# Accessing data in Activity Edge guards

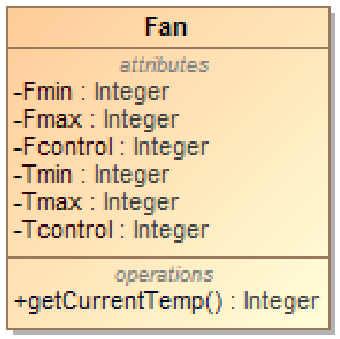
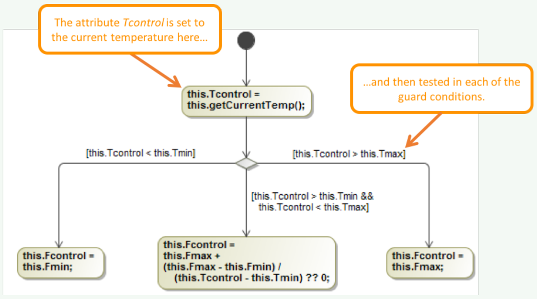
When an Alf expression is used to specify a guard on an Activity Edge, the guard condition may be based on two source of data:

1. If the Activity containing the Activity Edge has a context Classifier, then the values of the attributes of that Classifier can be accessed.
2. If the Activity Edge is outgoing from a Decision Node, then the *decision input value* for the Decision Node may also be used, if it has one.

> If a Decision Node has a single incoming Object Flow, then the value on this Object Flow becomes the decision input value for the Decision Node. Alternatively, the decision input flow may be designated explicitly by setting the **decision input flow** property of the Decision Node. If a Decision Node has a single incoming Control Flow (with no **decision input flow** or **decision input** Behavior), then it has no decision input value.

To access context Classifier attributes in Alf:

- Use Alf *this* expressions, such as *this.Tcontrol.*

> Consider the *Fan* Class shown in the image below. This Class is supposed to control the speed of a fan between given maximum and minimum values, in order to keep a control temperature between maximum and minimum values.
>
> 
>
>
> A simple Fan control class
>
> The Activity Diagram below shows the classifier behavior of this Class. The first Action in this Activity calls the *getCurrentTemp* operation and sets the *Tcontrol* attribute to the value returned from that call. The value of the *Tcontrol* attribute is then used to determine which branch is taken coming out of the Decision Node. Since all the Activity Edges are Control Flows, the Decision Node has no decision input value.
>
> 
>

The attribute Tcontrol is set to the current temperature here...
> ...and then tested in each of the guard conditions.
> this.Tcontrol = this.getCurrentTemp();
> [this.Tcontrol < this.Tmin]
> [this.Tcontrol > this.Tmax]
> [this.Tcontrol > this.Tmin && this.Tcontrol < this.Tmax]
> this.Fcontrol = this.Fmin;
> this.Fcontrol = this.Fmax + (this.Fmax - this.Fmin) / (this.Tcontrol - this.Tmin) ?? 0;
> this.Fcontrol = this.Fmax;
>
> Accessing attribute data in guards

To access the decision input value of a Decision Node in Alf:

---

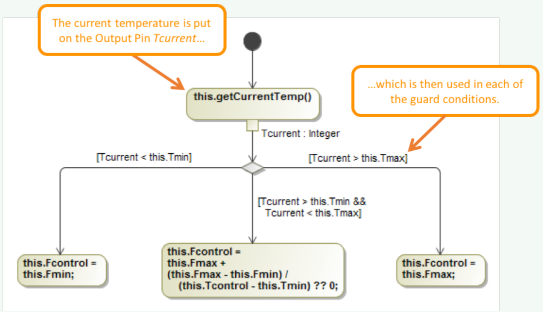
- Use the name of the Object Node (typically an Activity Parameter Node or Output Pin) that is the source of the decision input flow.

  OR

- Give the decision input flow a name and use that.

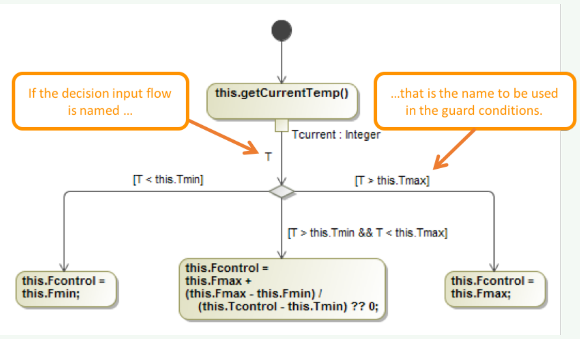
Unless the attribute *Tcontrol* is used elsewhere, its only purpose in the Activity shown in the previous example is to pass the current temperature value from the operation call to the guards. In this case, a simpler way to provide this value to the guards is to make it the decision input value for the Decision Node. In the revised Activity shown below, the result of the operation call is now provided on an Output Pin of the first Action called *Tcurrent* (via the shortcut for using Alf expressions in Opaque Actions). This Output Pin is then connected to the Decision Node by an Object Flow, so the value on the Output Pin becomes the decision input value for the Decision Node. This value can then be accessed in the guard expressions using the name of the Output Pin.

> It is important that the type of an Output Pin used in this way be set properly, so that the guard expressions type check correctly. In this case the Output Pin has been given the type *Integer,* which is consistent with the return type of *getCurrentTemp* and the types of the *Tmin* and *Tmax* attributes to which *Tcurrent* it will be compared. Note that, while the type of the *Tcurrent* Output Pin is shown in the diagram below, the types of Pins are not usually shown by default.


The current temperature is put on the Output Pin *Tcurrent*...
...which is then used in each of the guard conditions.
this.getCurrentTemp()
Tcurrent : Integer
[Tcurrent < this.Tmin]
[Tcurrent > this.Tmax]
[Tcurrent > this.Tmin && Tcurrent < this.Tmax]
this.Fcontrol = this.Fmin;
this.Fcontrol = this.Fmax + (this.Fmax - this.Fmin) / (this.Tcontrol - this.Tmin) ?? 0;
this.Fcontrol = this.Fmax;

Accessing the decision input value in guards

In general, when a Decision Node has a decision input value, the Alf compiler will use the name of the nearest named Object Node providing that value. Most commonly, this will be an Activity Parameter Node or Output Pin, as in the example above. However, if the compiler cannot determine a single relevant name, or if you simply do not want the decision input value name to depend on the name of other Activity Nodes, you can, instead, name the Object Flow that provides the decision input value into the Decision Node. If the incoming decision input flow (whether determined implicitly or explicitly) has a name, then the Alf compiler will use this as the name for the decision input value. For example, in the further revised diagram below, the Object Flow has been given the name *T,* which is now the name used for the decision input value in the guards, rather than the Output Pin name *Tcurrent.*


If the decision input flow is named ...
this.getCurrentTemp()
...that is the name to be used in the guard conditions.
Tcurrent : Integer
T
[T < this.Tmin]
[T > this.Tmax]
[T > this.Tmin && T < this.Tmax]
this.Fcontrol = this.Fmin;
this.Fcontrol = this.Fmax + (this.Fmax - this.Fmin) / (this.Tcontrol - this.Tmin) ?? 0;
this.Fcontrol = this.Fmax;

Accessing the decision input value using the Object Flow name