SEVENTH FRAMEWORK PROGRAMME

Designing for adaptability and evolution in system of systems engineering

Extension of standard profiles for DANSE Modelling: Support for analysis and optimization

D_6.5.2

Deliverable Information			
Nature	R	**Dissemination Level**	PU
Project	DANSE	**Contract Number**	INFSO-ICT-287716
Deliverable ID	D_6.5.2	**Date**	2014-08-01
Status	Final	**Version**	1.0
Contact Person	Christoph Etzien	**Organisation**	OFFIS
Phone		**E-Mail**	Christoph.etzien@offis.de

AUTHORS TABLE

Name	Company	E-Mail
Christoph Etzien	OFFIS	Christoph.etzien@offis.de
Evgeny Shindin	IBM	EVGENSH@il.ibm.com
Alessandro Mignogna	ALES S.r.l.	alessandro.mignogna@ALES.EU.COM
Alexandre Arnold	EADS-FR	alexandre.arnold@eads.net

CHANGE HISTORY

Version	Date	Reason for Change	Pages Affected
0.1	09.07.2014	Structure of Document	all
0.2	28.07.2014	Added content of previous deliverable	all
1.0	01.08.2014	Finalization of document	all

CONTENT

Content of Figures

Content of Tables

1 Introduction

This document serves as a report for the profile extension of the “Unified Profile for DoDAF/MODAF” **(**UPDM). The modeling task (T6.2) has identified a UPDM sub set to be part of the domain meta model (see D6.2.1). To complement this meta model several extensions for UPDM have been developed and will be validated during the prototype iterations. To enable the test case users to apply the DANSE methodology including the implementation of the domain meta model this task provides an extension profile for UPDM. How these extensions are used to support the DANSE methodology is described in detail in the modeling deliverable (D6.2.3). In this deliverable only a technical documentation is given.

While the extension part to enable stochastic modeling is based on one sub-profile extension there are four different extension parts addressing the dynamicity gap. The first one is the simulation extension part which addresses the dynamicity issue on a system dynamics or operational time scale. This part of the extension allows to specify which are the properties of interest during simulation. The next part the GCSL (“Goal and Contract Specification Language” task T6.3) annotation allows to add text-pattern based structured requirements to the UPDM model in order to be feed to the analysis tools. The third extension part addresses the long term dynamicity and supports the architecture exploration by integrating the concept of graph rewriting system to the modeling language. The last extension part covers the optimization of SoS architectures by changing them. This also supports the architecture exploration phase but integrates a solver for the optimization aspect directly.

The profile specified in the first version of this report (D6.5.1) turned out to be quite stable in term of the required stereotypes. The main changes have taken place in the GCSL part which could be reduced and in the Architecture Generation part as well where other relations as originally identified turned out to be relevant for the test cases.

2 Rhapsody Profile Implementation

2.1 Stochastic

A proposal of how to put stochastic data in the model has been integrated into the CAE. It is based on a set of attribute stereotypes that can be applied to any block attribute. This idea is close to the suggestion of the non-normative distribution extensions made in appendix of the SysML 1.3 specification, but adds the possibility to regenerate a distribution-based random value whenever needed (and not only at initialization), as explained in the next paragraphs. This addition is important because even the same person never performs the same task in the exact same amount of time, so that the duration of the task shall be recalculated every time.

