# Building Non-Normative Extensions for Property-Based Requirements

## Overview

Annex E.3 addresses extending requirements that are fundamentally textual in nature. They may be extended with various enumerations (for example RiskKind or VerifyMethodKind), and they may have different modeling constraints applied to the requirements relationships, but the requirements are only expressed as text strings.

Expressing requirements as text strings alone fundamentally limits their ability to be evaluated and verified. This Annex addresses a more formal expression of requirements generally referred to as property based requirements (PBR); one that includes quantitative specification of numerical parameters, relationships, equations and/or constraints.

Current users of text-based requirements have frequently expressed a basic need to represent numerical requirements more precisely, both to reduce ambiguity and facilitate verification by analysis and other methods. This basic need can be decomposed into three primary needs: 1) Requirements must have numerical properties (properties capable of representing numerical values), 2) these numerical properties must be typeable (preferably by ValueType) to account for quantity kind and units, and 3) these numerical properties must be bindable (preferably using BindingConnector) to other model elements (e.g. ConstraintParameters) so they can be evaluated using analysis tools. For the purpose of this discussion, a requirement that meets these three conditions is said to be a property-based requirement.

This kind of property-based requirement is intended to be used with the overall system model to assist in specifying and architecting systems. More generally, the system model may be used as a model-based specification, such as when block instances with specific property values represent the requirement. In this latter case, the model-based specification can further refine the property-based requirement.

Users of property-based requirements may desire a more elaborate capability than the primary need described above. For example, it may be desirable for the requirement to contain a constraint or mathematical expression that formally states an acceptance condition, threshold, or goal. This may alternatively need to be expressed as a set of valued pairs, elaborating both the conditions and the acceptance thresholds for each condition, or by an arbitrary graphical relationship. Some users may want the property-based requirement to formally own a behavior representing the functionality of the requirement, or the behavior by which it is satisfied or verified.

The need for this kind of property-based requirement is illustrated in the simple example of specifying a vehicle’s required stopping distance for various initial speeds and road conditions. The requirement can be expressed in a table as follows:

The Vehicle *stopping distance* shall not exceed the values in Table E.8‑1 as a function of *initial speed* and *pavement condition*.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Initial Speed (mph) | Pavement Condition (wet/dry) | Req'd Stopping Distance-Dry (feet) |  | Initial Speed (mph) | Pavement Condition (wet/dry) | Req'd Stopping Distance-Wet (feet) |
| 0 | dry | 0 |  | 0 | wet | 0 |
| 10 | dry | 4 |  | 10 | wet | 6 |
| 20 | dry | 17 |  | 20 | wet | 22 |
| 30 | dry | 38 |  | 30 | wet | 50 |
| 40 | dry | 67 |  | 40 | wet | 89 |
| 50 | dry | 104 |  | 50 | wet | 139 |
| 60 | dry | 150 |  | 60 | wet | 201 |
| 70 | dry | 205 |  | 70 | wet | 273 |
| 80 | dry | 267 |  | 80 | wet | 357 |
| 90 | dry | 338 |  | 90 | wet | 451 |
| 100 | dry | 418 |  | 100 | wet | 557 |

Table E.‑ Example of Requirement in Tabular Form

An alternative expression in plot format can be:

The Vehicle *stopping distance* shall not exceed the values in Figure E.8‑1as a function of *initial speed* and *pavement condition*.

Figure E.. Example of Requirement in Graphical Form

The input/output parameter relationship or constraint can be specified in equation form, such as in the following example:

**Stopping distance <= (1/(2\*32.174\*alpha)\*(5280\*Initial Speed/3600)^2)**

**Start Speed = 0 …100**

|  |  |
| --- | --- |
|  | **alpha** |
| **dry** | **0.8** |
| **wet** | **0.6** |

More generally, the input and output parameter values may be complex functions of other parameters, and may have probability distributions associated with them.

This annex addresses mechanisms and approaches for building SysML profiles to enable property-based requirements. While examples of property-based requirement profiles are provided in this annex, these are not to be considered normative or even authoritative. Instead, they are intended to be illustrative of the kind of extensions that some users may find desirable. Ultimate responsibility for the compatibility of any property-based requirement profile with a particular requirements management process or toolset rests fully with the user.

## An Example PBR Profile Based on ConstraintBlock

Using «constraintBlock» as a base class for PBR may prove compact, simple, and intuitive. The following example first establishes a PBR user profile, and then employs that profile for a simple user example.

### Profile/Stereotypes of PBR Based on ConstraintBlock

Figure E.8‑2 shows use of both «abstractRequirement» and «constraintBlock» to define a new PBR stereotype, named RequirementConstraintBlock in this example for clarity.

