

Software Architecture in Action

Flavio Oquendo, Jair C Leite, Thais Batista



Chapter 5

Specifying Behavior of Software Architectures

Learning outcomes of this chapter

■ You will learn:

- the architectural constructs to express behavior: activities, actions, interactions;
- the auxiliary constructs to define activities and actions: equations for defining actions and protocols controlling interactions;
- the SysADL notation to represent these constructs in terms of diagrams.

+ The structure of this chapter

- Introduction
- Behavioral viewpoint and views
 - Behavioral viewpoint
 - Behavioral views
- Behavioral Constructs
 - Activity
 - Action
 - Equations in the action definition
 - Relating definition of activities and actions
 - Relating components and connectors with activities and actions
- Diagrams for behavioral views
 - Block definition diagrams
 - Activity diagrams
 - Parametric diagrams
- Relating structural and behavioral viewpoints
- Describing the architecture from the behavioral viewpoint
- Summary
- For further reading



Behavioral viewpoint and views

Behavioral Viewpoint

Conceptual overview

- The *Behavioral Viewpoint* is concerned with the specification of the behavior of the elements described in the Structural viewpoint
- It specifies behaviors by using the activity and parametric diagrams
- The Behavioral specification includes both the definition and the use of how software architecture elements perform activities to consume and produce data
 - the activities and actions of components and connectors
 - the protocol of ports of components and connectors
 - the equations for defining the semantics of activities and actions

Behavior

Conceptual overview – 1/2

- The behavior of a software architecture refers to the way the elements performs activities and interactions to achieve the required system functionality
 - component (simple or composite), connectors, and configurations have a behavior
 - component, connectors, and configurations have ports that define interactions behavior (protocols)
- An example of a software behavior is the computation the average temperature in the room

Behavior

Conceptual overview – 2/2

- A component behavior represents its functional behavior
- A connector behavior represents the interaction between components
- A configuration behavior represents the composition among components and connectors
- Each behavior is expressed by an activity
 - an activity is expressed by actions
 - actions are internal actions or interaction actions

Behavioral views

Conceptual overview

- Using the SysADL constructs of the behavioral viewpoint, we can define different behavioral views
- A *behavioral view* shows a subset of the behavior description of a software architecture
 - views describe the definitions of activities, actions, equations, and protocols that are used in the definition of the architectural behavior
 - views describe how the defined activities, actions, equations, and protocols are used and composed to define the architectural behavior, which achieves the required system functionality



Behavioral constructs

Behavioral Constructs

- In SysADL we specify the behavior of software architecture using
 - activities – to specify the overall behavior of structural elements
 - actions – to specify elementary behaviors
 - constraints – to specify equations



Activity

Activity

Conceptual overview – 1/2

- An activity represents the behavior of a software architecture element by expressing
 - the consumption and production of data via *in* and *out* pins
 - the equations – invariants, pre- and post-conditions – associated to that behavior

Activity

Conceptual overview – 2/2

- An activity is executed when all *in* pins receive data, without interruption until send the result to *out* pins
 - when an activity consume data from the *in* pin, it is executed
 - when the results are sent to the *out* pins, the activity wait for their consumption
 - after the consumption, the activity is ready to receive new data
- An example of the software behavior is to calculate the average temperature using data provided by the temperature sensor

Activity

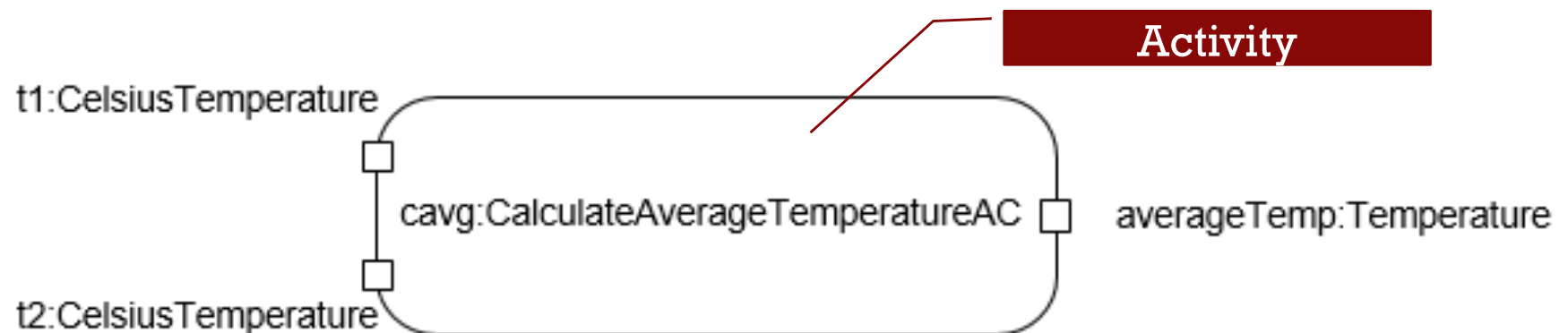
Use in SysADL

15

Activity use

- We can use an activity to specify the behavior of components, connectors, and their ports
- We need define activity types before using them
- We can allocate a activity type to components, connectors, and ports

SysADL notation



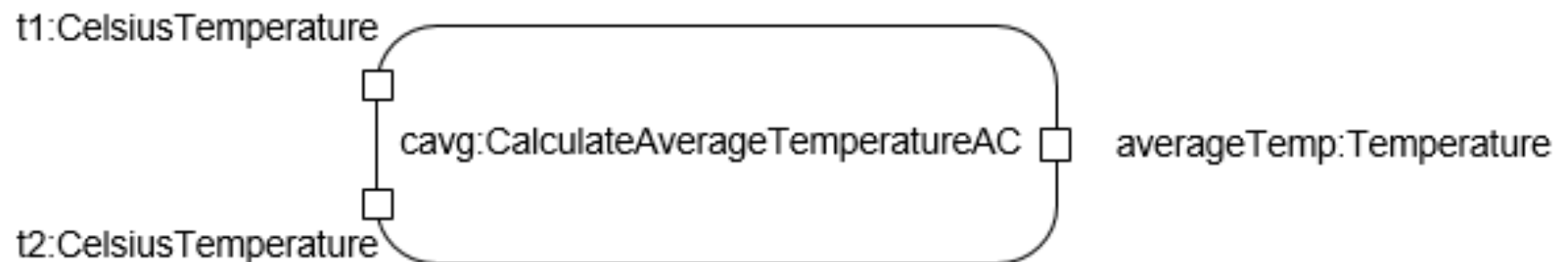
Activity

Use in SysADL – example

Example description

- We represent activities as rounded rectangles with a name and a type
- We represent parameters as small rectangles in the border of an activity
- In the example, *cavg* is a use of the *CalculateAverageTemperatureAC* type.
 - *cavg* has with two input parameters - *t1* and *t2* - and an output parameter - *average*

SysADL notation



Activity

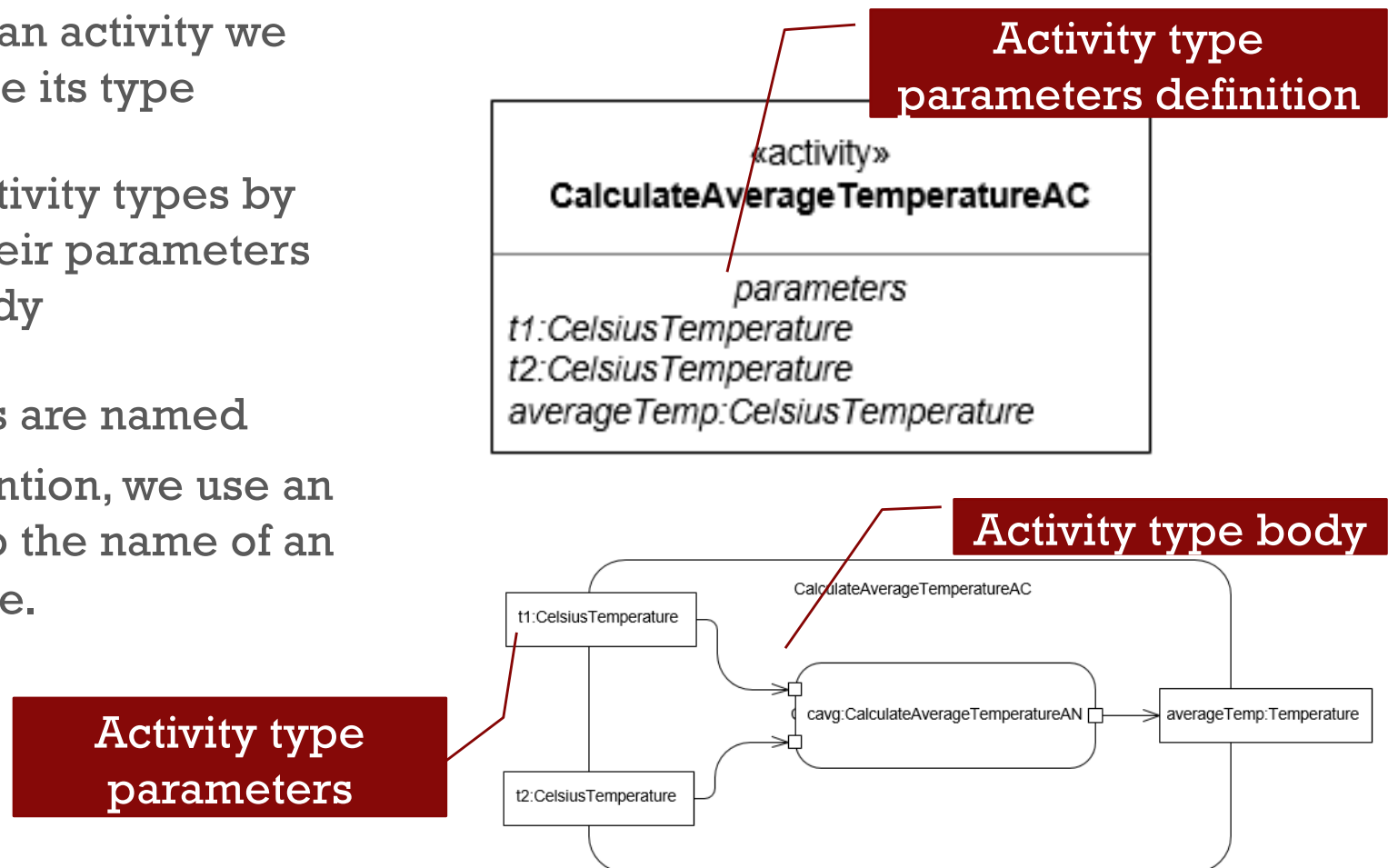
Definition in SysADL – 1/2

17

Activity definition

- Before using an activity we need to define its type
- We define activity types by specifying their parameters and their body
- Activity types are named
 - As a convention, we use an AC suffix to the name of an activity type.

