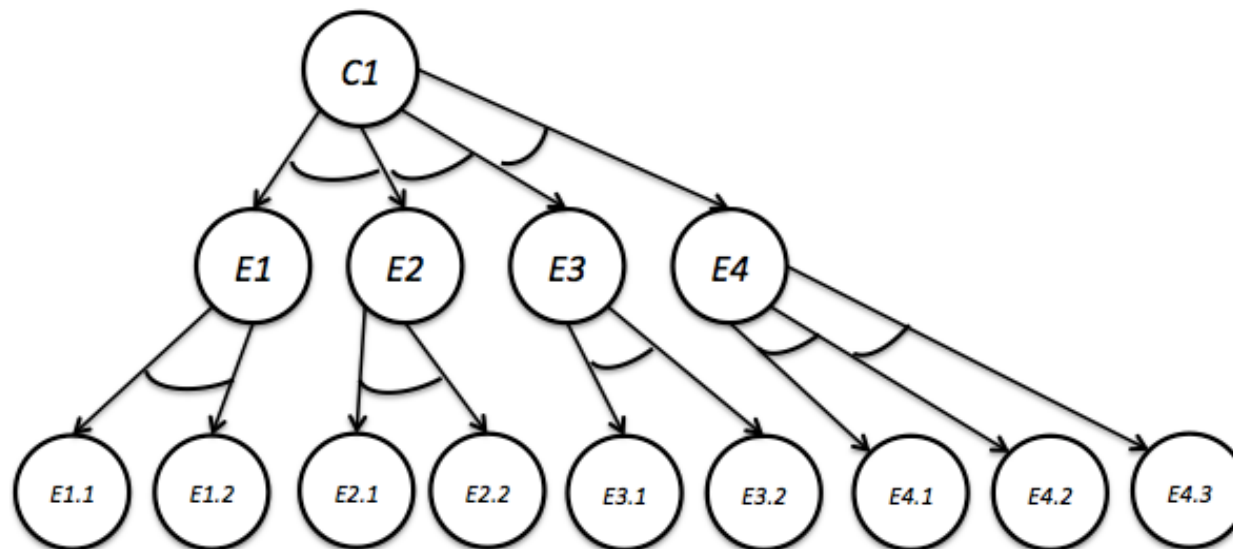
RTC System ARCH1 – effects (2/2)

- Here we depict the effects to cause C1 (continued)
 - E3 - modify the RoomTemperatureControllerCP configuration, which is to (AND)
 - replace the sm:SensorsMonitorCP component by the **sm:NewSensorMonitorCP** component
 - add a delegation between s3 and localTemp3
 - E4 - modify the RTC System ARCH1 configuration, which is to (AND)
 - replace rtc:RoomTemperatureControllerCP by **rtc:NewRoomTemperatureControllerCP**
 - add the s3:TemperatureSensorCP component in ARCH1 configuration
 - add c3:FahrenheitToCelsiusCN between localTemp3:CTemperatureIPT of rtc:RoomTemperatureControllerCP and current:FTemperatureOPT of s3:TemperatureCP

Modifiability example

RTC System ARCH1

- An and-or tree of the C1 cause and the corresponding effects



Modifiability example

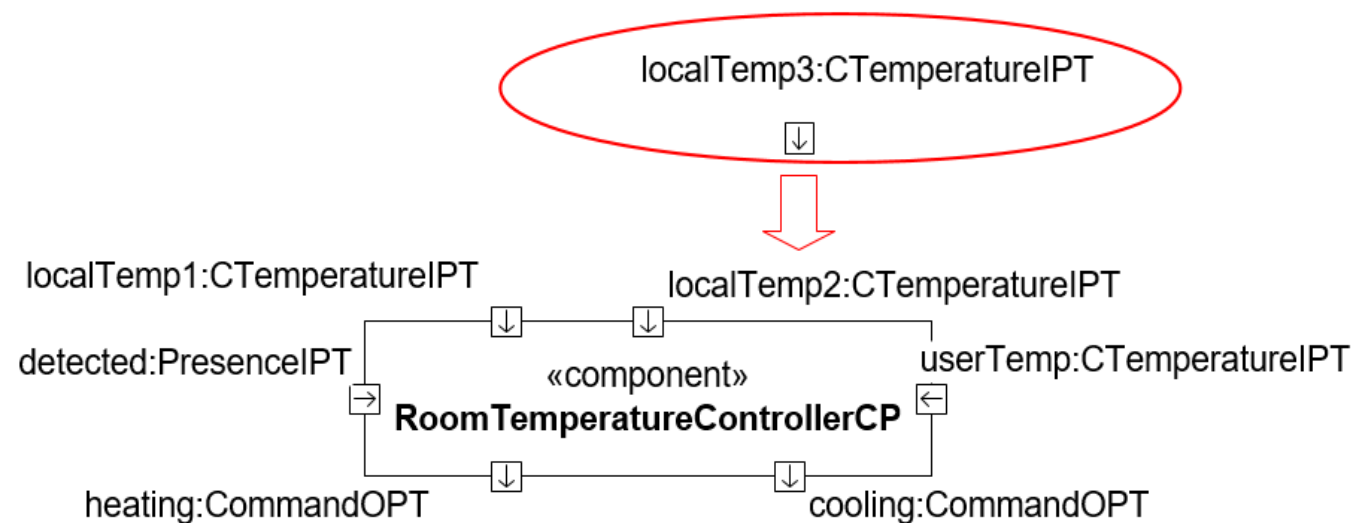
Quantifying the modification

- Considering the previous analysis, we have 9 modifications:
 - 7 **add actions**
 - 2 **replace actions**.
- Summarizing
 - the cause C1 states that **to add a new temperature sensor** we need to perform the effect actions E1, E2, E3 and E4.
 - each of the effects
 - E1 states that we need **to define a new *RoomTemperatureControllerCP***.
 - In order to do it we must
 - **add a new *RoomTemperatureControllerCP*** as a subtype of *RoomTemperatureController* and
 - **add the *localTemp3:CTemperatureIPT* port** in the new *RoomTemperatureControllerCP*.

Modifiability Example

ARCH1

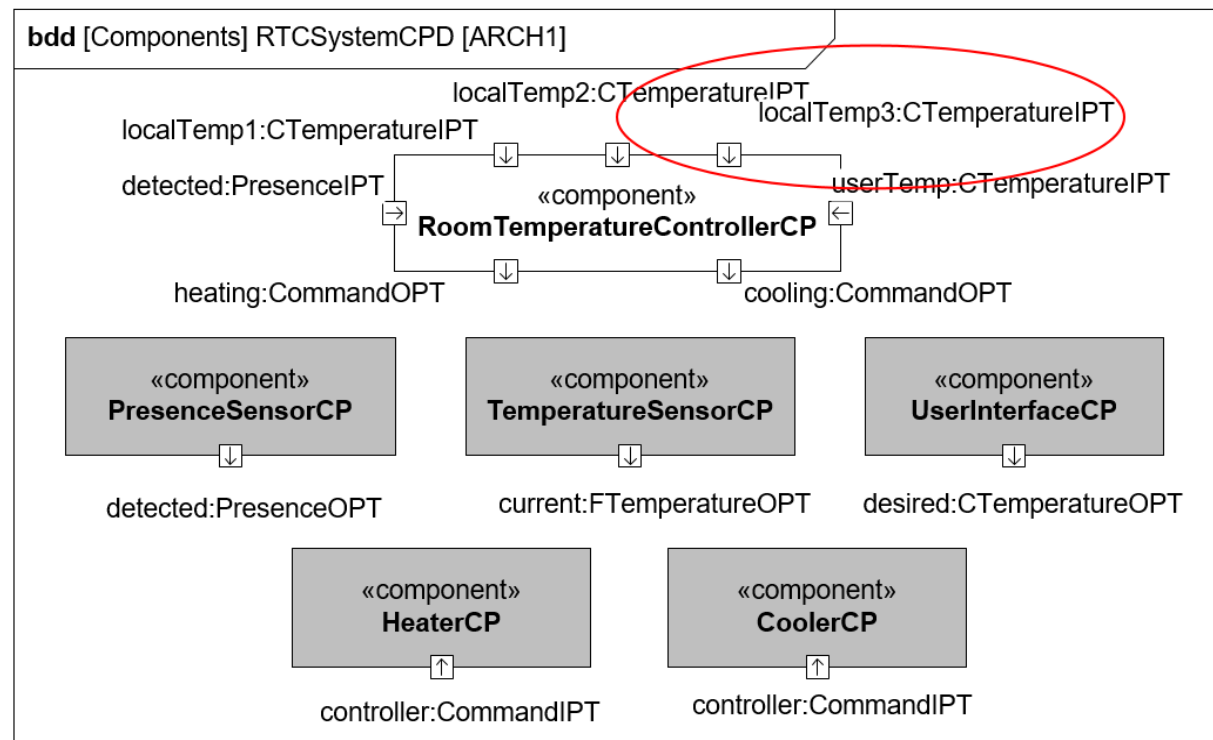
- Adding a new port to the *RoomTemperatureControllerCP* definition.