Lightningbolt:Users:skygazer:Documents:3. Career:SysML:1. SysML Spec:5-RTF:SysML 1.5 RTF:Property Based Requirements:Proposal:20160331 Proposal:Annex E.8 pdfs:Figure E.8.2.pdf

Figure E.. Example of a PBR Profile Based on Constraint Block

Basing PBR on ConstraintBlock provides flexibility in expressing the name of required numerical values as ConstraintParameters, which can be typed by ValueTypes and related to properites or parameters of other model elements using binding connectors. Textual requirement statements may be restated as constraint expressions that reference these ConstraintParameters. The value bindings can then be used to evaluate the constraint expression and determine compliance with the requirement.

The numerical required value may then be stored as a DefaultValue of the ConstraintParameter. It may alternatively be specified directly in a constraint expression, rather than a default value, e.g. {requiredWeight = 1450} where requiredWeight is defined as a constraint parameter typed by a value type. Complex requirement criteria may be represented by a series of constraint expressions.

It is also noted that constraint blocks can have owned behavior, and that a constraint expression can be a value expression (with opaque behavior).

### Usage Example using PBR profile based on ConstraintBlock

The following example leverages the above PBR user profile based on ConstraintBlock to specify and evaluate the weight of a vehicle.

The requirement is captured via a PBR (RequirementConstraintBlock), which includes a constraint expression that reflects the textual requirements statement in terms of two defined parameters, actualMass and requiredMass. Both of these parameters are typed by the kilogram value type from the SI value types library. The required value for mass is expressed as a default value of the requiredMass parameter. Note that the required value may have alternatively been expressed as a second constraint expression, e.g. {requiredMass = 1450}. The vehicle itself is represented in the model by a block with a value property for mass, also typed by the kilogram SI value type.

As shown in Figure E.8‑3, the context for evaluating if the requirement has been met is established using a Requirement Context block. This method of context setting is a best practice that is not essential to this example. Both the Vehicle and the Vehicle Mass Requirement are used in this Requirement Context.

Lightningbolt:Users:skygazer:Documents:3. Career:SysML:1. SysML Spec:5-RTF:SysML 1.5 RTF:Property Based Requirements:Proposal:20160512 Proposal:Annex E.8 pdfs:Figure E.8.3.pdf

Figure E.. Example of Requirement Evaluation Context Using PBR Based on Constraint Block

Figure E.8‑4 shows a parametric diagram of the Requirement Context block, useful for establishing the method of evaluating compliance of the vehicleMass value with the Vehicle Mass Requirement. As with any parametric model, it does not actually perform the evaluation/analysis, but it does specify the key relationships so that an evaluation tool may determine if the weight requirement has been met.

Lightningbolt:Users:skygazer:Documents:3. Career:SysML:1. SysML Spec:5-RTF:SysML 1.5 RTF:Property Based Requirements:Proposal:20160512 Proposal:Annex E.8 figures:Updated figs:Figure E.8.4.pdf

Figure E.. Example of Parametric Diagram Using PBR Based on Constraint Block

## An Example PBR Profile Based on Constraint

Constraints are arguably the most straightforward way for representing system requirements. Their specification may be provided by opaque constraint expressions, which can be expressed in formal (and computable) languages like OCL. This allows the constraint statement to be applied directly to a specific design, without necessarily applying a formal evaluation context.

### Profile/Stereotypes of PBR based on Constraint

Figure E.8.5 shows use of both «abstractRequirement» and «constraint» to define a new PBR stereotype, named CbRequirement in this example.

Lightningbolt:Users:skygazer:Documents:3. Career:SysML:1. SysML Spec:5-RTF:SysML 1.5 RTF:Property Based Requirements:Proposal:20160331 Proposal:Annex E.8 pdfs:Figure E.8.5.pdf

Figure E.. Example of a PBR profile based on Constraint

### Example using PBR profile based on Constraint

Figure E.8.6 shows how requirements are specified on the model representing a specification. Note that, as modeled here, the requirement represented by Constraint2 applies to any instance of the Vehicle block while the one represented by Constraint1 applies to instances of Vehicle block which are “used” as defined by the “vehicle” role of the Context block, such as the design weight of the vehicle on a bridge or vehicle transporter.

Lightningbolt:Users:skygazer:Documents:3. Career:SysML:1. SysML Spec:5-RTF:SysML 1.5 RTF:Property Based Requirements:Proposal:20160331 Proposal:Annex E.8 pdfs:Figure E.8.6.pdf

Figure E.. Example of PBR based on Constraint used in different contexts

Figure E.8.7 show a particular case where testedVehicle is an instance of the Vehicle block and AnalysisContext an instance of the Context block, as specified above. A simple evaluation of model constraints using a classical OCL evaluator would produce a report showing that Requirement/Constraint2 is met, while Requirement/Constraint1 is violated.

Lightningbolt:Users:skygazer:Documents:3. Career:SysML:1. SysML Spec:5-RTF:SysML 1.5 RTF:Property Based Requirements:Proposal:20160412 Proposal:Annex E.8 pdfs:Figure E.8.7.pdf

Figure E.8. Establishing an Analysis Context for evaluating requirement compliance using PBR based on Constraint.