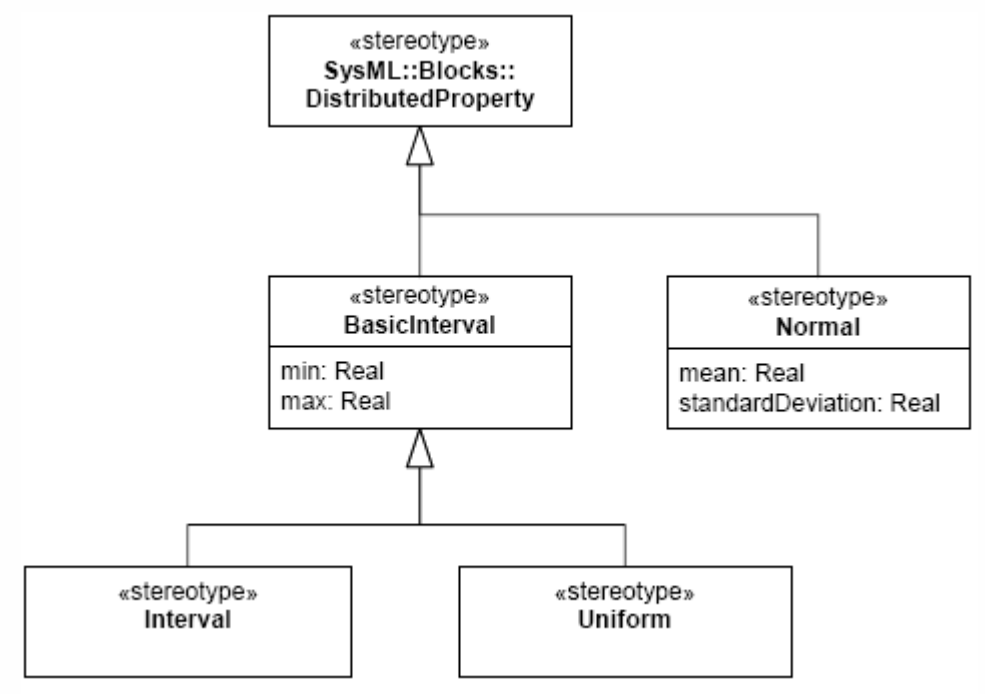

Figure 2-1: (SysML) Stereotypes to represent stochastic models

Before explaining more in-depth the stochastic proposal, here are the key drivers that lead to it:

- be simple enough for the end-user (the SoS architect)
- be easily readable and exploitable by the run-time technologies
- be consistent and make sense from a modeling point of view
- be able to use a single distribution to generate several random values

Stochastic behavior is classically represented by the concept of random variable. The probability distribution of the variable is interpreted as the probability that the variable takes up a certain value when it is observed. To follow this classical approach, given a random variable R that takes values over the reals and a real variable V, the assignment "V = R_observe()" can be seen as an observation of the random variable, provided that the "R_observe()" function is automatically or manually defined to generate new random values.

A simple example to illustrate this could be the statement "real initialForce = force_observe()", where all such assignment statements are interpreted as distinct observations of the variable force. For random timing delays, one would have to instantiate a random real variable "delay" and to use it as "tm(delay_observe())".

Based on these reflections, here is the list of the proposed stochastic stereotypes (inside the DANSE profile):

- **UniformRandomReal** → automatic addition of "RhpReal observe()" at code generation for generating a real based on uniform distribution
  - min:String
  - max:String

- **UniformRandomInteger** → automatic addition of "RhpInteger observe()" at code generation for generating an integer based on uniform distribution
  - min:String
  - max:String

- **NormalRandomReal** → automatic addition of "RhpReal observe()" at code generation for generating a real based on normal distribution
  - mean:String
  - standardDeviation:String

- **NormalRandomInteger** → automatic addition of "RhpInteger observe()" at code generation for generating an integer based on normal distribution
  - mean:String
  - standardDeviation:String

- **CustomRandomReal** → manual addition of "RhpReal observe()" to generate the random numbers (based on any kind of distribution)
  - customObserveFunction:String

- **CustomRandomInteger** → manual addition of "RhpInteger observe()" to generate the random numbers (based on any kind of distribution)
  - customObserveFunction:String

The reason for declaring all the attributes above as "String" is to be able to put expressions in there and not only numerical values (e.g. the mean value of the normal distribution of the city traffic is probably not fixed but dependent on the time of the day and the day of the week...). In addition to this, it would be worth considering the creation of an API allowing the overwriting of these expressions at any time.

Stereotype:	Applied to:	Meaning:

Stereotype:	Applied to:	Meaning:
UniformRandomReal	Attribute	The attribute is a real, generated according to a uniform distribution between ***min*** and ***max***
UniformRandomInteger	Attribute	The attribute is an integer, generated according to a uniform distribution between ***min*** and ***max***
NormalRandomReal	Attribute	The attribute is a real, generated according to a normal distribution with ***mean*** and ***standardDeviation*** parameters
NormalRandomInteger	Attribute	The attribute is an integer, generated according to a normal distribution with ***mean*** and ***standardDeviation*** parameters
CustomRandomReal	Attribute	The attribute is a real, generated according to a custom distribution defined by the user in ***customObserveFunction***
CustomRandomInteger	Attribute	The attribute is an integer, generated according to a custom distribution defined by the user in ***customObserveFunction***

Table 2-1: Stereotypes related to stochastic aspects

2.2 Simulation

The DANSE profile for SysML provides a set of stereotypes to integrate the SoS model with useful information to support the SoS model simulation and analysis.