SysADL notation



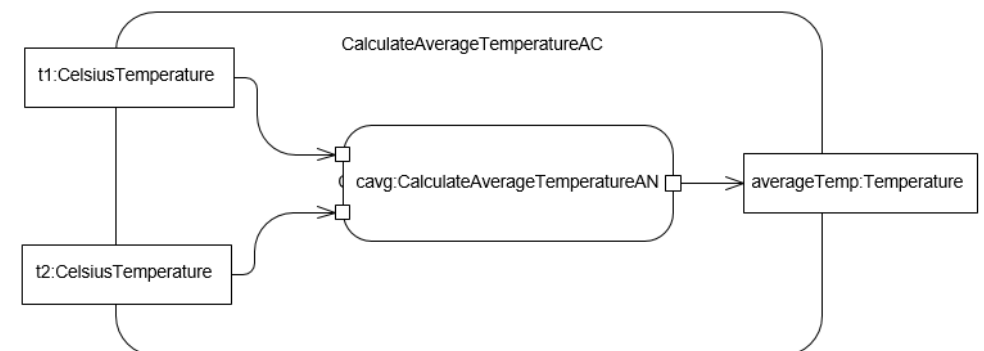
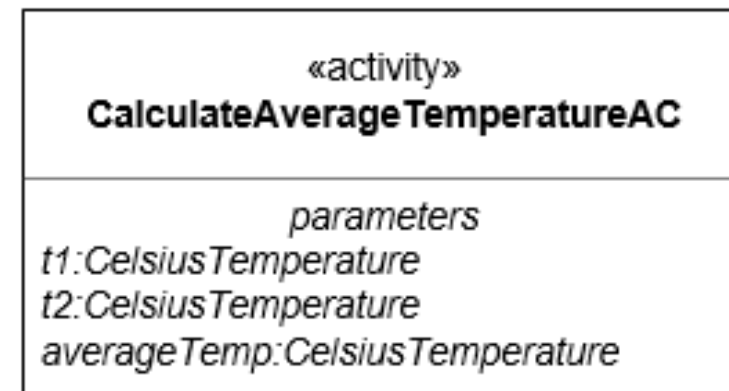
Activity

Definition in SysADL – 2/2

Notation description

- We represent an activity type using two notations
 - the <<activity>> stereotype defines the name and the parameters
 - the rounded rectangle represents the activity body
 - we also use activity diagram is a rectangle to represent the body in a model (see later)

SysADL notation



Activity

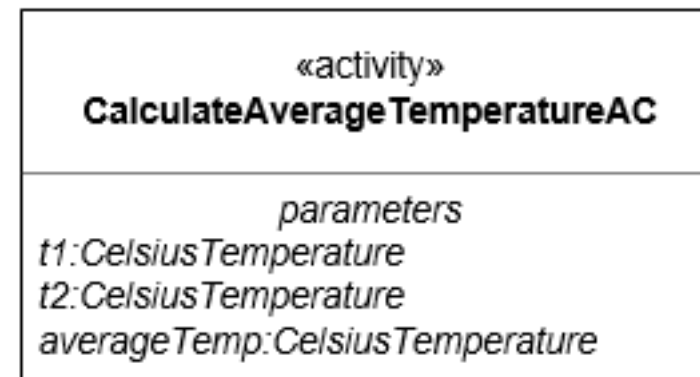
Parameters definition in SysADL

Activity parameters definition

- When we define an activity parameter we should provide its name and its type
- We need to use previously defined data or value types

SysADL Notation

- In this example we have the definition of the parameters of the *CalculateAverageTemperatureAC* activity type
- *t1*, *t2* and *averageTemp* are parameters



Activity

20

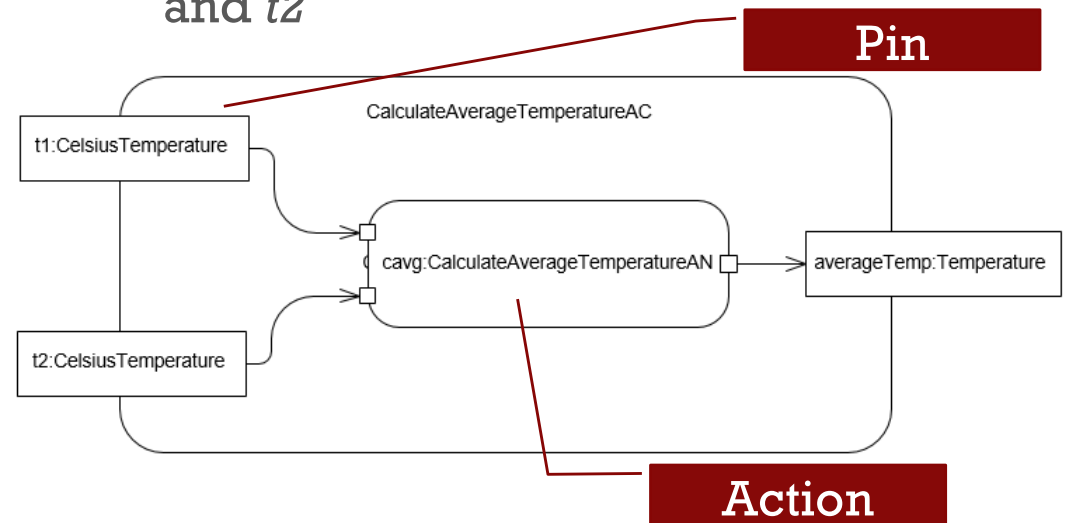
Body definition in SysADL -1/2

Activity body definition

- When we define an activity body we should provide its *pins*, *actions*, and *flows*
 - *pins* are representations of parameters
 - *in pins* represents input parameters
 - *out pins* represents output parameters
 - *Continued in next slide*

SysADL Notation

- In this example we have the definition of the body
 - it represents the calculation using *cavg*, the out pin *average* and two in pins *t1* and *t2*



Activity

21

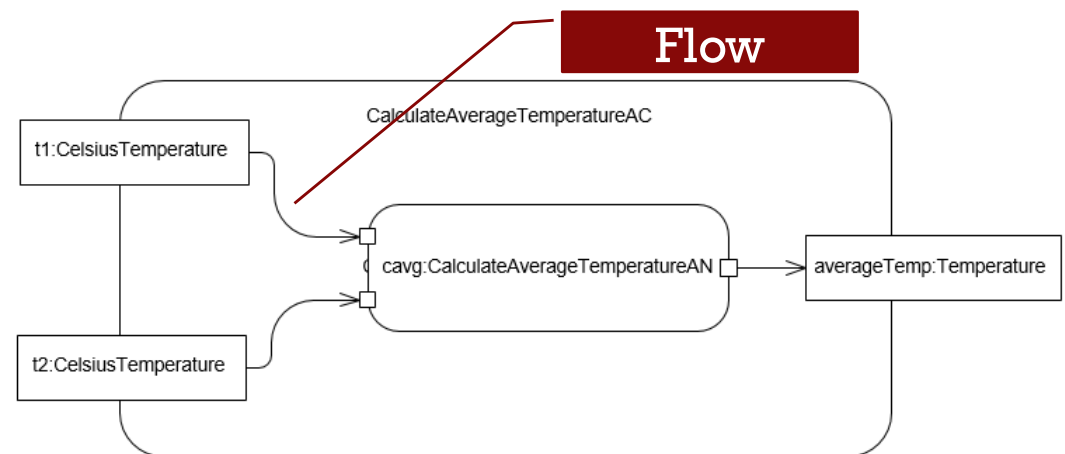
Body definition in SysADL -2/2

Activity body definition – cont.

- *flows* represent how data flows through the *actions*
 - flow arrows represent how control change from an action to another
 - flow arrows should link in and out pins appropriately
- *actions* are a simple behavior inside an activity (see later)

SysADL Notation

- In the example the data t1 e t2 flow from the in pins to the *cavg* action.
- the averageTemp flows from the *cavg* action to out pin.