Modifiability example

ARCH1

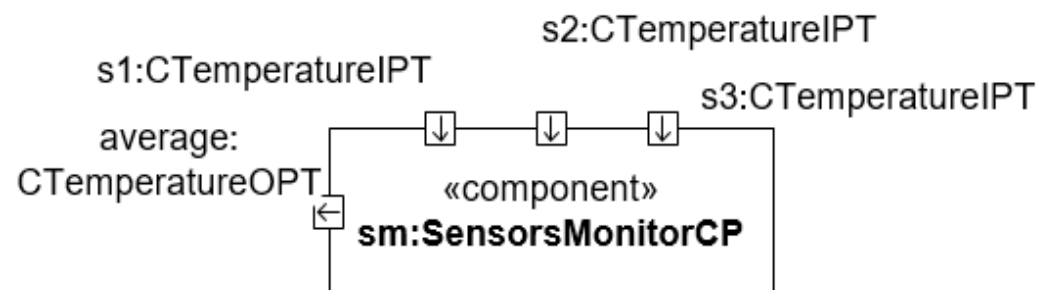
■ The new BDD showing the new port



Modifiability example

ARCH1

- E2 states that we need to **add a new *SensorsMonitorCP* component *definition***, which involves
 - adding a new *SensorsMonitorCP* as a subtype of *SensorsMonitorCP*, and
 - adding the *s3:CTemperatureIPT* port to the *SensorsMonitorCP* component definition

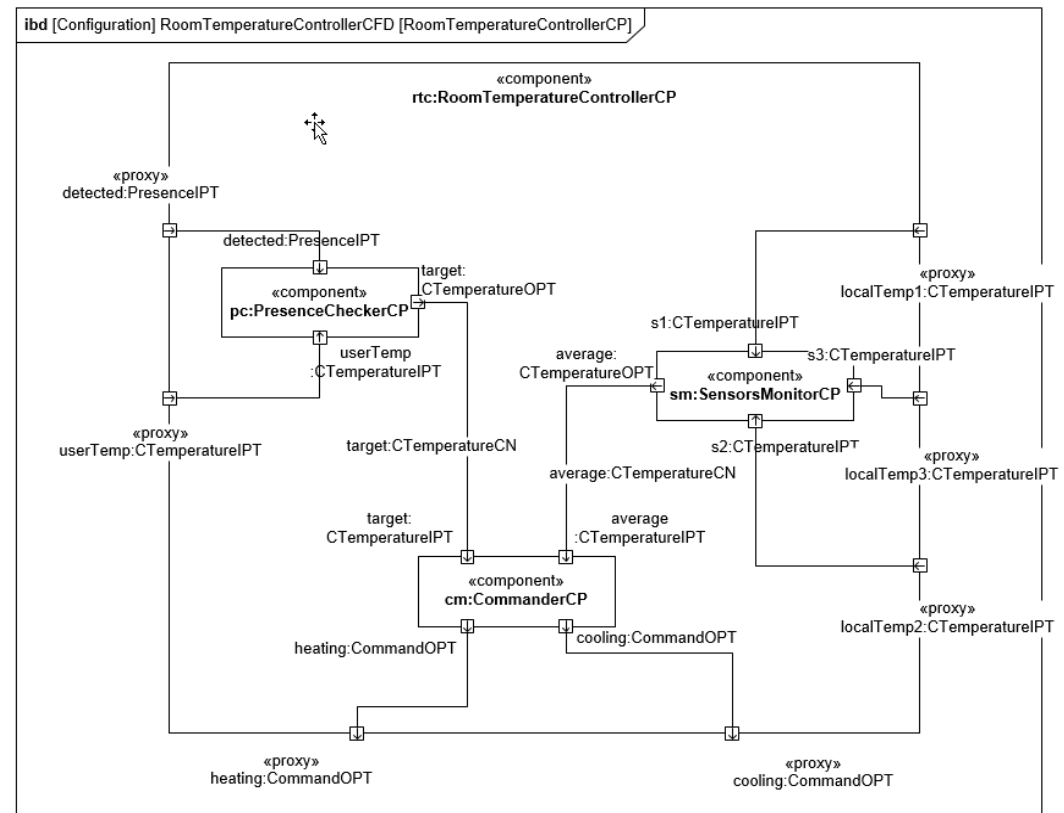


Modifiability example

29

ARCH1

- E3 states that we need to **modify the *RoomTemperatureController CP* configuration**, which is to
 - replace the *sm:SensorsMonitorCP* component by the *sm:NewSensorsMonitorCP* component, and
 - add a delegation between *s3* and *localTemp3*



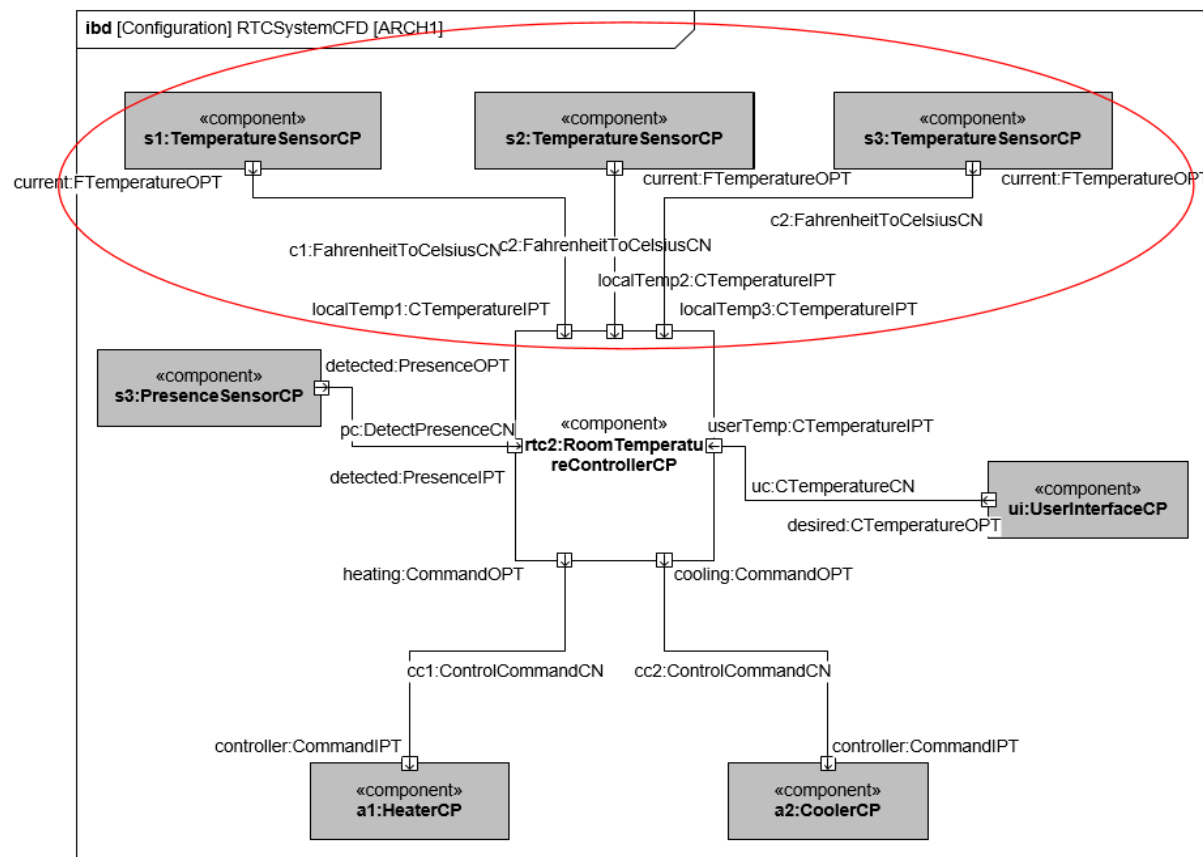
Modifiability example

ARCH1

- E4 states that we need **to modify the *ARCH1* configuration**, which is to
 - replace *rtc:RoomTemperatureControllerCP* by *rtc2:NewRoomTemperatureControllerCP*;
 - add the *s3:TemperatureSensorCP* component in *ARCH1* configuration;
 - add *c3:FahrenheitToCelsiusCN* between *localTemp3:CTemperatureIPT* of *rtc:RoomTemperatureControllerCP* and *current:FTemperatureOPT* of *s3:TemperatureCP*
- The resulting configuration in next slide

Modifiability example

The new configuration of ARCH1



Applying the Modifiability Tactics

Designing a new architecture – ARCH2

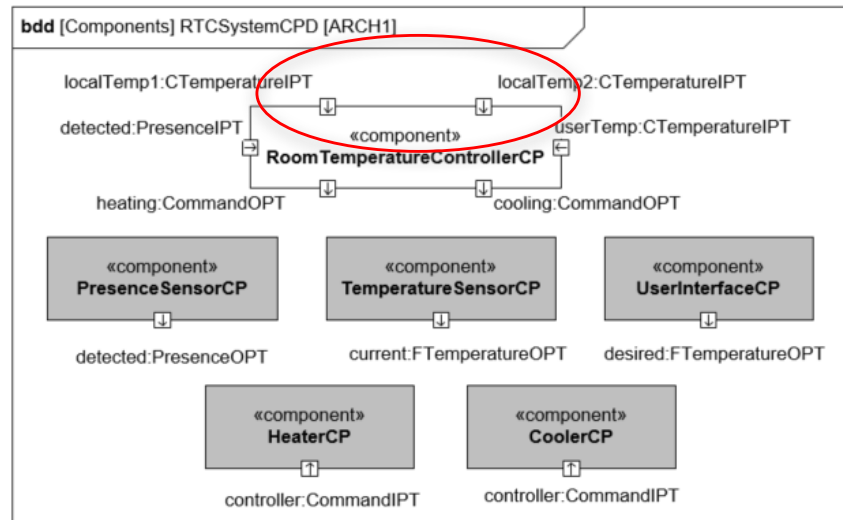
- To minimize the ripple effect (T2 modifiability tactic) we **redesign the architecture of RTC System to be more modifiable** (cheaper to modify) than ARCH1.
 - we use a **multiplex port** in the *SensorMonitorCP* component
 - using multiplex port **we can add new temperature sensors** without modifying the *RoomTemperatureControllerCP* component
- In the next slides we present the differences between the two architectures (ARCH1 and ARCH2)