2.2.1 FMI Stereotype

The FMI standard has been selected as the standard to support heterogeneous model composition and simulation within the DANSE project. SoS models are expressed as the composition of Constituent System (CS) models. Constituent Systems, specified using different languages and tools, are then exported as executable models compliant with the FMI standard (Funtional Mockup Units – FMU).

In order to correctly represent CS models within the SoS model, specified in UPDM/SysML Rhapsody, the DANSE profile has been integrated with stereotypes to support the SoS Model simulation.

In particular, the DANSE profile specifies a stereotype to assign an FMU nature to the CS blocks in SysML.

Table 2-4 reports the set of tags that the FMI stereotype for DANSE declares.

Stereotype:	***Applied to:***	***Meaning:***

Stereotype:	***Applied to:***	***Meaning:***
FMI	Blocks	Assign an FMI nature to a SysML Block constraints specification. Consists of the following TAGS: • Name • URI

Table 2-2: FMI Stereotypes

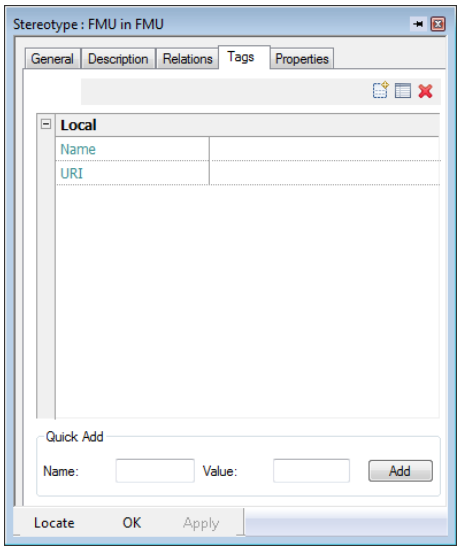

Figure 2-2 – SysML FMI Stereotype

The **Name** represents the name of the FMU file that represents the executable model of the SysML block. The **URI** TAG represents the location of the FMU file.

The FMI stereotype shall be applied to all SysML blocks that represent a Constituent System model with an associated executable model.

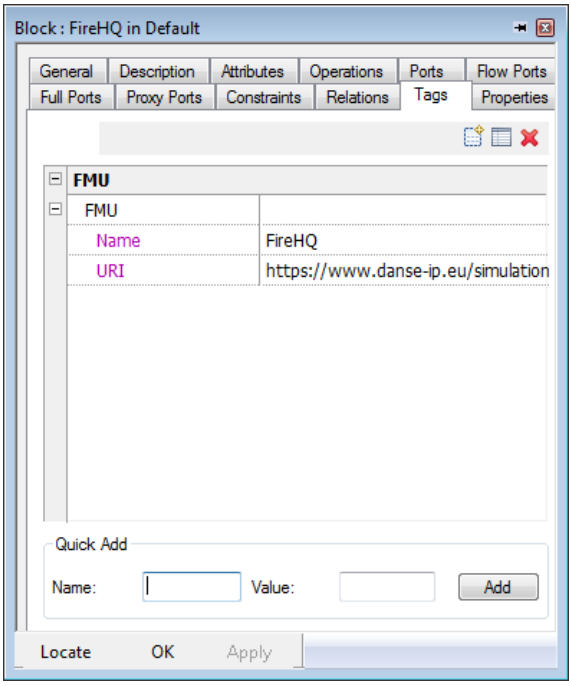

Figure 2-3 – FMI stereotype applied to a SysML Block

2.2.2 FMUParameter Stereotype

An ability to use of the shelf libraries and support reuse of components is critical for any modelling and simulation platform. This ability reduces the time required for the modelling and improves simulation quality by using standard and verified components.

One of the approaches to improve component reuse is to parameterize its behaviour, so in each concrete usage of the component the values of parameters will be set accordingly. FMI 1.0 standard introduces a special type of variability called “parameter”. A variable with “parameter” variability does not change after the initialization.

A new <<FMUParameter>> stereotype was created to mark variables of Constituent Systems that represent FMU parameters. <<FMUParameter>> applies on SysML attributes and doesn’t include any tags.

2.2.3 Traceable Stereotype

IBM Rhapsody for UPDM/SysML has been selected as the environment for the SoS specification and configuration in DANSE. The idea is that a SoS designer should be able to fully specify and configure the SoS structure, behavior and analysis from an unique framework. The scope of the Traceable stereotype, defined within the DANSE profile for SysML, is to allow the user to specify the set of model elements that the simulation back-end shall trace during the simulation.