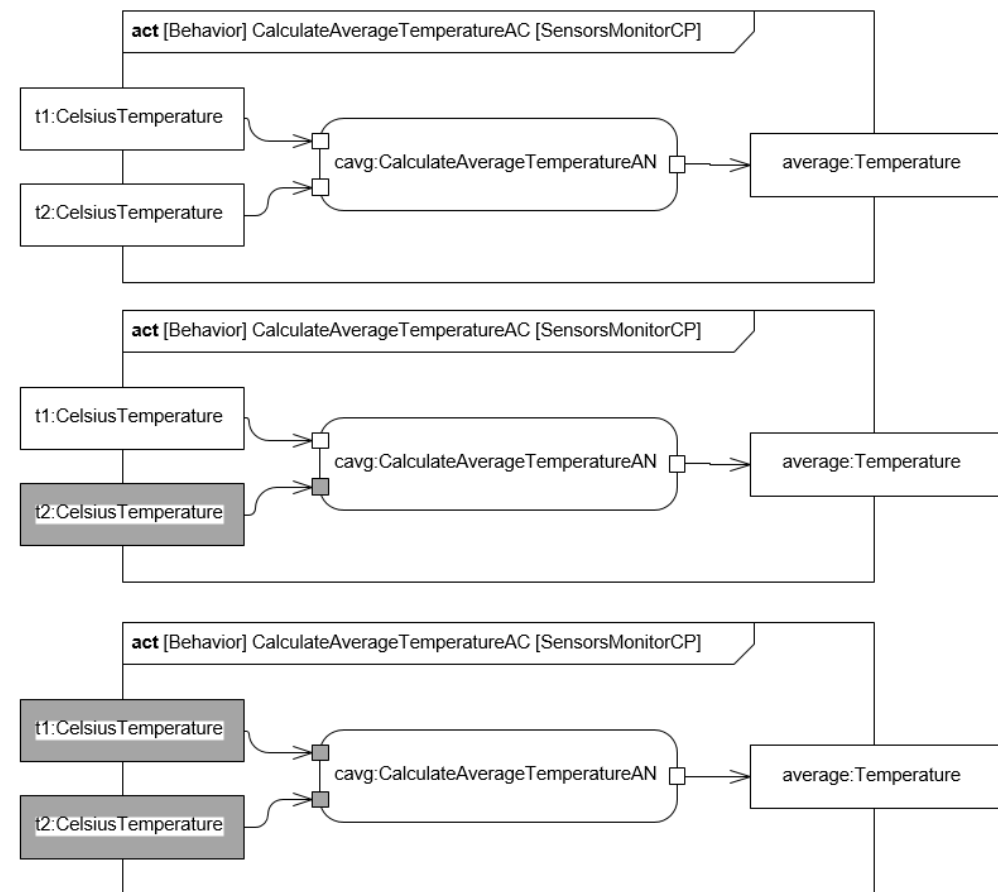


Activity behavior

Example – 1/2

22

- This example illustrates the sequence of steps involved in the behavior of an activity.
- (a) illustrates the activity with no data in its pins.
- (b) shows that the activity received data in the *t2* pin.
- (c) shows that the activity received data in the *t1* pin.

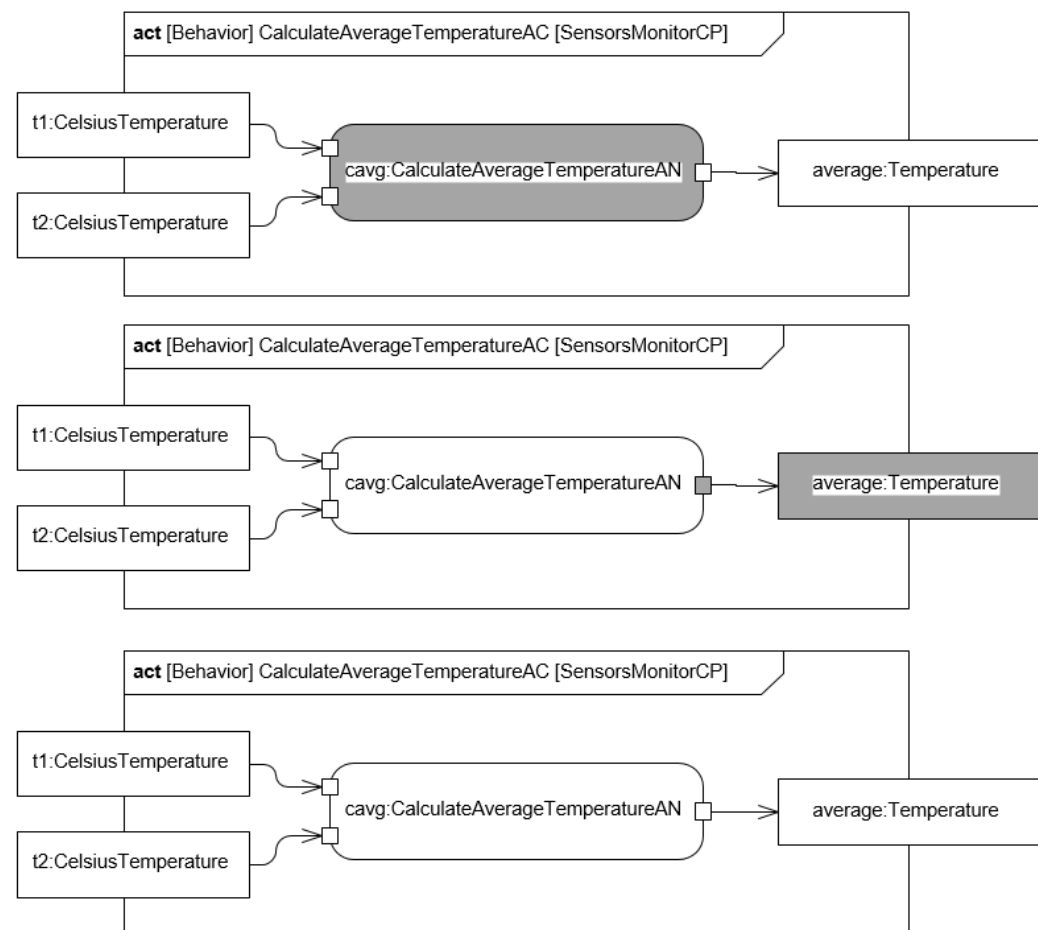


Activity behavior

Example – 1/6

23

- (d) after receiving data in all *in* pins, the action starts -
- (e) when it finishes, the *averageTemp* out pin receives data
- (f) after that, the activity returns to the initial state of waiting data (a)





Action

Action

Conceptual overview – 1/2

- An action refers to a simple behavior that receives input parameters and calculates the output parameters
 - an action executes from beginning to end without interruptions
- An action is executed when all *in pins* receive data and ended when it puts data in the *out pin* (start-stop semantic)
 - when an action consume data from the *in pin*, it is executed
 - when an action provides its results in the out pin it stops
- An example of an action is a function to calculate the average using two input parameters

Action

Conceptual Overview – 2/2

- Actions may be
 - internal actions, which are
 - SysADL built-in actions
 - user-defined actions
 - interaction actions formed by one of the two SysADL built-in interactions
- An action is defined by
 - pre-conditions, expressed in terms of the input parameters
 - post-conditions, expressed in terms of input and output parameters
 - both are expressed using equations
- An example of a post-condition is represented by an equation expressing that the average between two temperatures is the sum of them divided by 2

Action

Use in SysADL

27

Action use

- We can use an action to specify a call to a simple behavior
 - an action is used inside an activity
- When we use an action we should provide its name and its type (see next slide)
 - parameters are represented as pins
- You should first define action types and then use actions of the defined type

SysADL notation

- We represent an action use as a rounded rectangle
- We represent pins in actions using a small square in the border of an activity

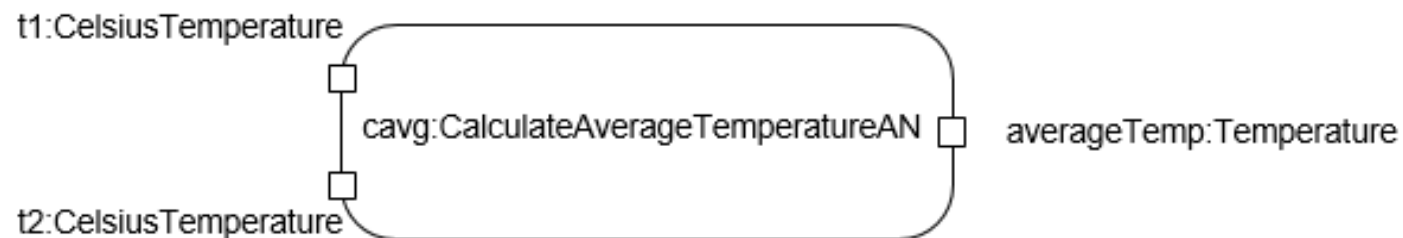
Action

Definition example in SysADL

Example

- In this example, we have the use of the *cavg* action of *CalculateAverageTemperatureAN* type (see later)

SysADL notation



Action

Definition in SysADL

Action definition

- Before using an action we need to define its type
- Action types are named
 - As a convention, we use an AN suffix to the name of an activity type.
- Pre and Post-conditions are expressed using equations

SysADL notation

- We represent an action type using the <<action>> stereotype.
- A rectangle represents the action and contains two additional compartments:
 - parameters, to define their names and types
 - constraints, to define pre and post-conditions equations

Action

Definition example in SysADL

Example

- In this example we have the definition of the *CalculateAverageTemperatureAN* action type of the previous example

SysADL notation

«action» CalculateAverageTemperatureAN
<i>parameters</i> in t1:CelsiusTemperature in t2:CelsiusTemperature out average:CelsiusTemperature
<i>constraints</i> «post-condition» CalculateAverageTemperatureEQ(t1, t2, average)