Applying the Modifiability Tactics

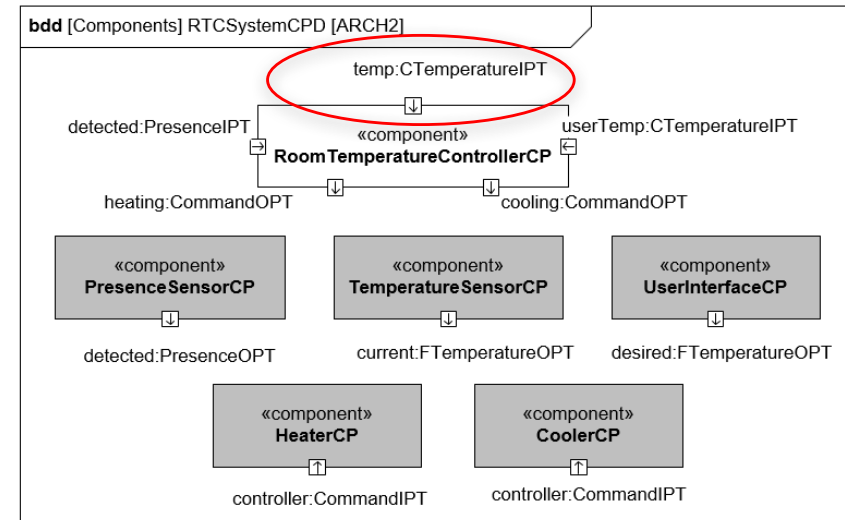
33

The Component BDDs of ARCH1 and ARCH2

ARCH1



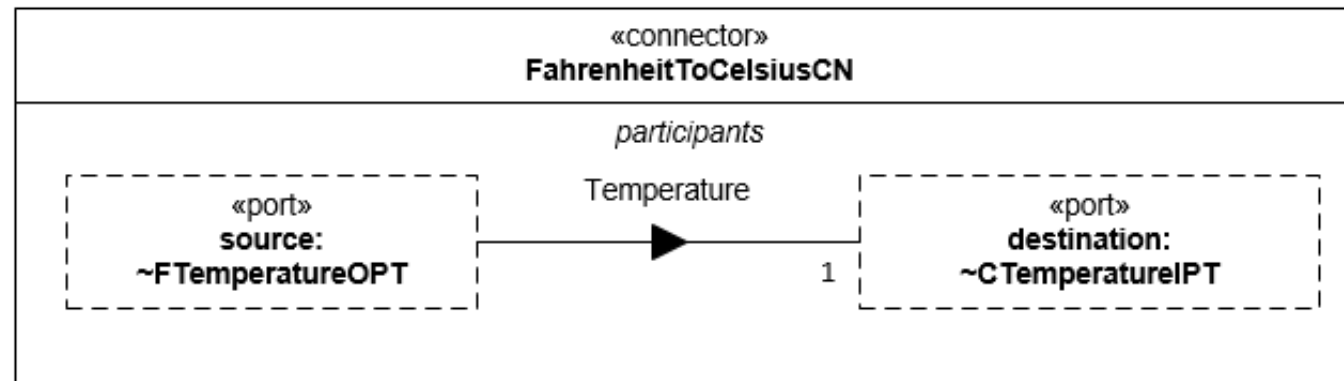
ARCH2



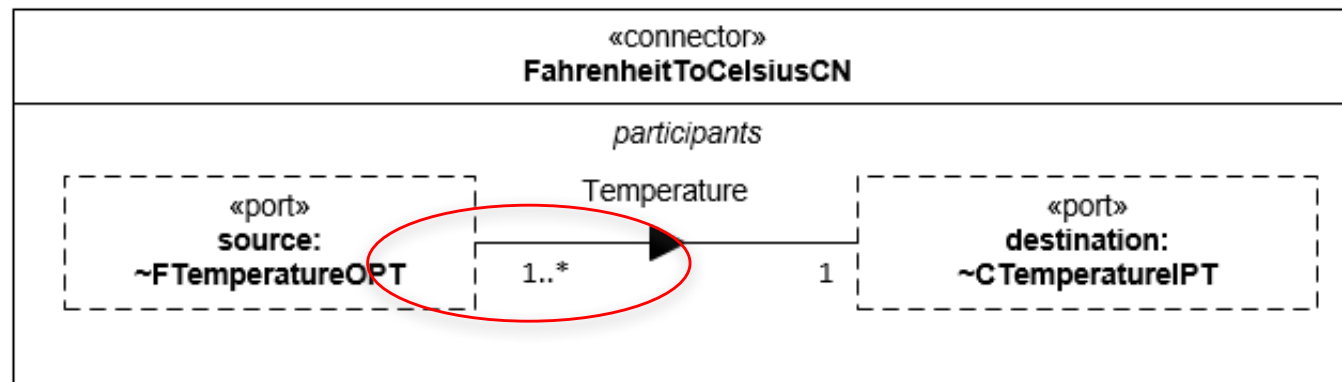
Applying the Modifiability Tactics

The Connector BDDs of ARCH1 and ARCH2

ARCH1



ARCH2

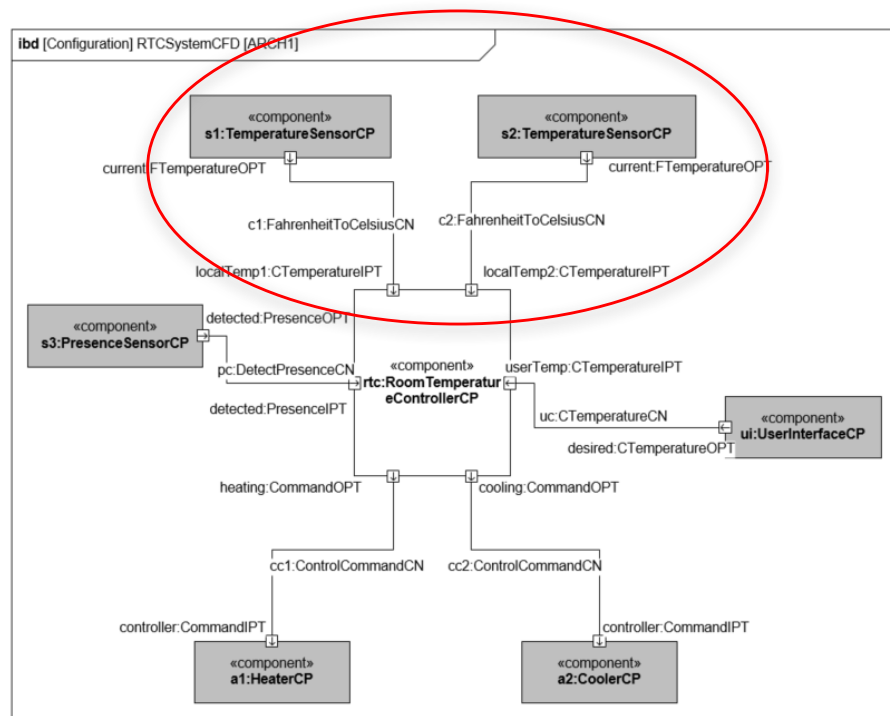


Applying the Modifiability Tactics

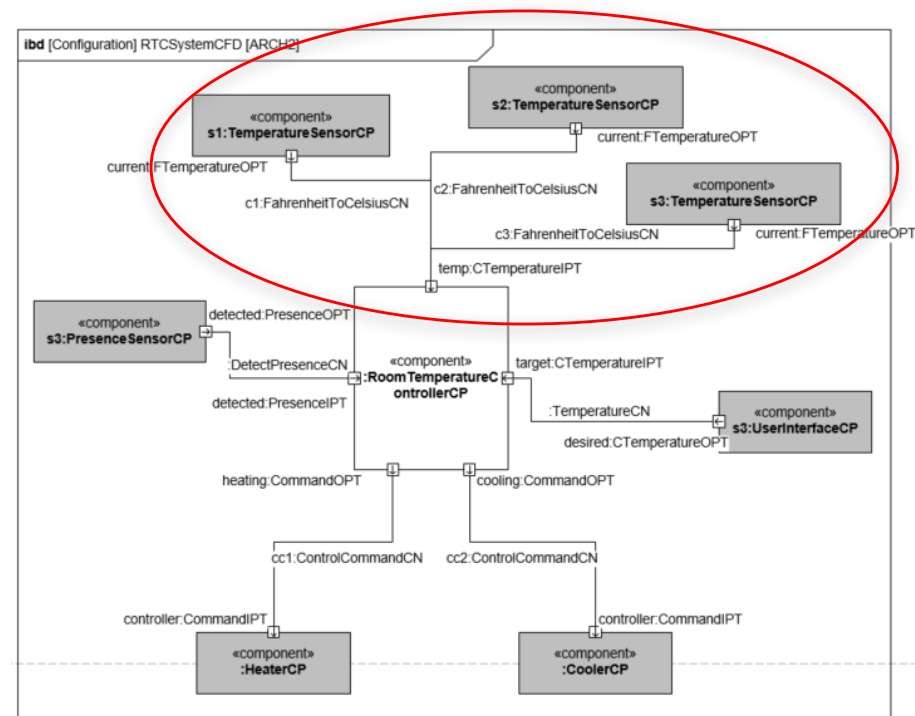
35

Two different configurations of ARCH1 and ARCH2

ARCH1



ARCH2



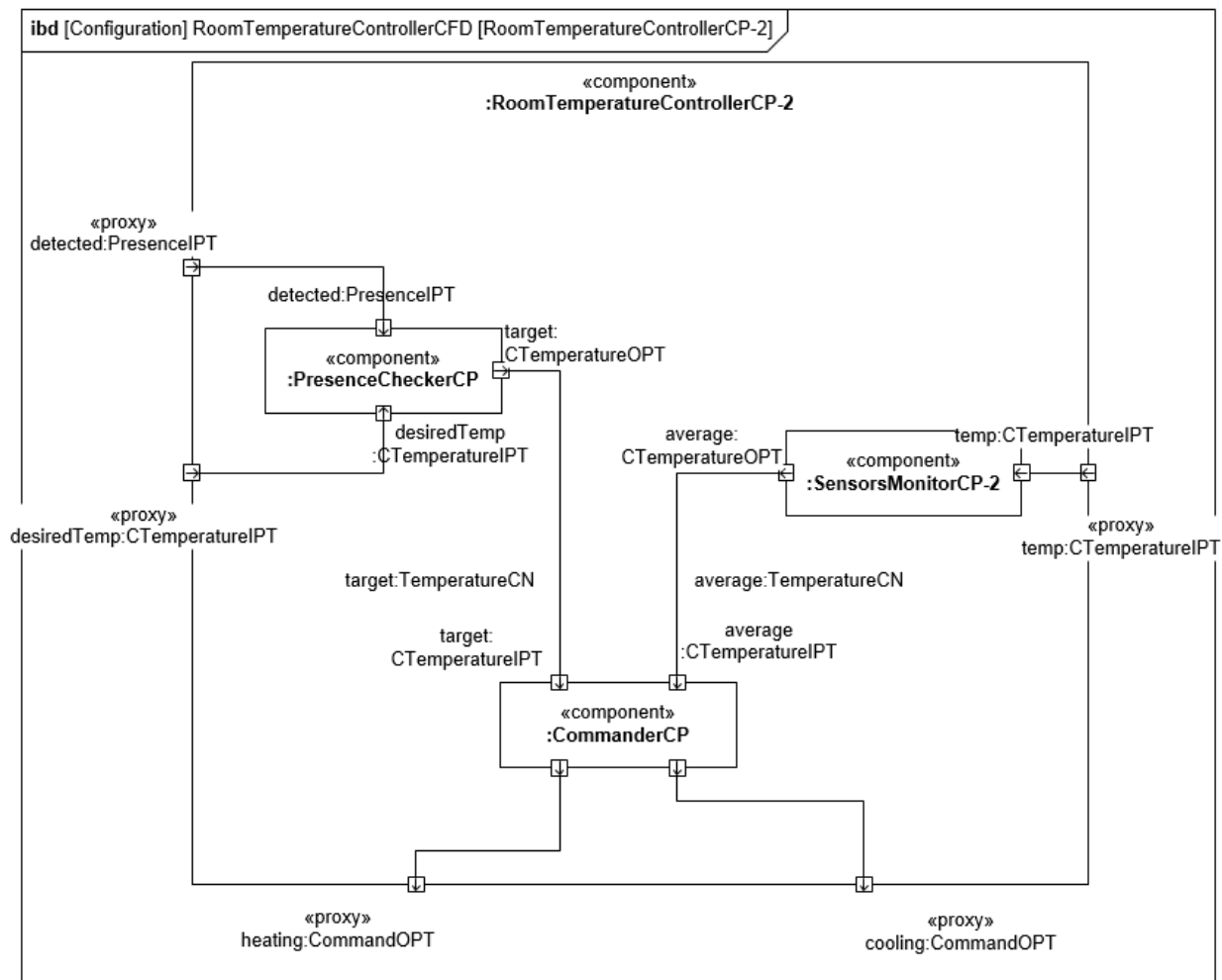
Modifiability example

Analysing the ripple effect in ARCH2

- In ARCH2, the *FahrenheitToCelsiusCN* connector supports connection from 1 or more *FTemperatureOPT* ports to 1 *CTemperatureIPT*.
- In the ARCH2 definition, we have a *RoomTemperatureController* CP with just one *CTemperatureIPT* Port.
- We need now to depict the effects to cause C1, considering ARCH2:
 - C1 – add a new temperature sensor component of a defined type to improve accuracy requires to
 - E1 - modify the architecture description (*ibd configuration*)
 - add *s3:TemperatureSensorCP* in ARCH2,
 - add *c3: FahrenheitToCelsiusCN* between the *localTemp3:CTemperatureIPT* of *rtc:RoomTemperatureControllerCP* and *current:CTemperatureOPT* of *s3:TemperatureSensorCP*.

Modifiability example

The resulting configuration of ARCH2



Analysis

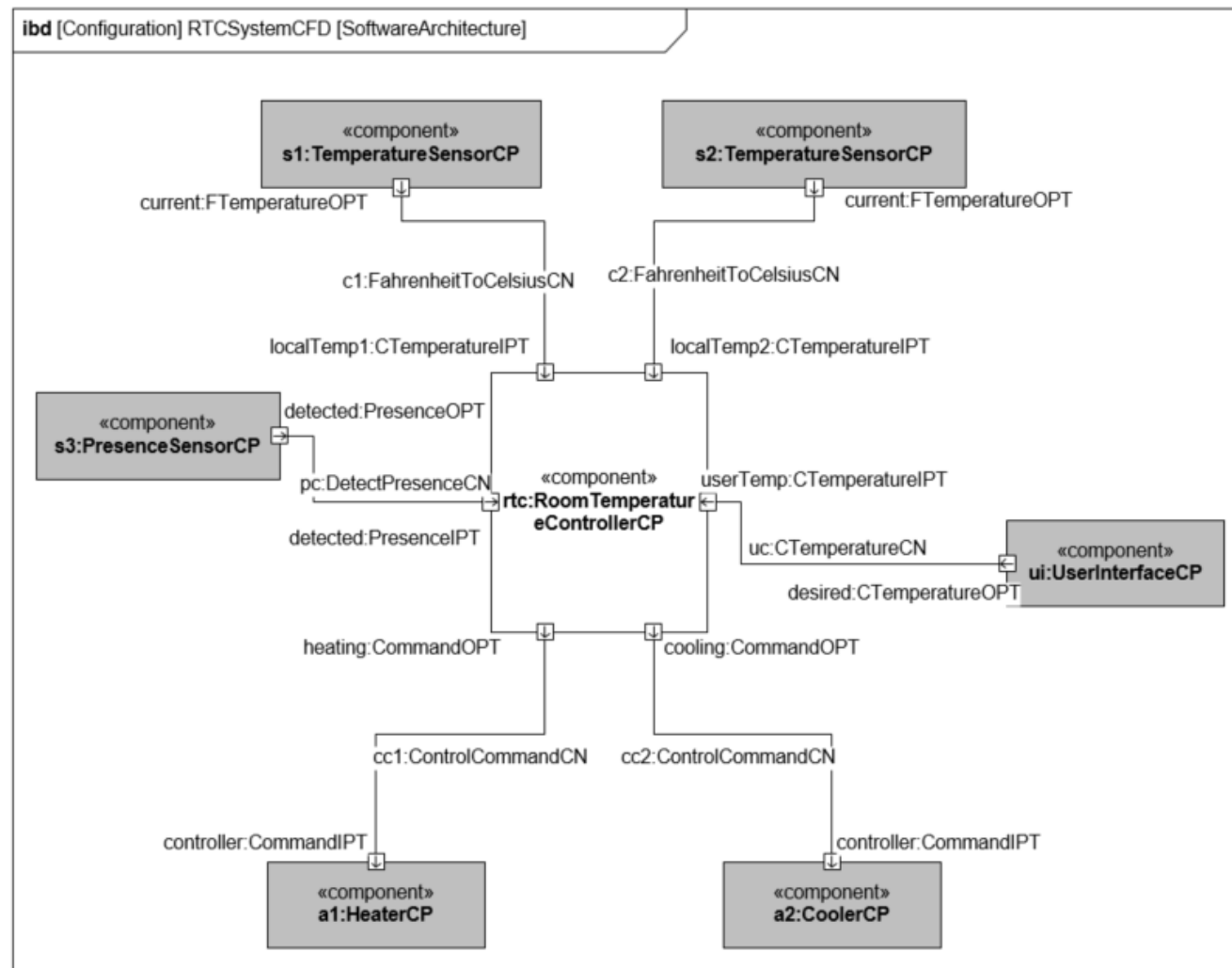
Comparing ARCH1 and ARCH2

- To add a new sensor in ARCH1, the Ripple effect determines that we need to do 9 modifications
 - Quantification of the modification quality attribute in ARCH 1
 - 9 modifications (7 add and 2 replace)
- The same modification in ARCH2 requires only 2 modifications
 - Quantification of the modification quality attribute in ARCH2
 - 2 modifications (2 add)
- Therefore, ARCH2 is more modifiable than ARCH1