The Traceable stereotype applies to SysML attributes and defines a single Boolean tag: ***TraceEnabled***. By default the ***TraceEnabled*** tag is set to false. Setting the flag to true will require the back-end simulator to trace the specified attribute during the model simulation (Table 2-3).

Stereotype:	***Applied to:***	***Meaning:***
TRACEABLE	Attributes	Enable tracing mechanisms for the selected attribute. Consists of the following tag: • TraceableElement

Table 2-3: Traceable Stereotypes

Figure 2-4 shows how the stereotype is specified and can be used in a SysML Rhapsody model.

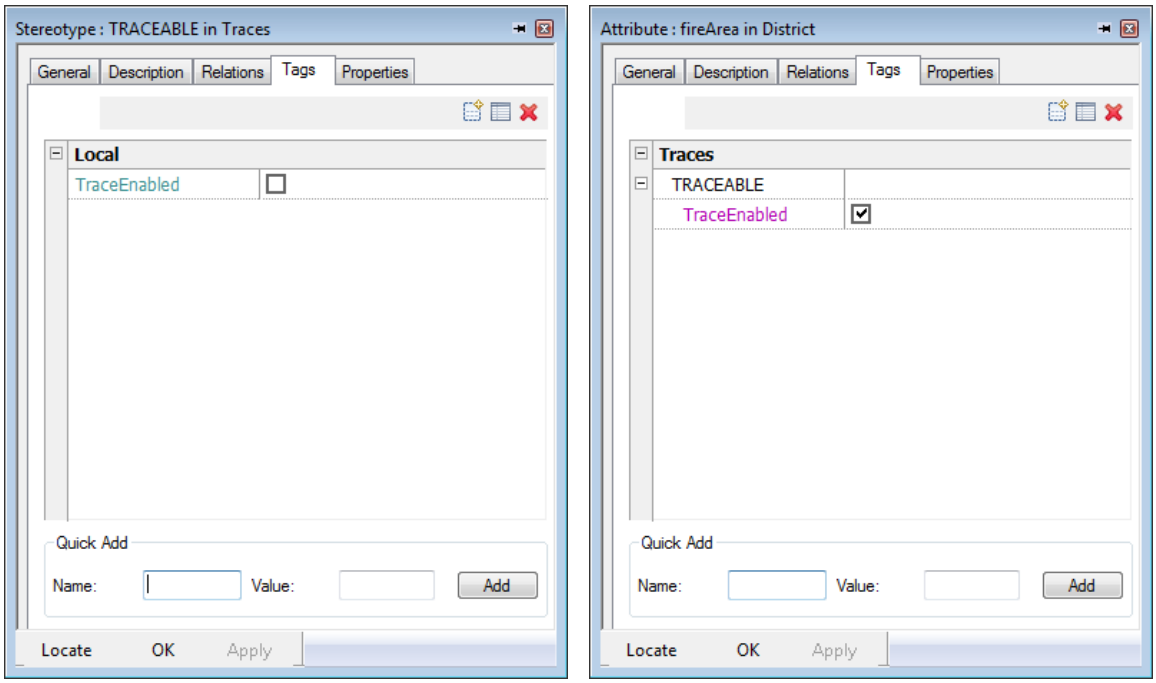

Figure 2-4 – TRACEABLE Stereotype definition (a) and use (b)

2.2.4 Metrics Stereotype (PROPOSAL)

The DANSE profile for SoS modeling defines a stereotype to enable the possibility to specify metrics on the SoS model. The simulation and performance analysis back-end should then be able to process such metrics and to provide results to the user.