Constraints

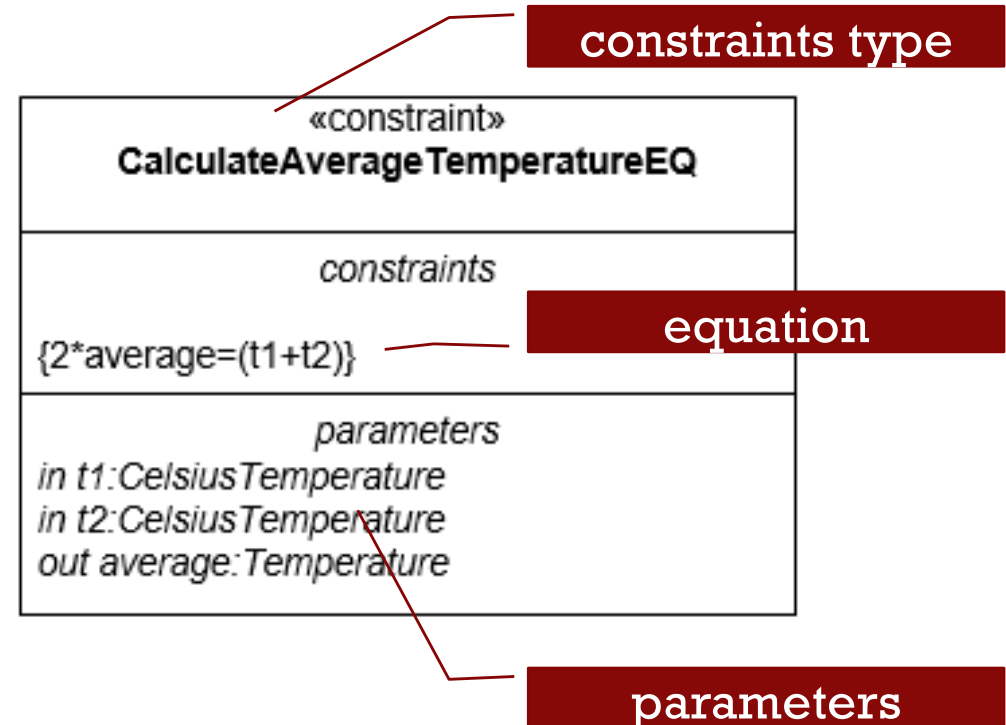
Constraints

Constraints definition in SysADL – 1/2

Constraints definition

- Before using an equation to specify an action we need to define its type
- We need to define
 - the parameters and their types
 - equations using the parameters, specifying pre and post-conditions
- As a convention, we use an EQ suffix to the name of a constraint type represented by equations

SysADL notation



Constraints

Constraints definition in SysADL – 2/2

Constraints type example

- We represent a constraint type using the `<constraints>>` stereotype.
- In this example we have the definition of the *CalculateAverageTemperatureEQ* constraint type of the previous example
 - it contains three parameters (*t1*, *t2*, and *average*)
 - the equation express the output parameter (*averageTemp*) in terms of the input parameters (*t1*, *t2*)

SysADL notation

«constraint» CalculateAverageTemperatureEQ
constraints {2*average=(t1+t2)}
parameters in t1:CelsiusTemperature in t2:CelsiusTemperature out average:Temperature



Relationships

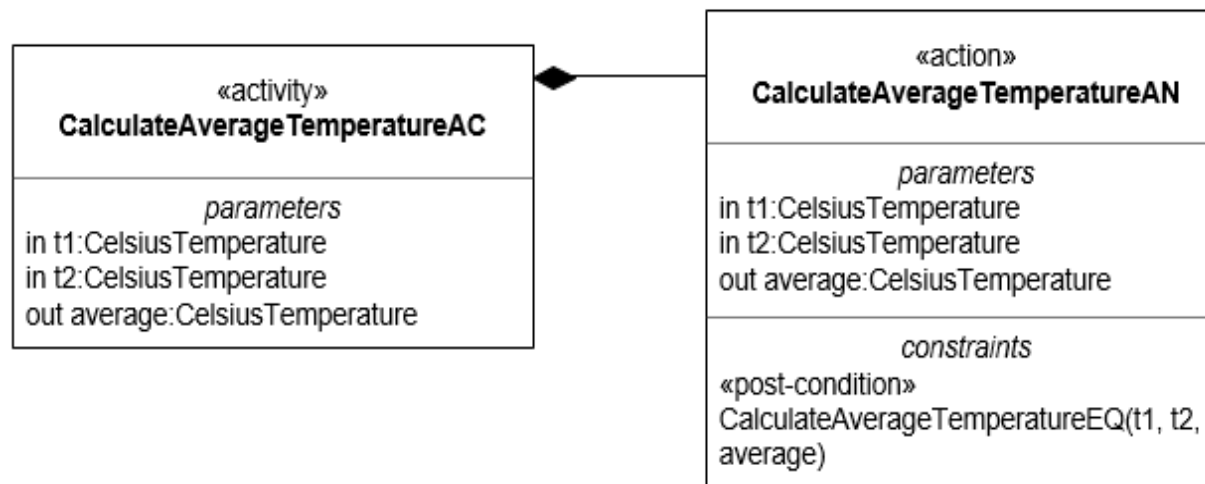
Activities and actions

Composition

Activities and action composition

- Since actions are part of activities, we can represent this relationship using the composition association
 - Cardinality is a constraint that defines how many actions are concurrently executed inside an activity. By default, the cardinality is 1
- This example illustrates a relationship between the *CalculateAverageTemperatureAN* action and the *CalculateAverageTemperatureAC* activity with cardinality 1

SysADL Notation



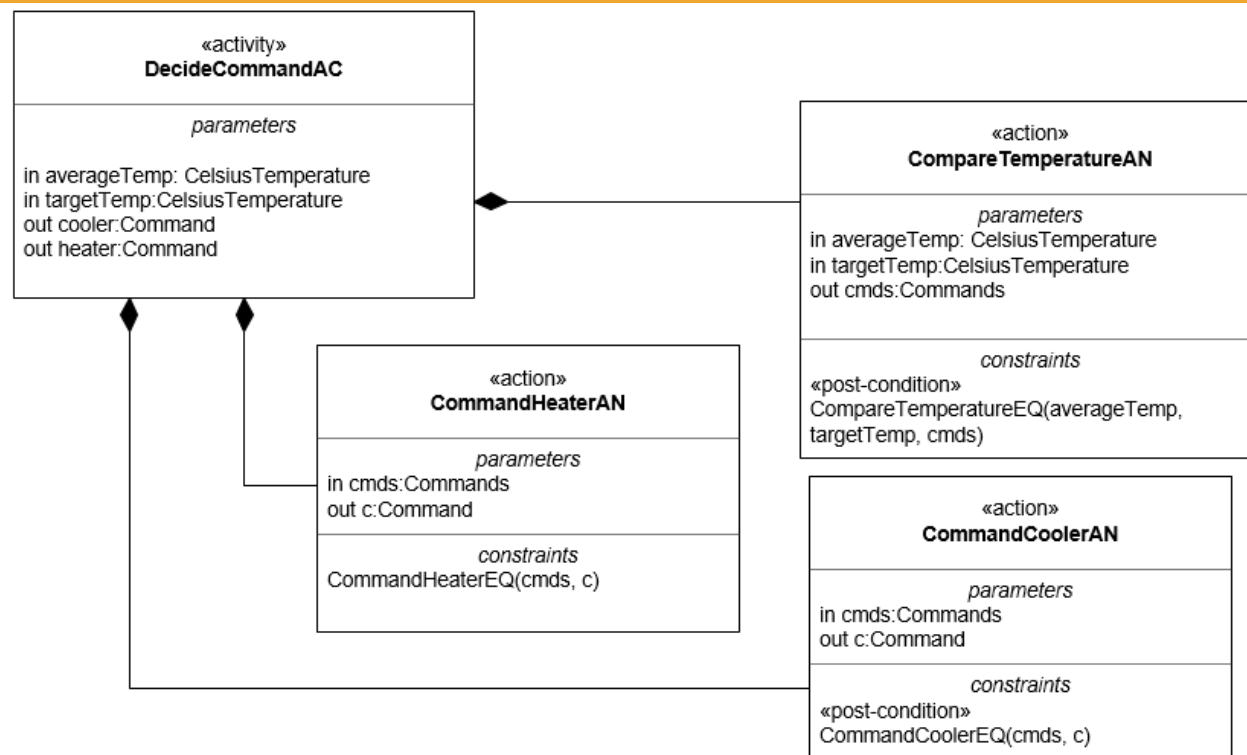
Activities and actions

Example 2

Example 2

- This example show that the DecideCommandAC activity is composed of three actions: CompareTemperatureAN; CommandHeaterAN; and CommandCoolerAN

SysADL Notation



Allocating Activities and Actions

Relating Components and Connectors with Activities and Actions

■ Components

- each component has an activity to express its behavior
- each activity must have **one** or **more** actions to describe the behavior details
- each action must have equations describing its pre- and post conditions

■ Connectors

- each connector has an activity to express its behavior
- the activity of a connector may have **none**, **one** or **more** actions to describe the behavior details
- each action must have equations describing the constraints governing its atomic behavior



Diagrams for Behavioral Views

Diagrams for Behavioral Views

Conceptual overview

- Behavioral views in SysADL are organized in terms of
 - Block Definition Diagrams (bdd)
 - Activity Diagrams (act)
 - Parametric Diagram (pa)

Block Definition Diagrams

Conceptual overview

- Activities and actions are defined using block definition diagrams
- Since activities are composed of actions, we represent activity-action compositions relationships in the bdd
- We also use bdd to define equations used in actions