Modifiability Example

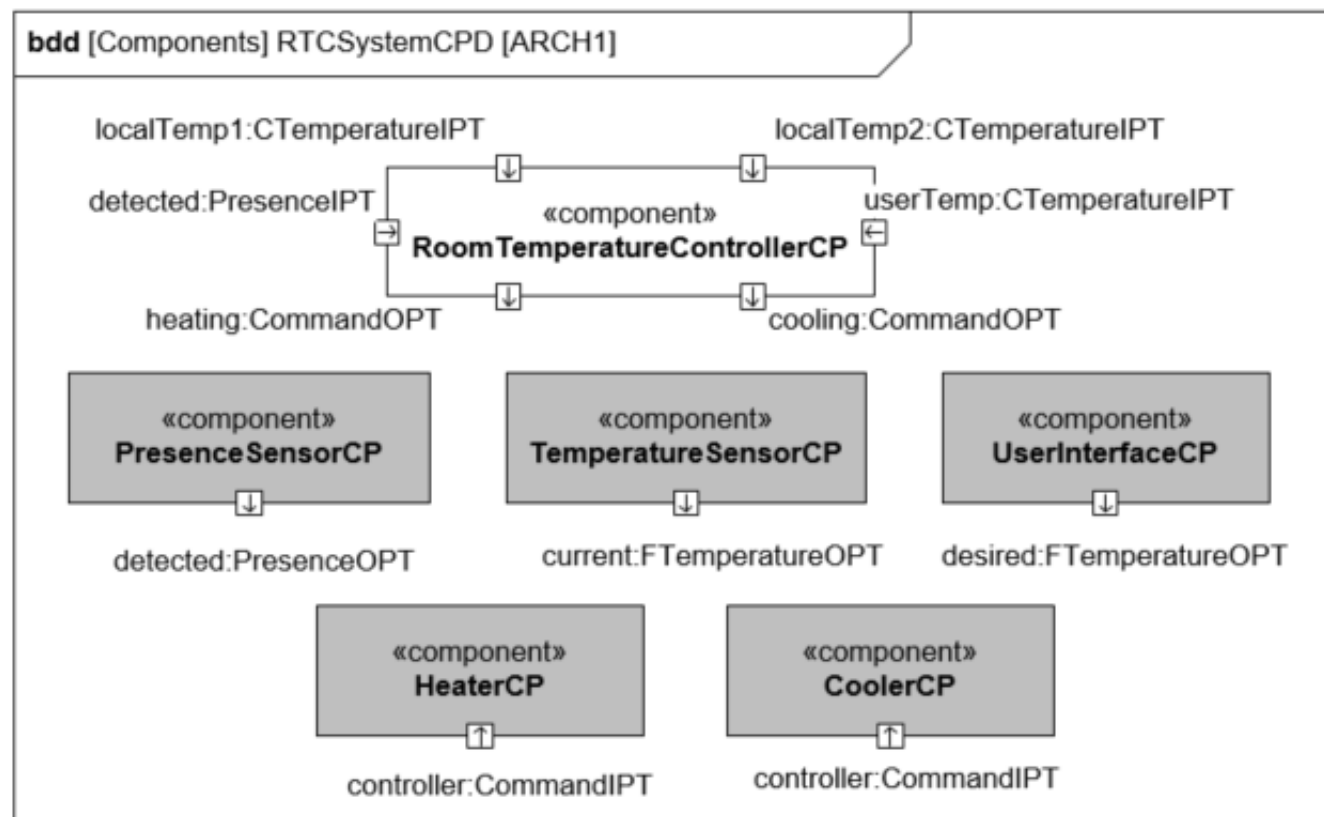
Example – RTC System ARCH1



Example – add a new sensor

RTC System ARCH1 – the bdd of the components before the modification

- modify the RoomTemperatureControllerCP component definition

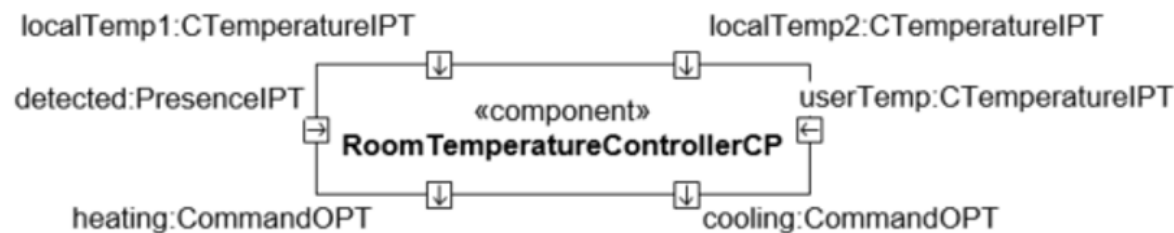


Example – add a new sensor

RTC System ARCH1 – the modification

- modify the RoomTemperatureControllerCP component definition, which requires to (AND)
 - add the localTemp3:CTemperatureIPT port in the RoomTemperatureControllerCP component type