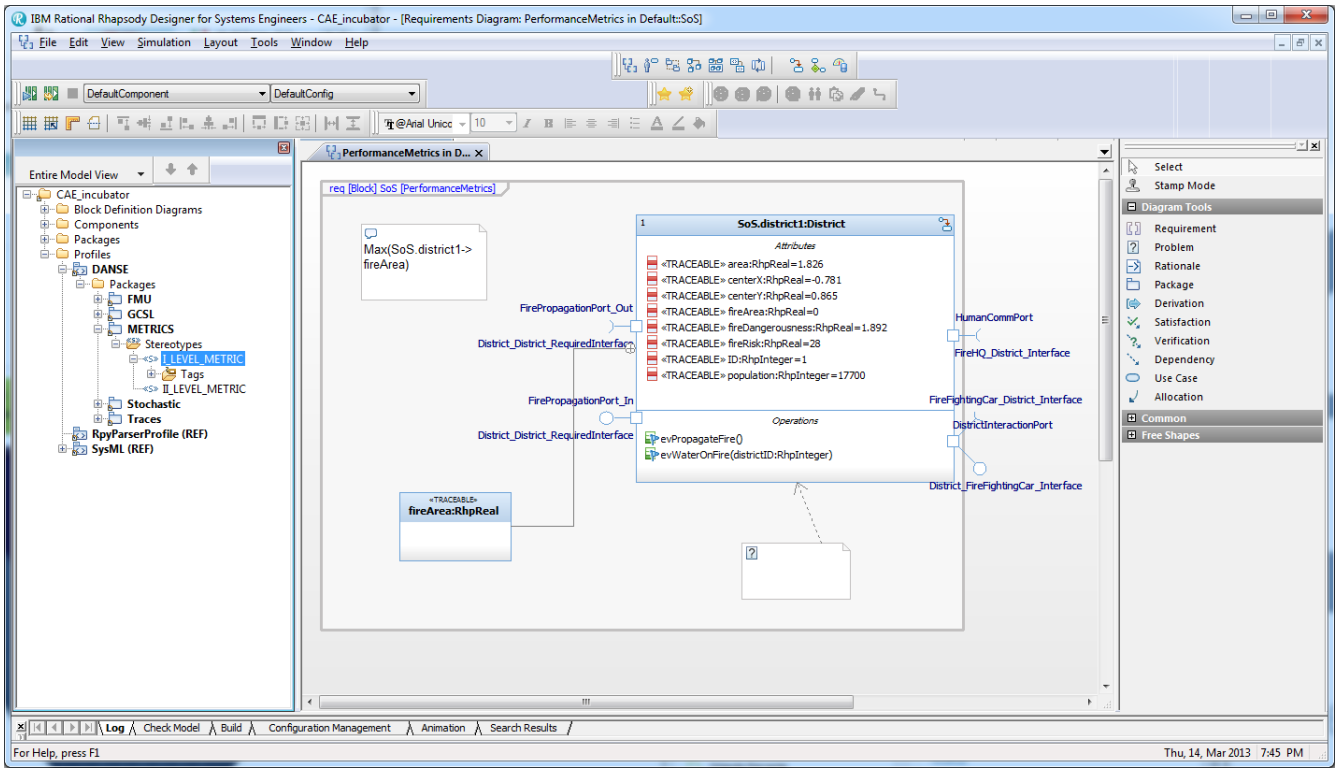

Figure 2-5 – Performance Metrics Diagram in SysML (Requirements Diagram)

The SysML Requirements Diagram (Figure 2-5) has been selected as the diagram to represent the metrics specification over the SoS model. Once the SoS model has been completed, the user can create a Requirements Diagram into the SoS model. The “Problem” block is used to represent a metric into the Requirement diagram. The DANSE profile for SysML defines the METRICS stereotypes to correctly represent METRICS that the user would like to evaluate with the simulation. Two stereotypes have been defined: I_LEVEL_METRIC and II_LEVEL_METRIC. Both stereotypes are applicable to SysML Problem Elements.

I Level Metrics are evaluated over system variables evolution during a single simulation run. II Level Metrics are evaluated over the results of I Level Metrics, calculated over a set of simulation runs.

Both stereotypes contain a tag called EXPRESSION that can be filled with the metric expression to be evaluated. The expression is entered as plain text; the syntax correctness of the expression is evaluated during the SysML model transformation.

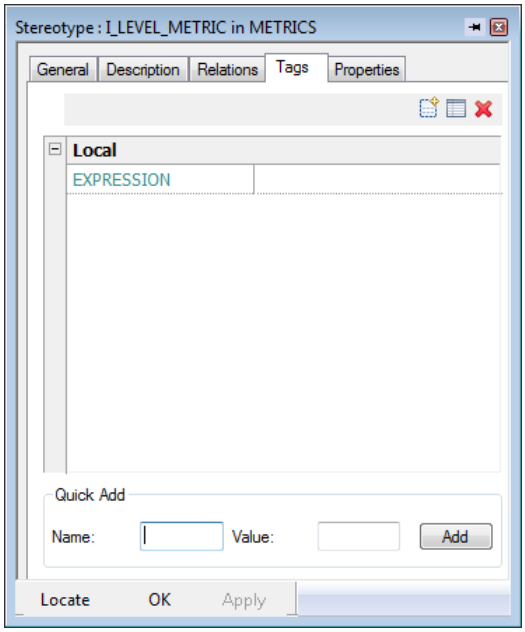

Figure 2-6 – METRICS stereotype TAG

2.3 GCSL

The DANSE profile for SysML allows the user to specify Goal and Contracts Specification Language (GCSL) constraints over the SoS model. The profile defines a GCSL stereotype that applies to SysML Constraint elements in order to represent a GCSL constraint. Table 2-4 shows the set of tag specified by the GCSL profile.