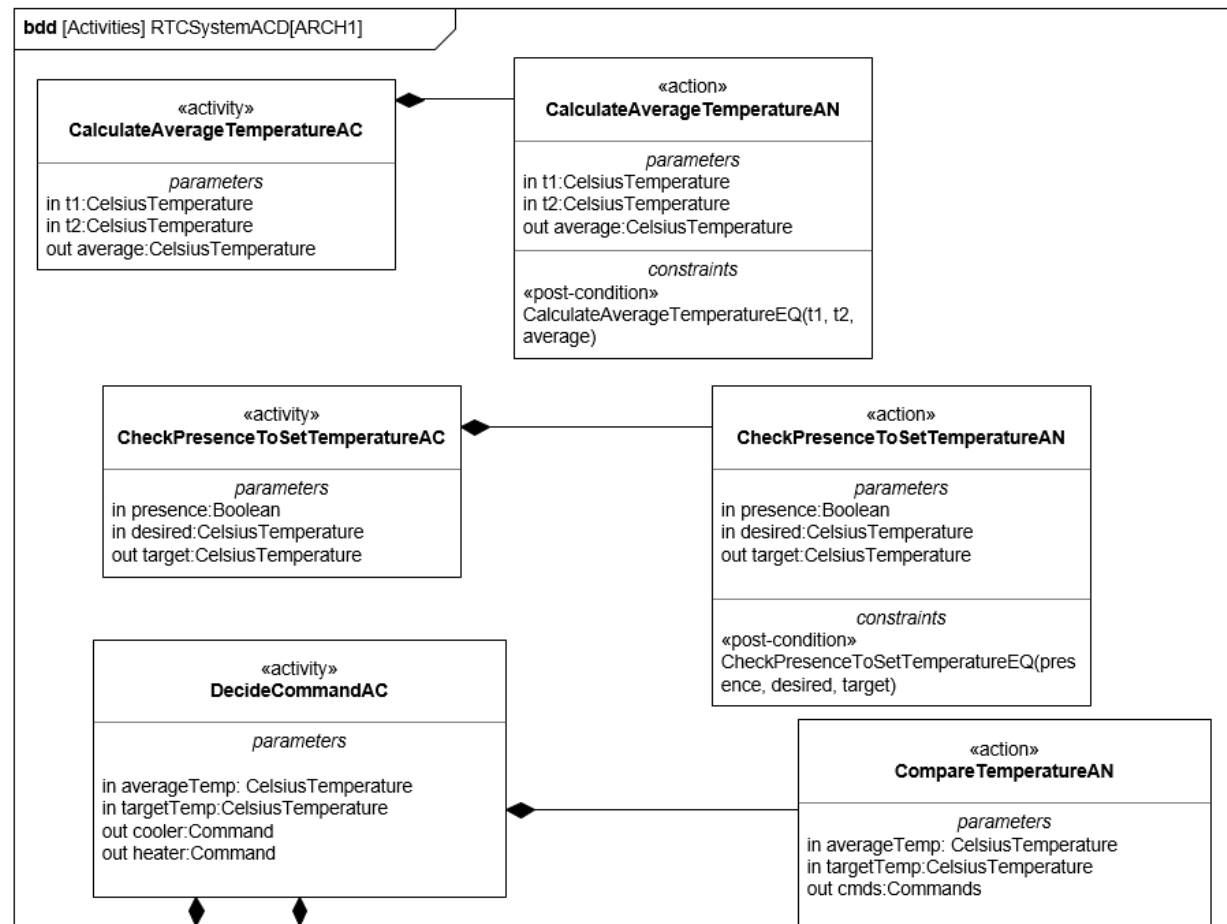
Block Definition Diagram

Example 1 description

- The *RTCSystemACD bdd* in the next slide is an activity diagram that partially describes the definition of the activities and actions in the architecture description of the *RTC System*.
- We can see in the bdd that
 - the *CalculateAverageTemperatureAC* activity is composed of only one action: *CalculateAverageTemperatureAN*
 - the *CheckPresenceToSetTemperatureAC* activity is composed of only one action: *CheckPresenceToSetTemperatureAN*
 - the header of the bdd indicates that it defines [activities] and by convention we use the suffix *ACD* in the name of the diagram to refer to activities definition

Block Definition Diagram

Example (partial)





Activity Diagram

The activity diagram

Overview

- We define architecture behavior using activity diagrams
- We define an activity diagram for describing the behavior of the following elements: components, connectors, and their ports
- An example of activity diagram to represent the average temperature behavior should describe the activity type, the correspondent component, the parameters and the actions

Activity diagram

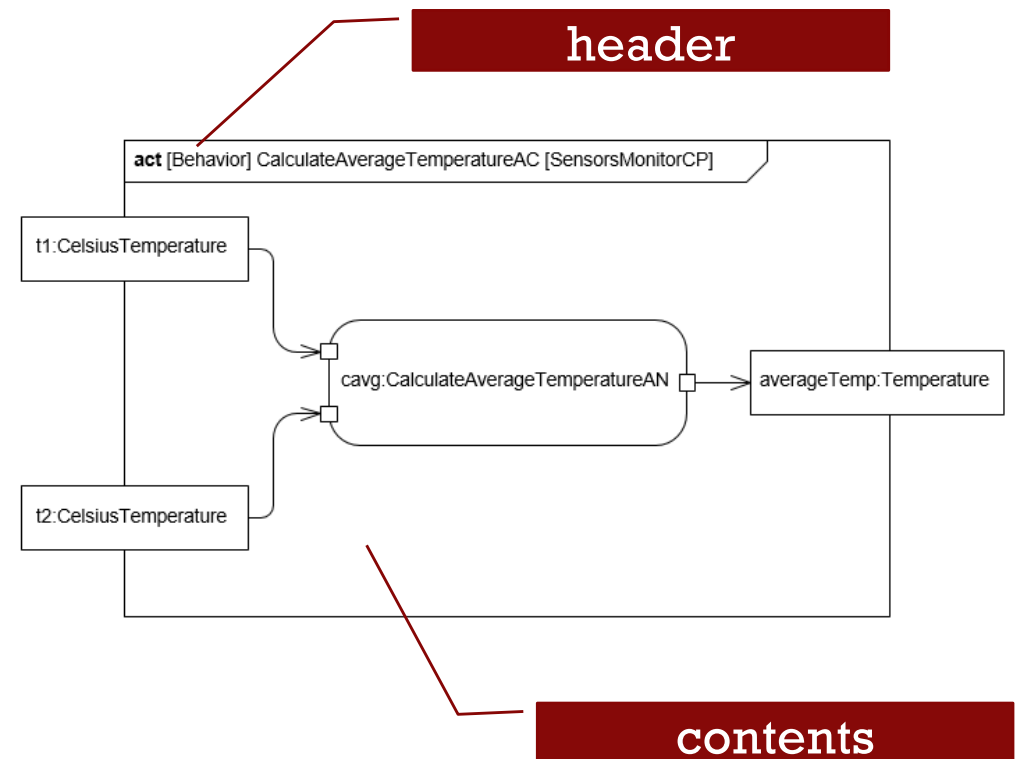
Header and body

45

Activity diagram definition

- We define an activity diagram by specifying its header and its contents
- The header describes the kind, name, and associated elements
- The contents represents the use of elements to define the body of an activity

SysADL Notation



Activity diagram

Header

Activity diagram header

- As a convention, in the the header we use the following elements:
 - **act** – indicates that the diagram is an activity diagram
 - **[Kind]** – indicates if the diagram is specifying a behavior or a protocol
 - **[Behavior]** – indicates a behavior of a component or a connector
 - **[Protocol]** – indicates a protocol of a port
 - **Diagram name** – used to uniquely identify the diagram
 - As a convention we include the AC suffix to the name
 - **[Element]** – used, optionally, when the diagram details an specific element

SysADL Notation

act [Behavior] CalculateAverageTemperatureAC [SensorsMonitorCP]

act [Protocol] CTemperaturePC [CTemperatureOPT]

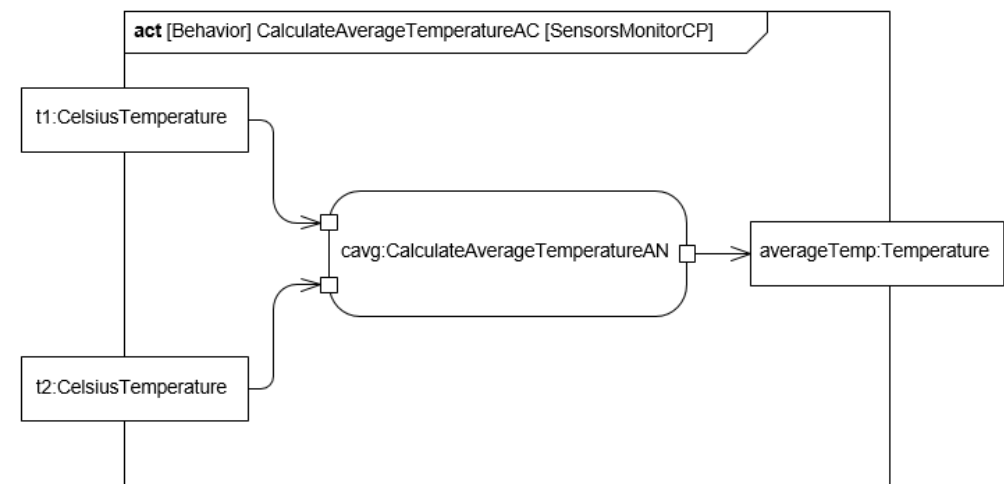
Activity diagram

Component and connector behavior

Behavior definition

- We can use an activity diagram to define the behavior of a component or a connector
 - the header should inform
 - that the diagram represents a behavior
 - the name of the activity type
 - the name of the component type or connector type whose behavior we are describing
 - the contents represents the body of the behavior with
 - the pins representing the parameters
 - the actions and the control flow that specify the behavior

