# ValueFlow (Aspect)

## Overview

The **ValueFlow** aspect is used to express the static properties of a component or an assembly of components, and the relationships and dependencies between those values. By static, we mean that these values are set at design time, and do not change while the system is functioning. For example, the weight and length of an engine component are captured within the ValueFlow aspect.

The ValueFlow aspect also captures the formulas needed to describe parametric components. For example, a driveshaft component design with parametric length will include a formula to describe its weight as a function of material density, diameter, and length.

The section **Model Objects** covers the nodes that hold and calculate values, and the section **ValueFlow Connection** covers the relationships between these nodes.

## Contexts

Most areas of the CyPhy language include the ValueFlow aspect, including designs (Component Assemblies), Components, Design Space models (DesignContainers), and Test Benches.

## Model Objects

### Property

A **Property** object represents a static property of an object or assembly. Properties of components cannot directly be changed by a designer at design time.

#### Attributes

##### **Value**

Holds the current value of the property.

##### **Tolerance**

TBD

##### **ComputationType**

TBD

##### **Disable**

TBD

##### **Default Value**

TBD

##### **Description**

A free-text description of the property.

#### Reference

A **Property** is also a “Reference”-type object. Its referent defines the unit used for the value. To set the unit for a **Property**, find the desired unit in the type library, then click and drag it over the **Property** you want to set.

#### Specialization: CADProperty

A **CADProperty** is a specialization of Property. In practice, the value of a CADProperty must be calculated within a CAD tool, after which the result can be written back to the model.

##### IsCurrent

This Boolean value is set to “True” if the ValueFlow entities that could affect this value have not changed since the value was last calculated.

##### Parameter Name

The key of the corresponding calculated value within a generated CAD model.

##### UnitOfMeasurement

???

### Parameter

A **Parameter** object represents a static property of an object that can be tuned by a system designer at design-time.

#### Attributes

##### Value

Holds the current value of the property.

##### Range

The valid range for the property. For continuous ranges, the format is [min:max]. For a set of enumerated values, the format is [value1 value2 value3].

##### ComputationType

TBD

##### Disable

TBD

##### Default Value

If the **value** field is not specified, this value is assumed.

##### Description

A free-text description of the parameter

#### Reference

A **Parameter** is also a “Reference”-type object. Its referent defines the unit used for the value. To set the unit for a **Parameter**, find the desired unit in the type library, then click and drag it over the **Parameter** you want to set.

#### Specialization: CADParameter

A **CADParameter** is a specialization of Parameter. When a CAD model is built, the value of this object is used to set a specific parameter in the component’s CAD representation. This is useful with parametrically-sizeable components, where the size of a component (or a feature of a component) depends on the size of another component that it is composed with.

##### Parameter Name

The key of the corresponding parameter in the component’s CAD model.

##### Parameter Type

Corresponds to the expected parameter type in the component’s CAD model. Values are Integer, Float, and Boolean.

#### Specialization: CyberParameter

[Description text]

### SimpleFormula

A **SimpleFormula** object performs a simple operation across all ValueFlow inputs, and sends the result along all ValueFlow outputs.

#### Method

The supported operations are: Addition, Multiplication, Arithmetic Mean, Geometric Mean, Minimum, and Maximum.

### CustomFormula

The **CustomFormula** evaluates a mathematical expression involving its inputs, and sends the result along its output.

#### Expression

The expression can include the **CustomFormula’s** named inputs, as well as supported mathematical operators and functions.

##### Supported Functions:

* sin(x)
* cos(x)
* tan(x)
* asin(x)
* acos(x)
* atan(x)
* sinh(x)
* cosh(x)
* tanh(x)
* asinh(x)
* acosh(x)
* atanh(x)
* log2(x)
* log10(x)
* log(x)
* ln(x)
* exp(x)
* sqrt(x)
* sign(x)
* rint(x)
* abs(x)
* min(x1, x2, x3, … )
* max(x1, x2, x3, … )
* sum(x1, x2, x3, … )
* avg(x1, x2, x3, … )

##### Supported Operators:

* &&
* <=
* >=
* !=
* ==
* >
* <
* +
* -
* \*
* /
* ^

##### Examples:

* MaterialVolume \* MaterialDensity
* 600 + max(TurretHoleDiameter,1100)
* ( (Width)^5 \* Depth) / Height

## ValueFlow Connection

### Semantics

A ValueFlow connection can be drawn between any two ValueFlow nodes (listed under Model Objects), with some restrictions (see Constraints, below).

A connection means that the value of the source node is assigned to the destination node(s). These values are calculated and assigned when the FormulaEvaluator component is executed. They are also automatically evaluated as an initial step by each other interpreter.

### Constraints and Restrictions

#### CustomFormula Output

The output of a CustomFormula node ***must*** directly connect to a Parameter or Property node that has a unit specified (reference is not null).

This is because the unit cannot necessarily be inferred from the expression. The expression may represent a polynomial fit equation, in which case the resulting unit would be incorrect. Since output unit cannot be inferred, the expression is evaluated using the input values directly without converting to SI equivalent values based on unit and unit check is not performed for the output based on input units.

#### Only One ValueFlow Input for Parameters and Properties

A Parameter or Property cannot have more than one ValueFlow input.

#### CustomFormula To CustomFormula and SimpleFormula To SimpleFormula Connections

The output of a CustomFormula node ***can*** directly feed another CustomFormula via a ValueFlow connection. Since output unit of a CustomFormula cannot be inferred so no unit check is performed, expressions are simply evaluated.

The output of a SimpleFormula ***can*** directly feed another SimpleFormula. Since unit check and conversion can be performed for all operations supported by a SimpleFormula, the formulas are evaluated using SI equivalent values then converted to the output unit of the connected Parameter or Property node.

#### CustomFormula To/From SimpleFormula

The output of a CustomFormula node ***cannot*** directly feed another SimpleFormula via a ValueFlow connection. This is because SimpleFormula expects its input to have a valid unit but the unit cannot necessarily be inferred from the expression of a CustomFormula.

The output of a SimpleFormula node ***cannot*** directly feed another CustomFormula via a ValueFlow connection. This is because the output of a SimpleFormula is an SI equivalent value which needs to be converted based on the unit of the attached node but a CustomFormula evaluates using the actual value of the inputs and not their SI equivalent value.

In order to flow outputs from SimpleFormula to CustomFormula and vice versa, the output must be connected to a ValueFlowTargetSpecification node first before connecting to the next formula.

## Interpreters