Stereotype:	***Applied to:***	***Meaning:***
GCSL	Constraint	GCSL constraints specification. Consists of the following TAGS: • GCSL_ID • Assumption • Guarantee • Probability Threshold • Condition

Table 2-4: GCSL Stereotypes

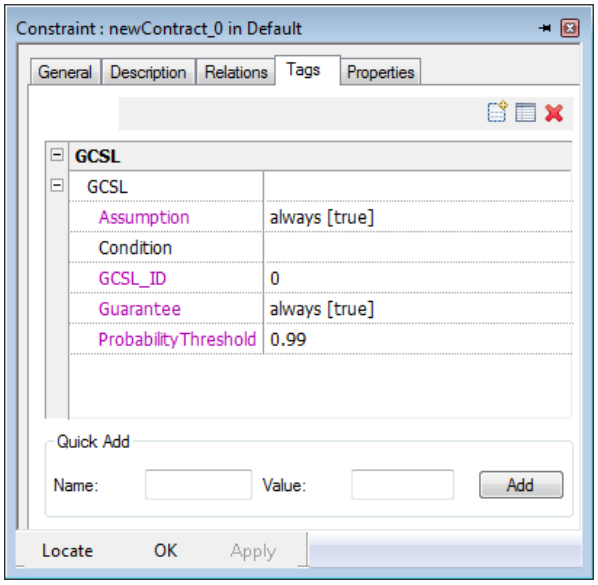

Figure 2-7 – SysML GCSL Stereotype

The **GCSL_ID** represents the identification of the expression; it should be unique for each internal block diagram (IBD). The **Assumption** and the **Guarantee** represent the GCSL expressions of the property that shall be verified; expressions are written as plain text and syntax check is executed during the model transformation phases. The ***Probability Threshold***, when specified, identifies a probabilistic goal and specifies the desired probability threshold for the goal expression. The **Condition** refers to the condition which guards the reconfiguration step of the dynamicity model.

The profile contains the documentation of the GCSL language as a plugin in Rhapsody to support the user while editing in Rhapsody. The GCSL – Editor (D6.4.2) provides a higher level of user support including syntax checking of GCSL expressions.

2.4 Architecture Generation

The architecture generation requires defining (local) changes which can be applied to an existing architecture to generate a set of alternatives. This is done by the concept of “StoryCharts” which are combined left and right hand sides of a transition rule. In the Rhapsody profile the stereotype “Rule” enable to create such charts. Since the UPDM views are defined for Rhapsody “Structure Diagrams” and “Object Model Diagrams” we need two custom views depending on these basic Rhapsody elements. The custom views allow using the same set of UPDM elements as the views DANSE is currently focusing on; namely the OV2 and SV1. Note that this is only a usability issue not a general limitation.

In the rule diagrams, UML elements are used as proxy elements which refer to the elements of the UPDM model. This is required because several rules may refer to the same model element and the rules shall not impact the model itself directly.

To create a rule element which refers to a certain model element (e.g. a certain type of CS) a proxy element is created by drag'n'drop from the drawing view into the rule diagram. To this element a stereotype is associated which can be a "Reader", "Creator", "Eraser" or "Embargo". The element itself are of type "Class", "Object" or "Flow" which are basic elements in Rhapsody. To ensure that the proxy element is of an type defined in the UPDM model a tag is used to refer to the type of UPDM elements. For example one would not only define "fireStation1" as a "Eraser" but also to define a "FireStation" type. This applies probably only for model elements like Classes or Objects but not for relations among them.

Stereotype:	Applied to:	Meaning:
Rule	StructuralDiagram	Custom diagram to specify a transition rule as StoryChart
Reader	Class	Tags an object or connection as required but unchanged
ReaderLink	Flow	
Creator	Class	Tags an object or connection to be created by the application of the transition rule: • Objects have a UML Class referenced to specify the type of the instance which is created • Flows have a enumeration associated which allows to specify the type of connection to create
CreatorLink	Flow	
Eraser	Class,	Tags an object or connection to be removed by the application of the transition rule
ErasorLink	Flow	
Embargo	Class	Tags an object or connection as forbidden elements for a match of the transition rule
EmabrgoLink	Flow	
!=	Flow	This link indicates that two matched objects must be not identical.