SysADL Notation



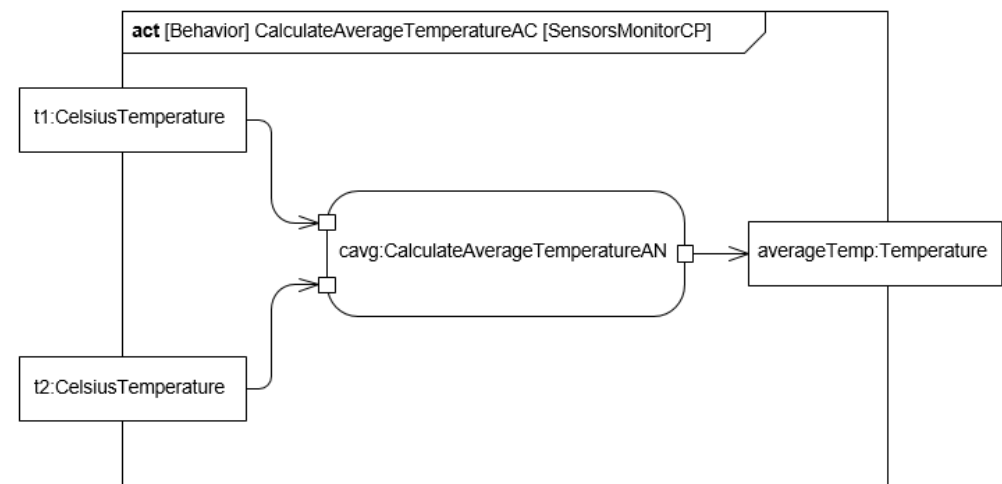
Activity diagram

Component Behavior Example

Example description

- We define the following activity diagram for describing the behavior of the *SensorMonitorCP* component
 - it defines two input parameters represented as pins: *t1* and *t2*
 - it defines an output parameter represented as a pin: *averageTemp*
 - it defines an action: *cavg*
- All types should be previously defined.

SysADL Notation



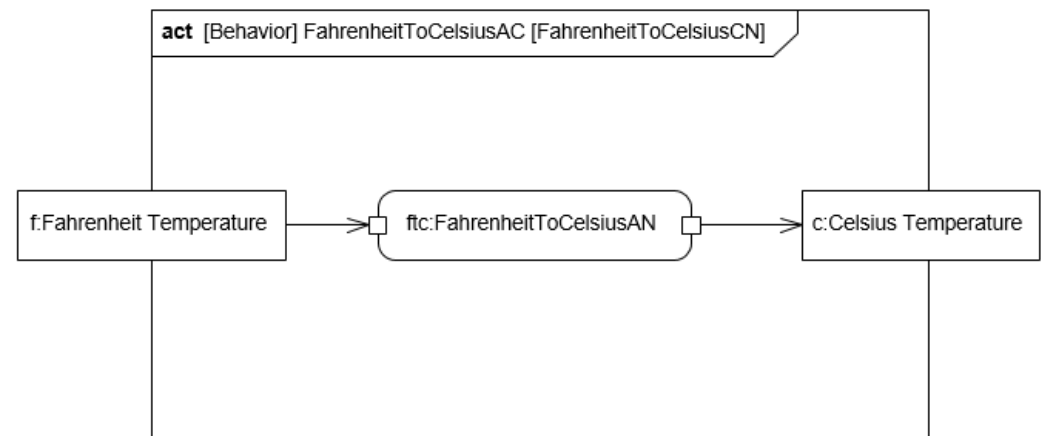
Activity diagram

Connector behavior example

Example description

- We define the following activity diagram for describing the behavior of the *FahrenheitToCelsiusCN* connector
 - it defines one input parameter represented as the *t* pin
 - it defines one output parameter represented as the *c* pin
 - it defines an *ftc* action
- All types should be previously defined

SysADL Notation



Activity diagram

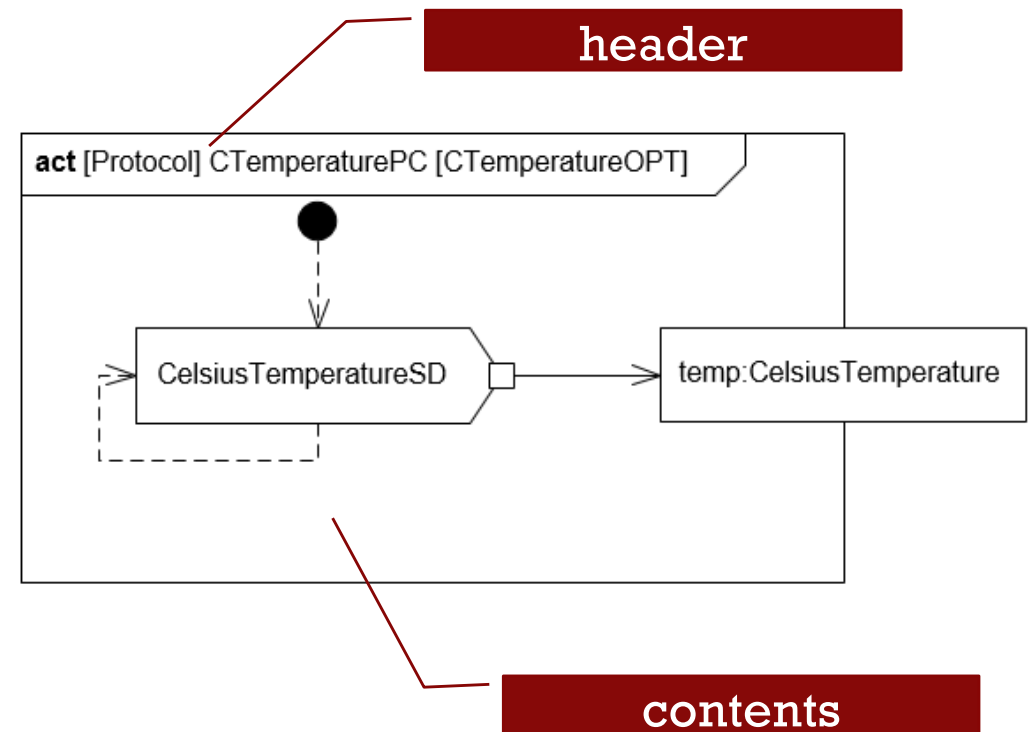
Port protocol – 1/2

50

Port protocol

- We can use an activity diagram to define the protocol of a port
 - the header should inform
 - that the diagram represents a protocol
 - the name of the activity type
 - the name of the port whose protocol we are describing
 - the contents represents the body of behavior (next slide)

SysADL Notation



Activity diagram

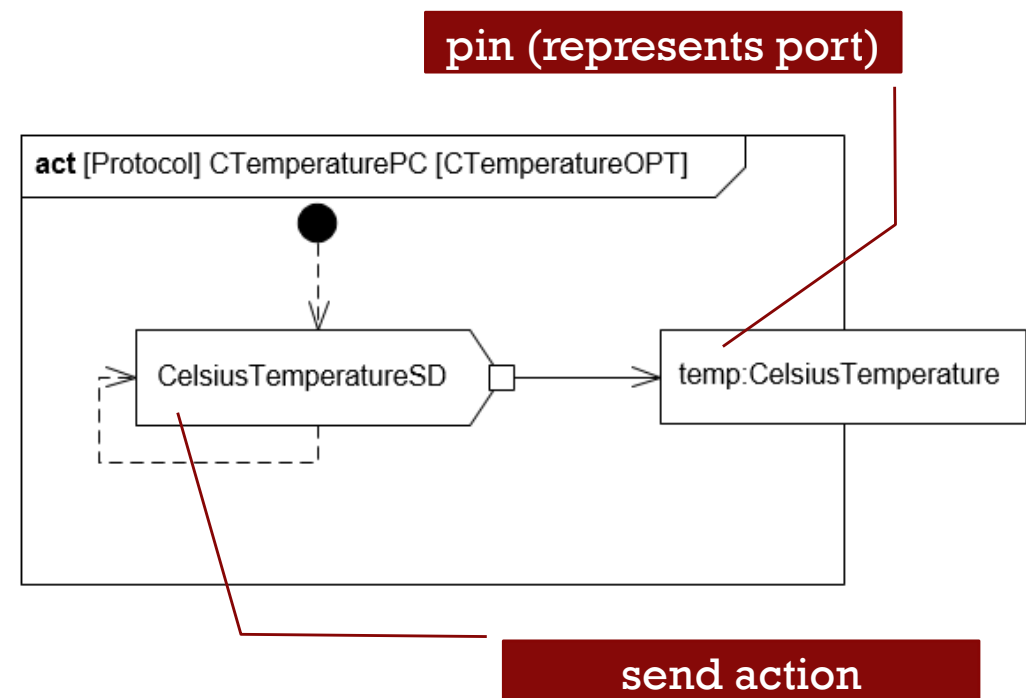
Port protocol – 2/2

51

Port protocol

- The activity diagram contents represent the body of behavior with
 - *pins* representing ports
 - the *send* or *receive* actions that specify what information the port is sending or receiving
- As a convention, we use the data or value type name and a suffix
 - SD for sending
 - RC for receiving