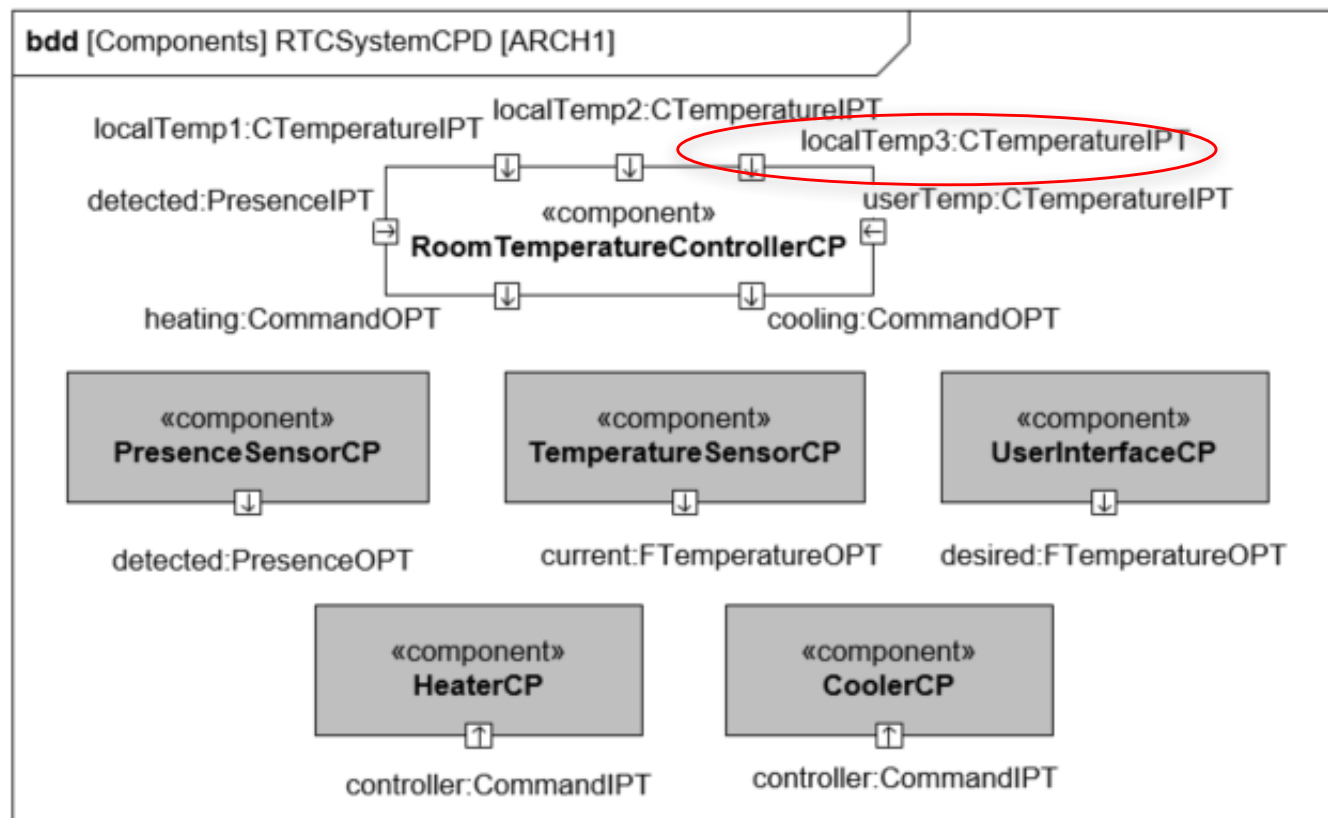
localTemp3:CTemperatureIPT



Example – add a new sensor

RTC System ARCH1 – the components bdd after modification

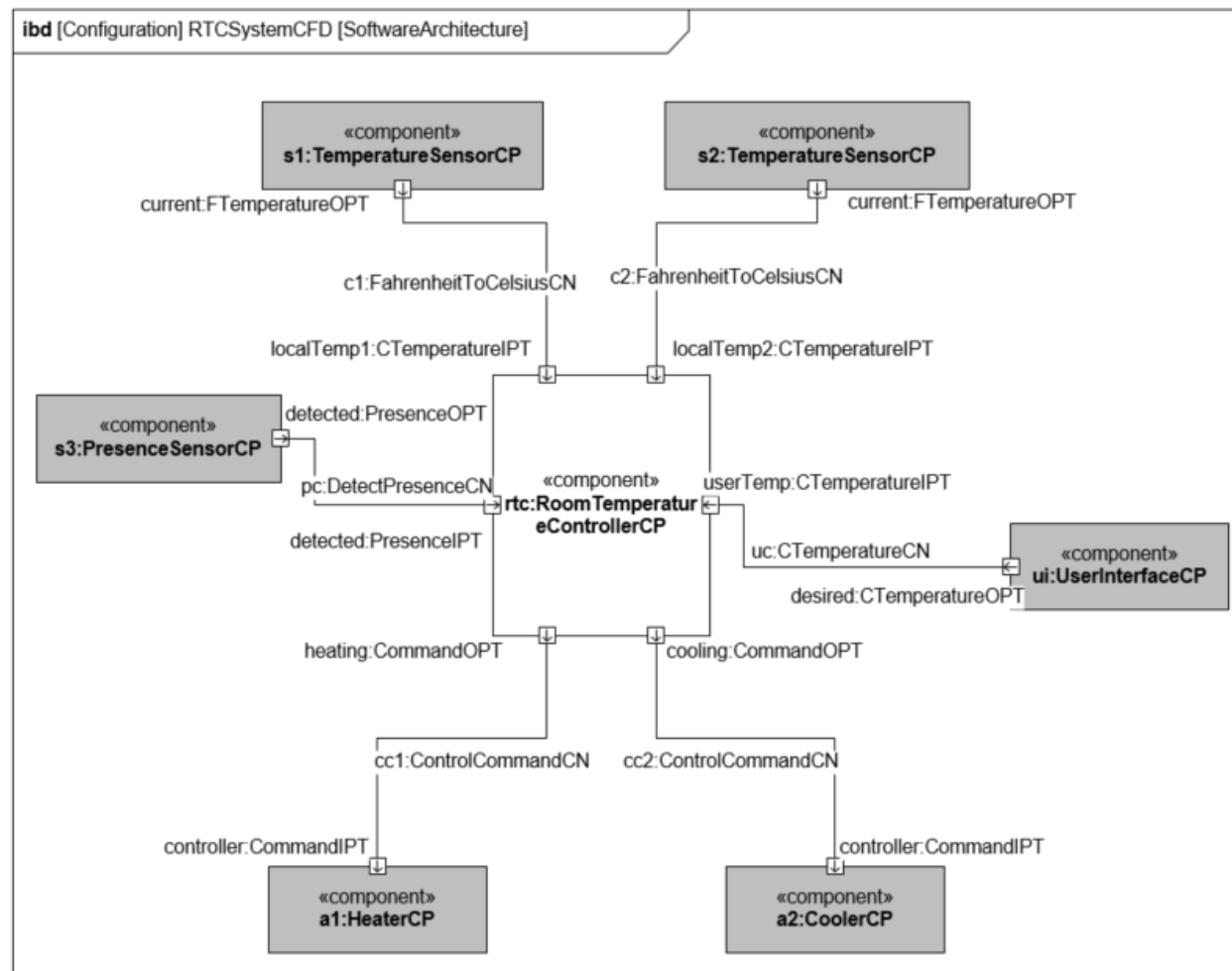
- modify the RTC System ARCH1 configuration



Example – add a new sensor

RTC System ARCH1 – ibd before the modification

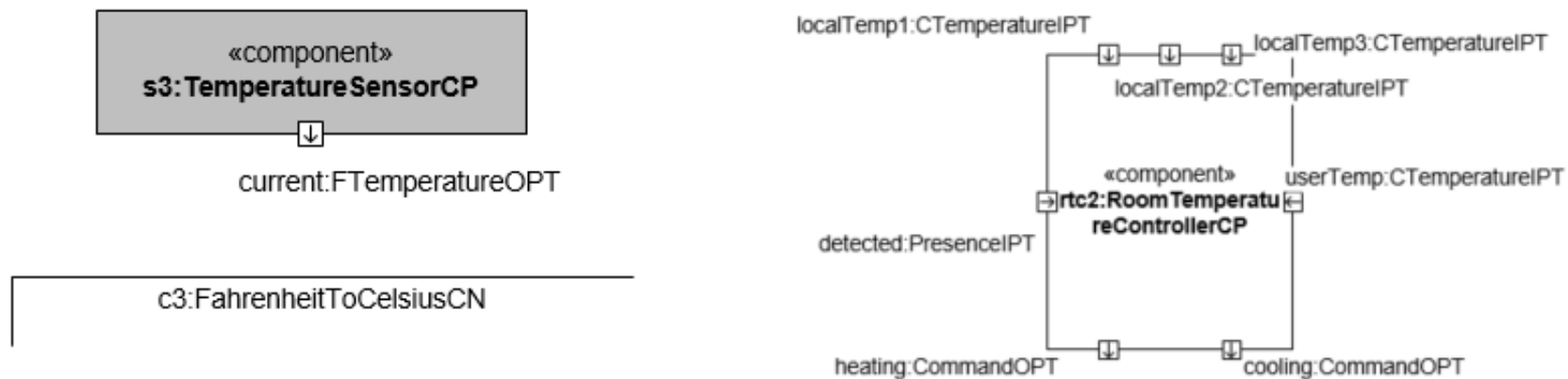
- modify the RoomTemperatureControllerCP configuration



Example – add a new sensor

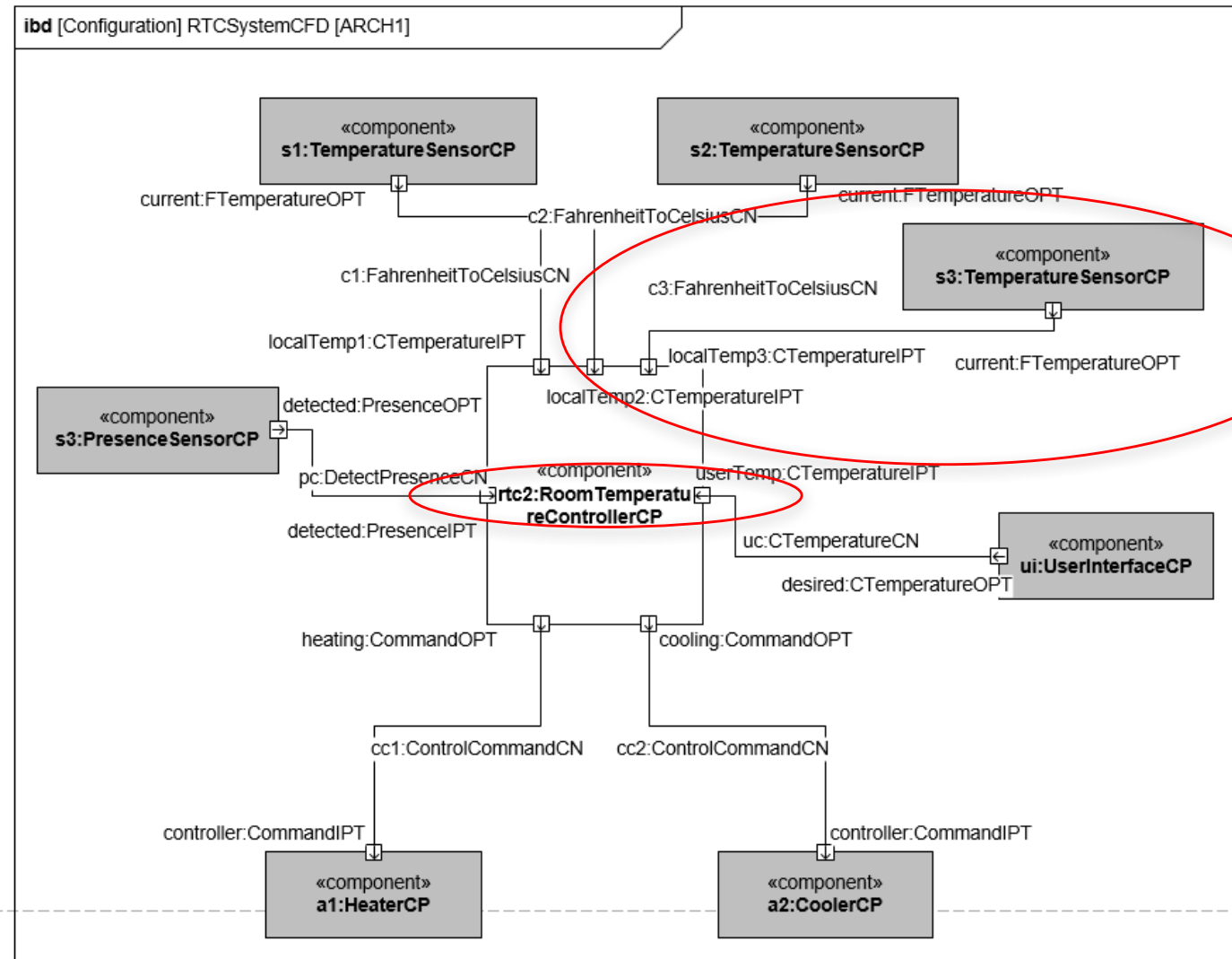
RTC System ARCH1 – the modifications in the ibd

- replace rtc:RoomTemperatureControllerCP by rtc2:RoomTemperatureControllerCP
- add the s3:TemperatureSensorCP component in ARCH1 configuration
- add c3:FahrenheitToCelsiusCN between localTemp3:CTemperatureIPT of rtc:RoomTemperatureControllerCP and current:FTemperatureOPT of s3:TemperatureCP