Table 2-5: Transformation related stereotypes

To refer to a meta model element for a relation between objects we need the meta types explicitly in the rule. Therefore an enumeration of meta model relations is provided to be used in the rule diagram. Currently the following set of relation types is included in the profile and can also be specified via a tag provided by the stereotypes applicable to Flows:

- ResourceInteraction
- Command
- Control

Further details can be taken from the DANSE Architecture Generation Tool manual.

2.5 Concise modeling

2.5.1 Introduction

The concise modeling is based on the SysML notation, which makes it convenient and well-pluggable into existing modeling environments. Working with concise modeling requires a concise profile in Rhapsody, and a Java plug-in that implements the smart behaviors required to work with the model, to prepare the database, to depict manual solutions, and to run the design space exploration.

Concise modeling adds few stereotypes and tags in order to expand the SysML notation and turn it into a concise modeling notation. Once we apply certain stereotypes to blocks, parts, arrows, constraints, attributes, etc, the concise plug-in knows they should be treated in the processing. Tags and stereotypes which are expanding standard SysML profile enlisted in concise profile. These stereotypes are required and must be applied in all relevant places to provide information to the concise plug-in that the corresponding elements are used in concise modeling. Elements that do not have the required stereotypes are "invisible" for the concise plug-in and cannot be treated as part of the concise model. The other side of the same fact is that one could decide we have a model that has packages and blocks that are part of a concise model and also not part of a concise model in a single model in Rhapsody.

The creation of the optimization model and data is performed by the concise plug-in and the program called "FreeMarker". FreeMarker is a very convenient and efficient way to create an optimization model and data files (*.mod, *.dat) from the model and data template files (*.tmod, *.tdat), using the Java API that provides information about the concise model. Usage of the template makes the model creation convenient and flexible. It enables easy introduction of "upgrades" to the concise modeling without a need to touch the plug-in source code.

One of the ways for custom "upgrades" is to use custom-defined stereotypes that may appear in the concise model. Once these stereotypes appear in the model, the template logic obtains information about elements, attributes, and links relevant to the specific topic. Then, creation of elements and constraints in optimization model for implementation of additional logic is possible. Since the set of these custom stereotypes is an essential extension of the SysML profile we include to this document descriptions for stereotypes which are processed by current implementation of the model template file.

2.5.2 Concise profile stereotypes

Stereotypes related to modeling layers:

The concise model has a hierarchical structure with 2 or 3 levels of abstraction, where the first level, the Functional level, actually shows the requirements, and Technical and Geometrical Levels describe what block can / should realize the requirements. The Technical and Geometrical levels of abstraction actually form one Physical level of abstraction together.

Stereotype:	Applied to:	Meaning:
functional	part, connector	The element belongs to the "Functional" Layer in the concise model
technical	part, connector	The element belongs to the "Technical" Layer in the concise model
geometrical	part, connector	The element belongs to the "Geometrical" Layer in the concise model

Table 2-6: Stereotypes related to modeling layer

Stereotypes related to connections between layers:

Very important parts of a concise model are definitions of interconnections between the abstraction levels. Often, they have the most important definitions of how the expanded or optimized model will look like. The table below contains stereotypes that can be applied to dependencies which are often serves as interconnections between abstraction layers.

Stereotype:	Applied to:	Meaning:
mappedTo	dependency	Applied to the dependency arrows that map the functional elements to the technical (physical) elements
allocatedTo	dependency	Used to denote an allocation dependency, usually between element in the Technical and Geometrical layers

Table 2-7: Stereotypes related to connections between layers

Stereotypes related to data structure:

The concise plug-in uses a database structure in order to supplement the concise model information. It means that the whole "picture" of the expanded model is composed of the information in the concise model and in the database. In order to create these structure and use database based on this structure specified stereotypes must be applied to the corresponding elements. Another extension of the SysML profile needed in concise modeling is connectors that actually have physical implementation (cables, pipes, etc.). Stereotypes listed below related to database structure and physical connectors.