SysADL Notation



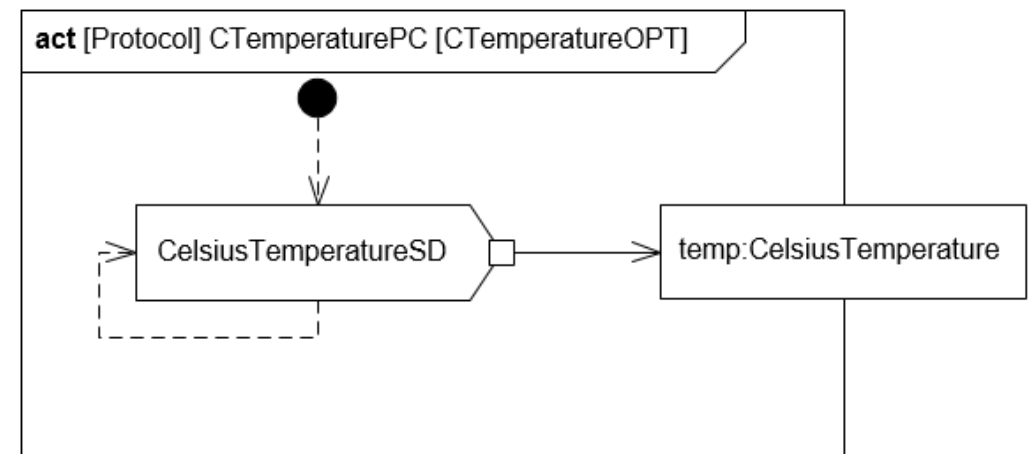
Activity diagram

Example

Port protocol example

- We define the following activity diagram for describing the behavior of the *CTemperatureOPT* port protocol
 - it defines a sending action, named *CelsiusTemperatureSD* with a pin
 - it sends a value (*temp*) of the *CelsiusTemperature* type to the out port
 - a control flow represents that after sending a value, it returns to send a next value
- All types should be previously defined

SysADL Notation





Parametric diagram

Parametric diagram

Conceptual overview

- We specify actions using parametric diagrams
- We define a parametric diagram for specifying an action of a behavior of the following elements: components, connectors, and their ports
- An example of a parametric diagram to represent how to calculate the average temperature behavior should describe the *constraints type*, the correspondent *parameters*, and the *equation* that governs the behavior

Parametric diagram

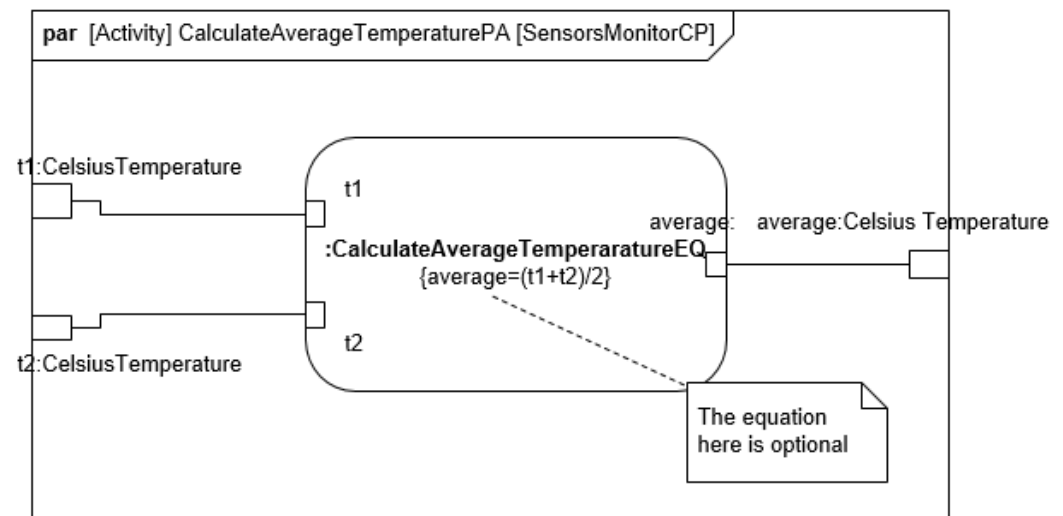
55

Use in SysADL

Parametric diagram use

- We can use the parametric diagram to specify an action
 - A constraint associates the parameters of an action to an equation that defines how the output parameters depends on the input parameters
- When we use an equation we should provide its name and its type (see next slide)
 - We should first define equation types and then use actions of the defined type

SysADL notation



Parametric diagram

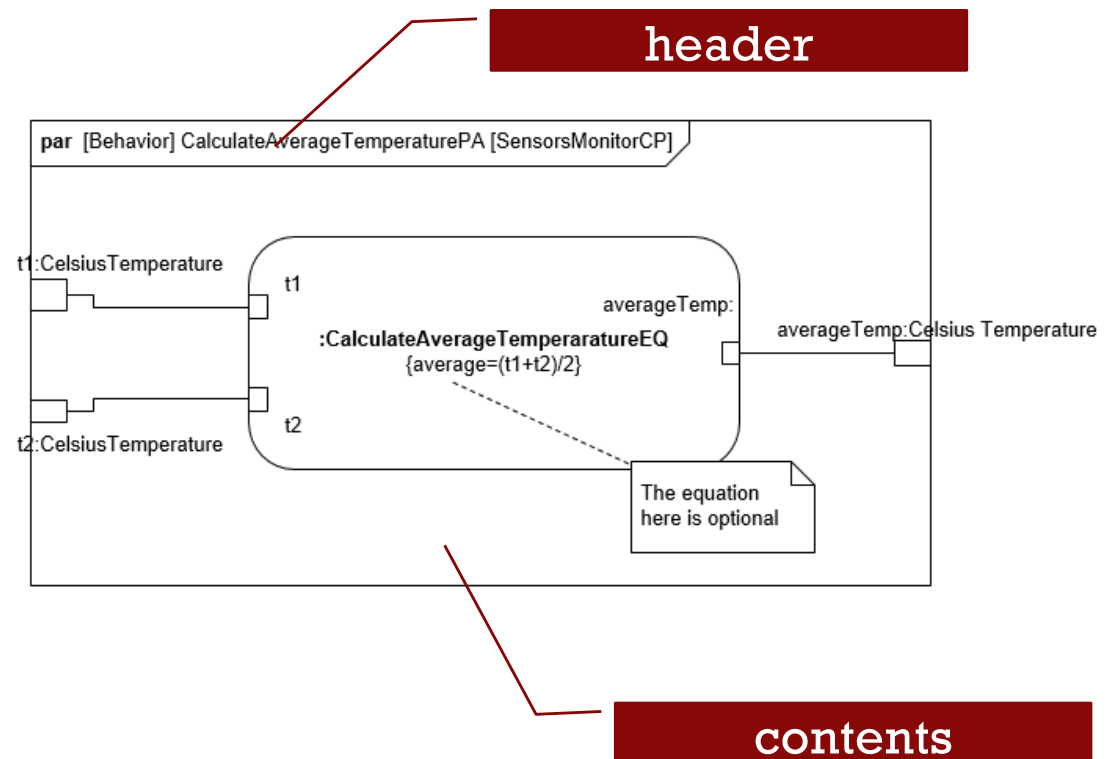
56

Header and content

Parametric diagram definition

- We define a parametric diagram by specifying its header and its contents
- The header describes the kind, name, and associated elements
- The contents represents the use of constraints to specify an action

SysADL notation



Parametric diagram

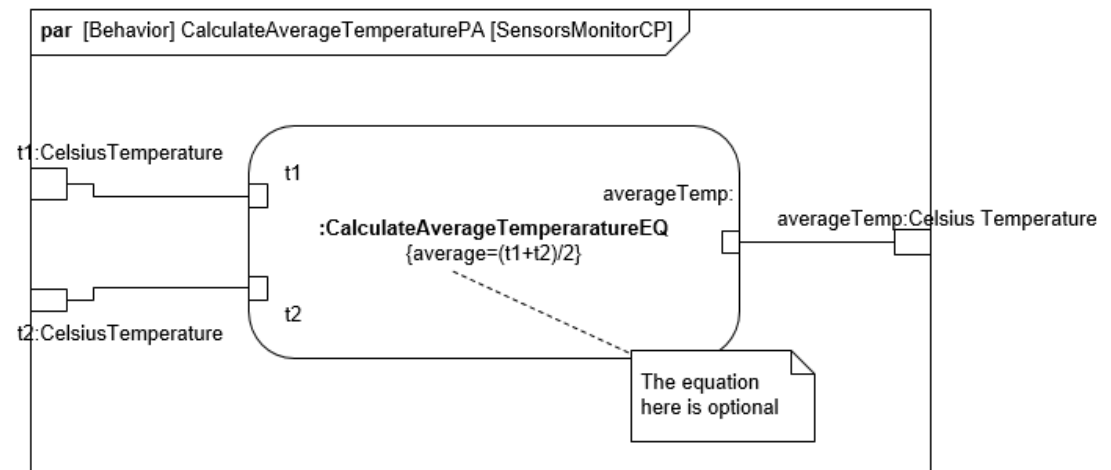
Header and content - example

57

Example description

- We define the following parametric diagram for describing the constraints of the *SensorMonitorCP* component
 - it defines two input parameters: *t1* and *t2*
 - it defines one output parameter: *averageTemp*
 - it uses an equation: *CalculateAverageTemperatureEQ*
 - the equation in curly brackets is optional but it should be in the constraint type definition (see later)
- All the parameters types and the equation type should be previously defined.