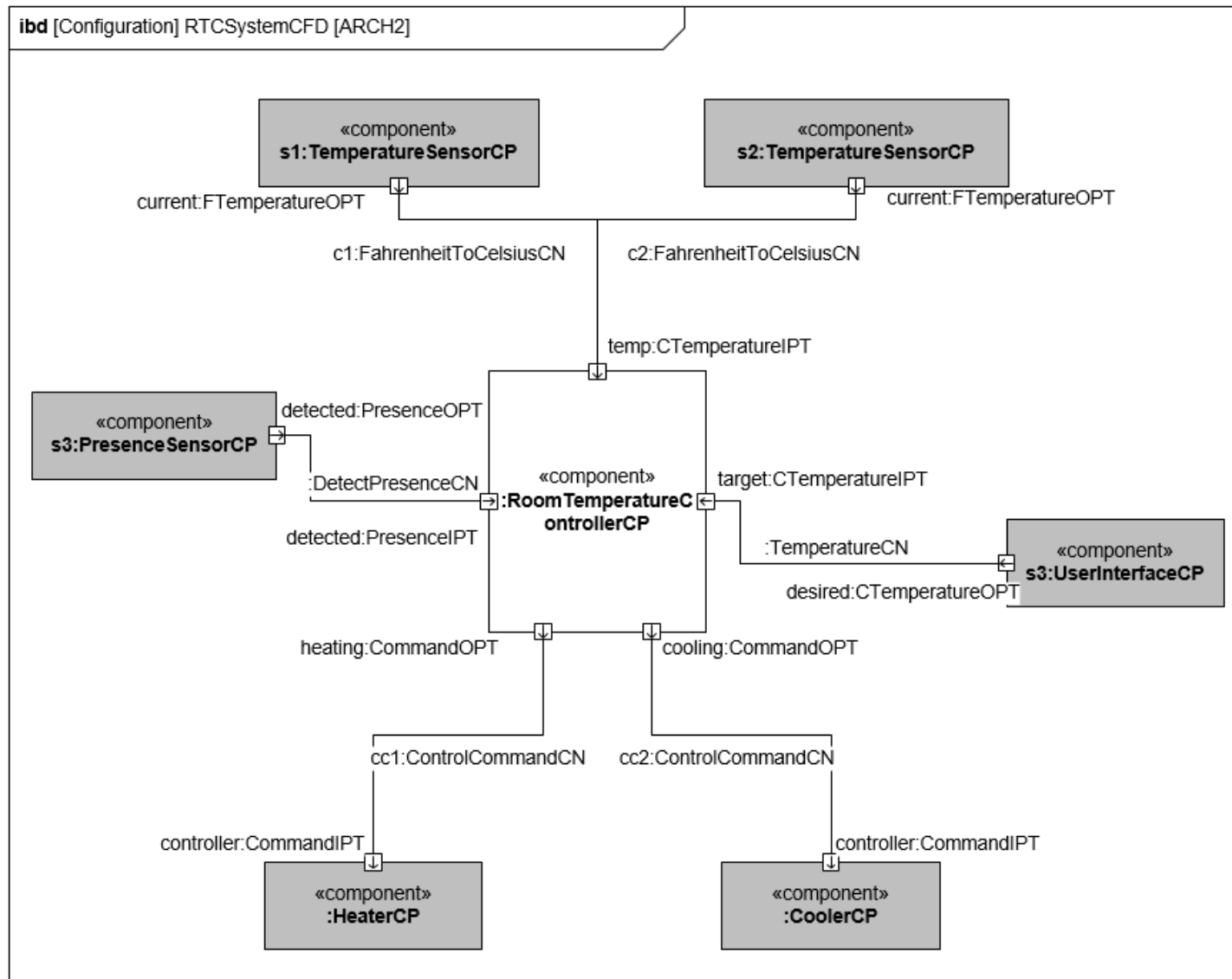


Example – add a new sensor

RTC System ARCH1 – the ibd after the modification



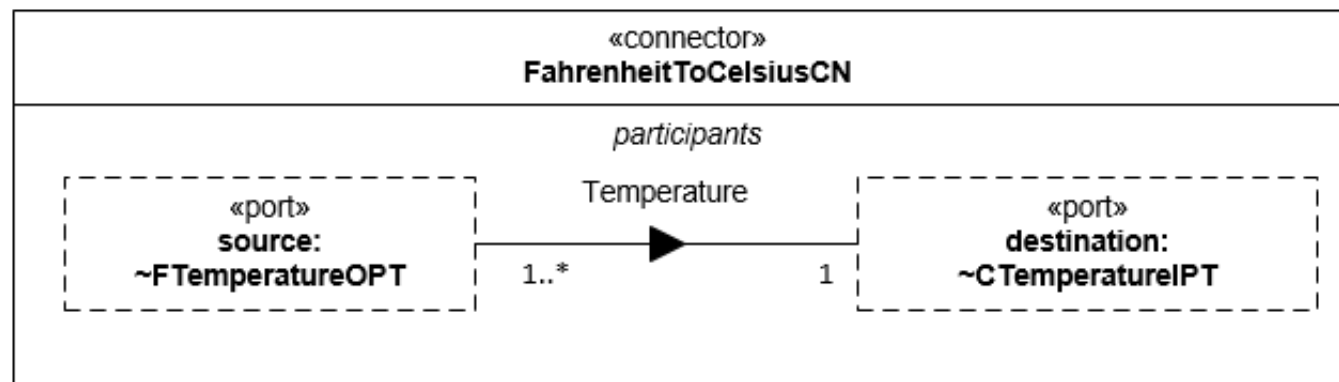
Example – RTC System ARCH2



Example – RTC System ARCH2

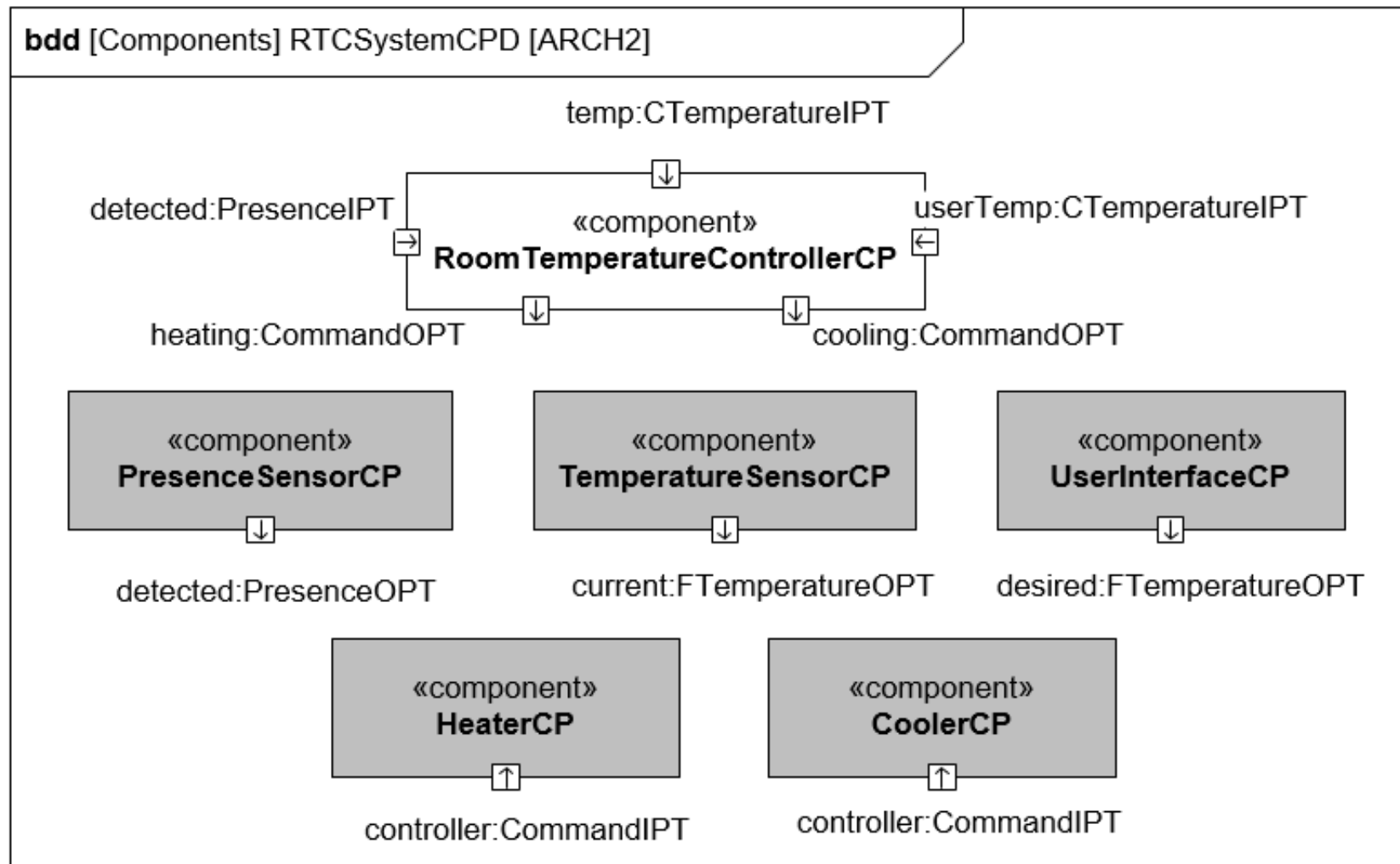
The FahrenheitToCelsiusCN connector

- In ARCH2, the FahrenheitToCelsiusCN connector supports connection from 1 or more FTemperatureOPT ports to 1 CTemperatureIPT