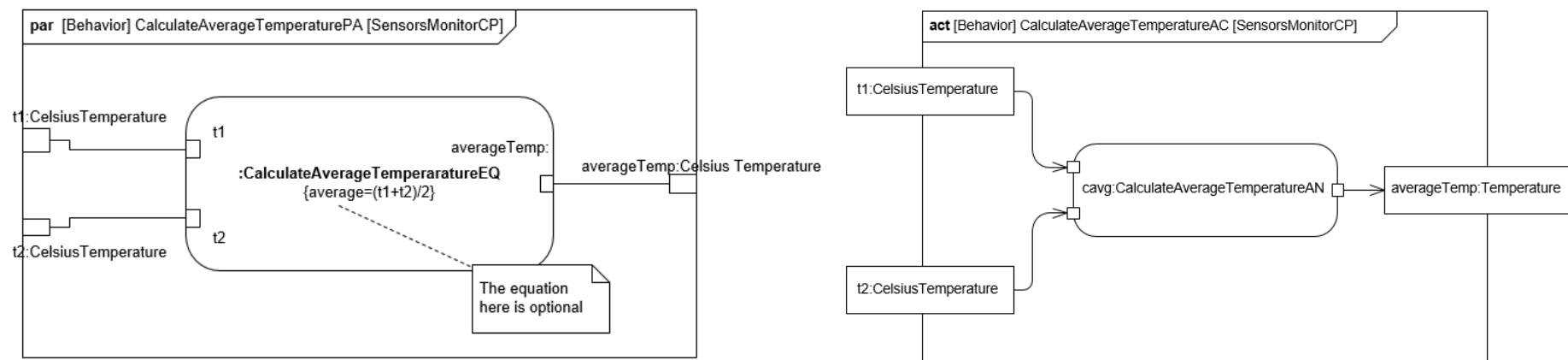
SysADL notation



Parametric and Activity diagram

Complementary views of the behavioral viewpoint

- A constraint in a parametric diagram is directly linked to an activity (action)
- the two input parameters of the parametric diagram are references to the input parameters of the activity diagram, as a convention they have the same name





Relating Structural and Behavioral Viewpoints

Relating Structural and Behavioral Viewpoints

Conceptual overview

- In SysADL, the behavioral viewpoint is allocated to the structural viewpoint.
- For each defined component, connector, and port, there is an activity specifying its behavior.
- The behavior of the configuration of a composite component or the overall software architecture can be derived from the specification of their constituent elements.

Component and Behavior

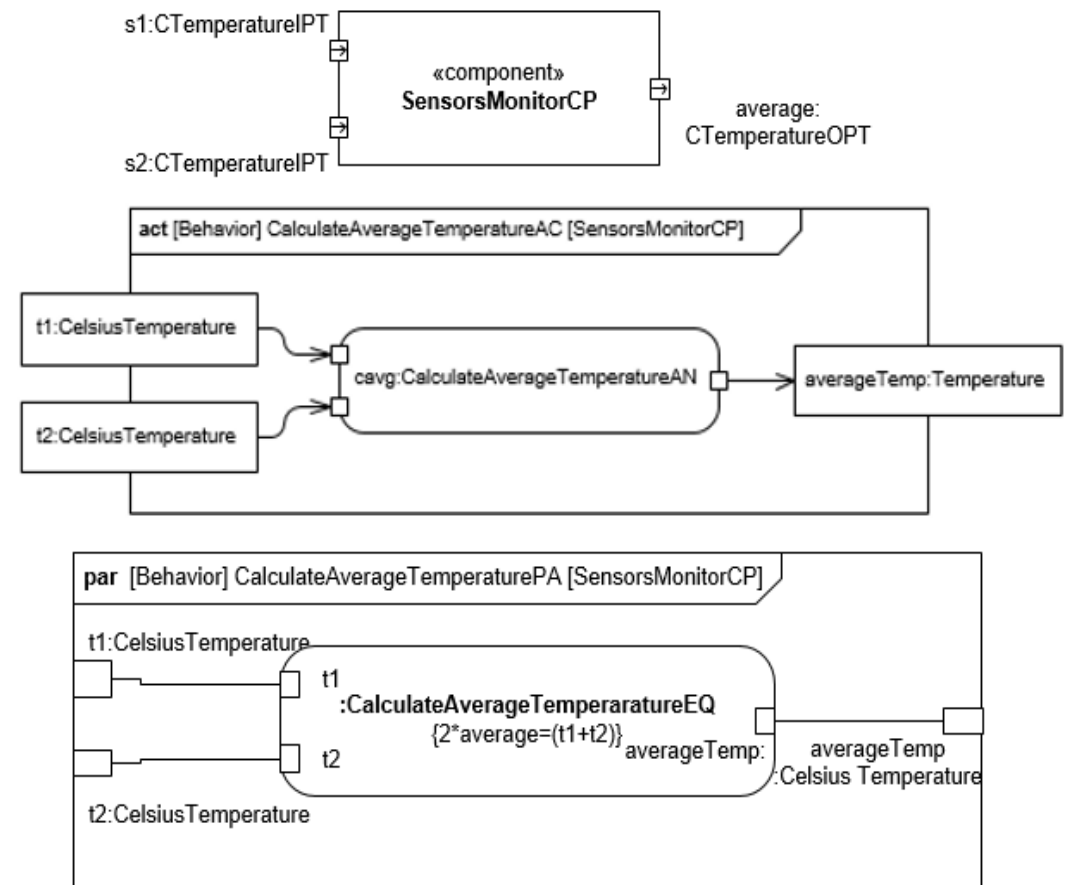
61

Example in the RTC System

Example

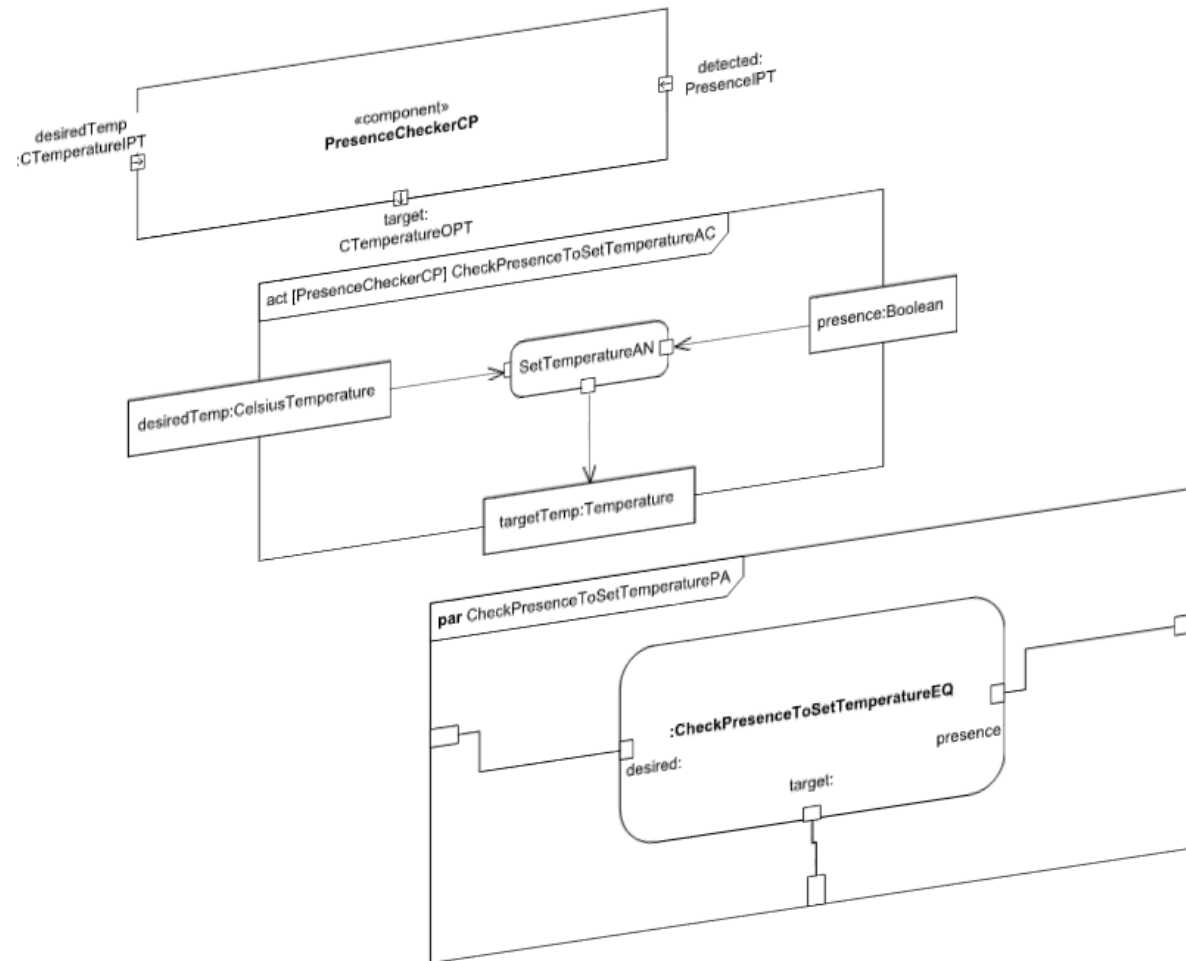
- In this figure, we show the SensorsMonitorCP component
- The *CalculateAverageTemperatureAC* activity diagram shows the behavior of the component
 - As a convention we associate the activity diagram with the component by informing its name in the header
- The *CalculateAverageTemperaturePA* parametric diagram shows the equation that constraining the behavior
- We can also see the association of ports with pins and parameters
- Both of them represent the type of data that can flow in the correspondent ports.

SysADL Notation



Component and Behavior

3D perspective of components and behavior



Summary

- In this chapter you learnt
 - the main behavioral software architecture concepts: activities, actions, constraints
 - auxiliary concepts used to define the main concepts: parameters, pins, flows, equations
 - the SysADL notation to represent these concepts
- You learnt how to
 - define the behavior of software architectures
 - using and defining activities, actions, constraints
 - use the SysADL diagrams to represent the architecture
 - organizing the behavioral view in terms of diagrams: bdd, act, and par

For Further Reading

- Hofmeister, C., Nord, R., Soni, D.: Applied Software Architecture. Addison-Wesley Professional, Boston (1999)
- Taylor, R.N., Medvidovic, N., Dashofy, E.M.: Software Architecture: Foundations, Theory, and Practice, 1st edn. Wiley, New York (2010)