Example – RTC System ARCH2

the bdd components



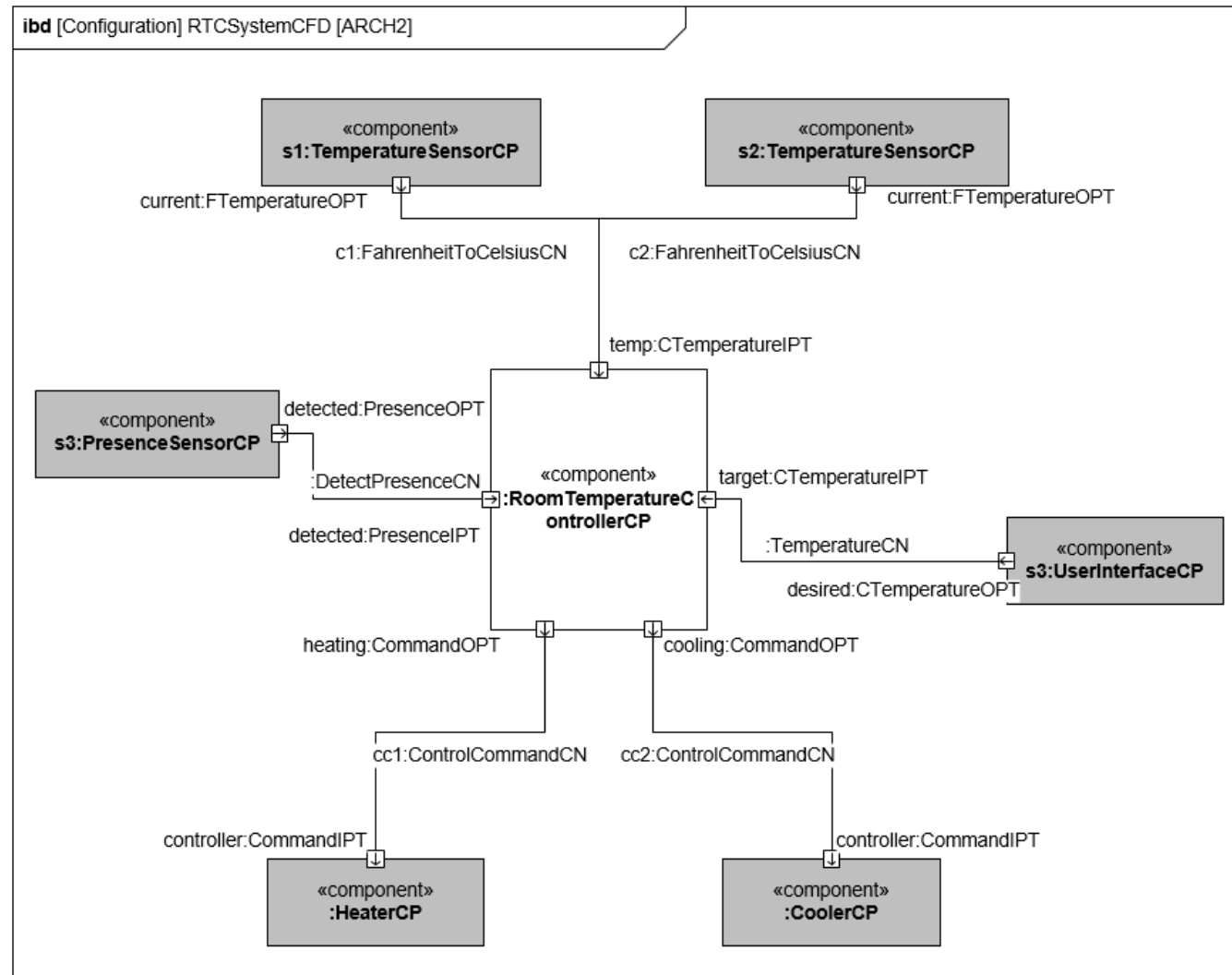
Modifiability example

RTC System ARCH2 – effects

- Here we depict the effects to cause C1, considering the ARCH2
- C1 – add a new temperature sensor component of a defined type to improve accuracy requires to (AND)
 - E1 - modify the architecture description (ibd/configuration)
 - add s3:TemperatureSensorCP in RTCSystem ARCH2
 - add c3: FahrenheitToCelsiusCN between the localTemp3:CTemperatureIPT of rtc:RoomTemperatureController CP and current:CTemperatureOPT of s3:TemperatureSensorCP
- Quantifying the modifications
 - 2 modifications (2 add primitives)

Example – add a new sensor

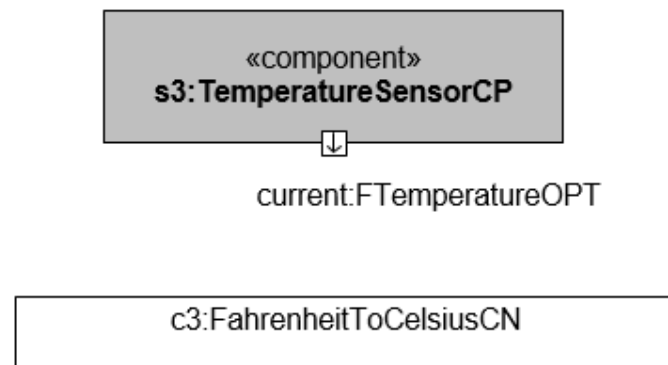
RTC System ARCH2 – ibd before the modification



Example – add a new sensor

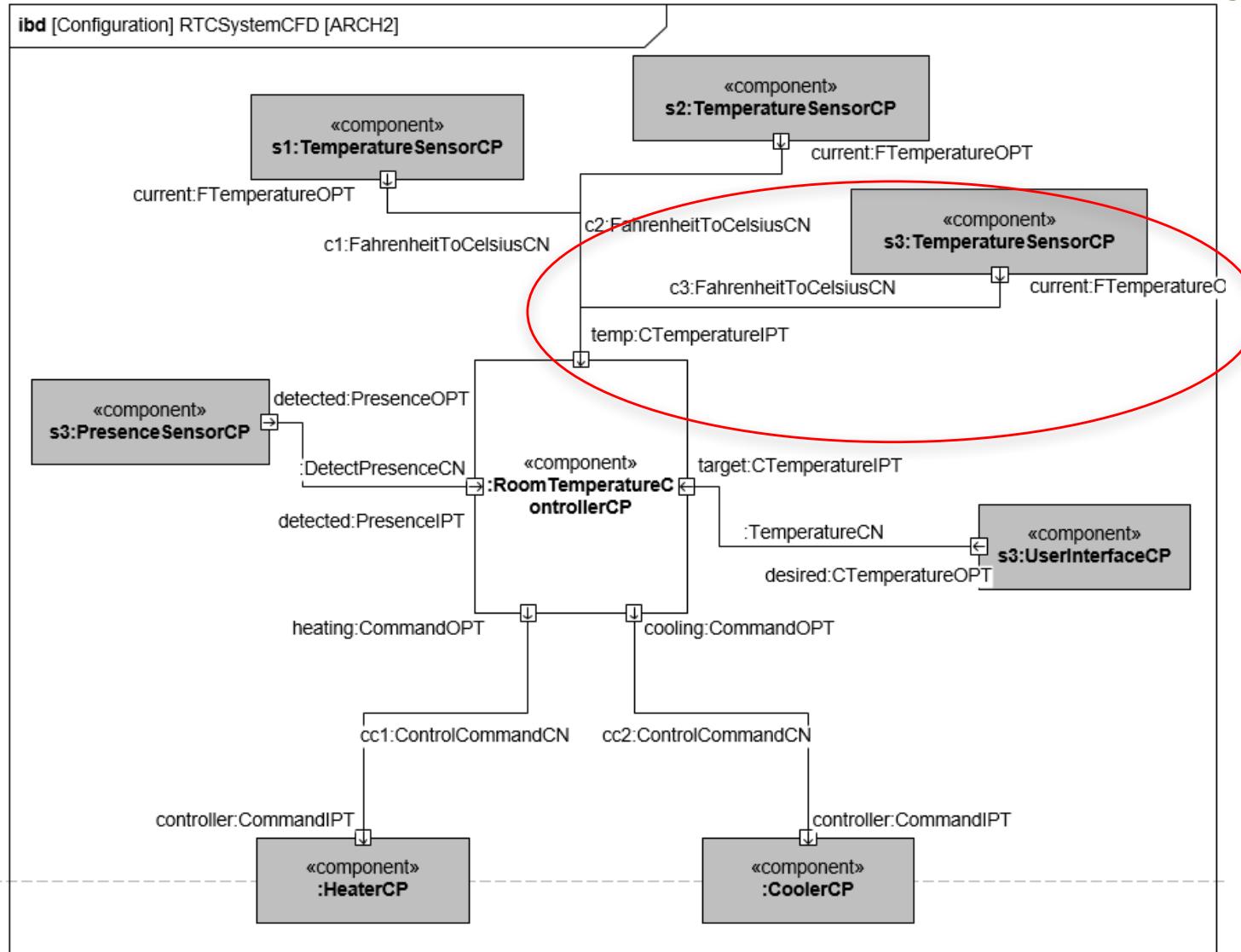
RTC System ARCH2 – the modifications in the ibd

- modify the architecture description (ibd/configuration)
 - create a new s3:TemperatureSensorCP component
 - create a new c3:FahrenheitToCelsiusCN connector between the localTemp3:CTemperatureIPT and current:CTemperatureOPT of s3.



Example – add a new sensor

RTC System ARCH2 – the ibd after the modification



Summary

- In this chapter you learnt
 - the modifiability concept
 - the architectural causes and effects of modifiability
 - tactics to improve modifiability
 - a taxonomy of modifiability primitives
 - a comparison technique to evaluate the modifiability in alternative architectures
- You learnt how to
 - express modifiability in software architecture using a cause-effect relationship
 - compare two alternative architectures to evaluate their modifiability by quantifying the ripple effect quality attribute

For Further Reading

- Bass, L., Clements, P., Kazman, R.: Software Architectures in Practice, 2nd edn. Addison Wesley, Reading (2003)
- Clements, P., Bachmann, L., Garlan, D., Ivers, J., Little, R., Merson, P., Nord, R.: Documenting Software Architecture: Views and Beyond. SEI Series in Software Engineering (2003)