Stereotype:	Applied to:	Meaning:
catalog	block	Means that the part of this type (of this block) will be chosen from the database. A database list will contain a list of elements, all having the same properties, but with various values. Like a catalog page of LEDs. Each

Stereotype:	Applied to:	Meaning:
		of them is a led, but the optimization engine will choose the one specific type that fits the most (for optimization objectives), under the constraints (if any).
	attribute	[The block having any attribute equipped with this stereotype also must have stereotype <<catalog>>] Attribute equipped with this stereotype is taken from the database table that is created for the block it is part of. In order to make the purpose of using this stereotype clearer, we remember that in the same part there also could be attributes with other stereotypes. For example, the attributes that are calculated using values of other attributes are marked with <<derived>>, <<optimized>> stereotypes.
inventory	part	Due to this stereotype, a single element in a concise model represents a whole list in a database. The concise plug-in and the optimization take into account the whole list, whenever it deals with a part marked by the <<inventory>> stereotype.
	attribute	For a part that is marked by the <<inventory>> stereotype, the value for each of the attributes marked by the <<inventory>> stereotype is taken from the database.
	connector	When a connector is marked by the <<inventory>> stereotype, a table of such connectors is created. All the instances of the connectors that are listed in the database are used in the optimization as potential connectors ("paths").
Typed Connector	connector	The attribute marked by this stereotype equipped with the "type" tag which must be set for model blocks. The meaning is that the paths between the items in the physical layer will not be just some connectors, but each will be a part of block [of the chosen type], and the attribute values of these connectors will be filled in an appropriate way.

Table 2-8: Stereotypes related to data structure

Stereotypes related to optimization goals and constraints:

Stereotype:	Applied to:	Meaning:

Stereotype:	Applied to:	Meaning:
optimized	part, connector, attribute, dependency	This stereotype is applied to elements that are being optimized by the optimization engine (for example, when the connection paths are decided on, or when the specific type is chosen from the catalog, or when the attribute is a part of objective function)
sow_goal_ attribute	attribute	This attribute is one of the objective functions in the optimization of the model. The attribute marked by this stereotype equipped with following tags: Action: can be minimize or maximize depending on optimization goal. Description: string described the optimization goal. isEnabled: can be true or false. Setting this attribute to be false "turned off" selected optimization goal. Priority: set priority for selected optimization goal.
derived	attribute	This attribute will be calculated using the attached constraint, rather from a database. Must be applied together with <<optimized>> stereotype.
sow_assign ment	constraint	Constraint will be used to calculate value of the attribute. Must be applied to constraints attached to attributes having the <<derived>> stereotype.
.sow_ constraint	constraint	Used to provide additional features to constraints that is marked with <<sow_optimization>>. The attribute marked by this stereotype equipped with following tags: Description: string described the constraint. isEnabled: can be true or false. Setting this attribute to be false "turned off" selected constraint. isVisible: can be true or false. Not used for optimization.
sow_ optimization	constraint	This constraint is part of the optimization

Table 2-9: Stereotypes related to optimization goals and constraints

Other stereotypes included in the concise profile:

Stereotypes below are currently not used or experimental and therefore these should currently not be applied to any elements.

Stereotype:	Meaning:
FailureCase, failureDependency,	Experimental and not fully implemented. Related to reliability algebra. Work on this algebra is in progress.

Stereotype:	Meaning:
spuriousDependency	
multiple	Experimental and not fully implemented. Applied to connectors. Connector equipped with this stereotype has "strategy" tag with can be set to one of multiplication strategies "one-to-one" or "all-to-all". Currently implemented only "all-to-all" strategy. Applying this stereotype to connector on functional level implies that all physical parts for which mapped functional parts connected by this connector must be directly or indirectly connected to each other.
sowModel	Not used. Basic class for all concise stereotypes.
conciseClass	Not used.
expand	Not used. Developed for some back-annotation features.

Table 2-10: Other stereotypes included in concise profile

2.5.3 Additional extensions

The custom stereotypes are defined in a flexible way. It means that the template looks for stereotypes that have a specific **prefix**, and a user defines custom **suffix** for the stereotype, according to a specific usage. The concise plug-in automatically adds necessary constraints to the optimization task when the processing model has prefixes which are defined in template (.tmod) file. Below we present list of prefixes related to various algebras currently implemented in template file.

Stereotypes related to resource algebra:

Resource algebra is used when some functionality (nodes defined on functional level) requires corresponding technical resources (from nodes defined on technical level) and information about quantities of these technical resources that can be given by each instance of specific type stored in database.

Stereotype prefix:	Applied to:	Applied with:	Meaning:
resourceRequest_	attribute	<<inventory>>	Must be applied to attribute of part on functional layer. This attribute specify request of specified quantity of specific resource.
resourceCapacity_	attribute	<<catalog>>	Must be applied to attribute of part on technical layer. This attribute specify quantity of specified resource supplied by specific part.
resourceUsed_	attribute	<<optimized>>	Applied on to attribute of part on technical layer. This attribute used to calculate quantity of resources used

Stereotype prefix:	Applied to:	Applied with:	Meaning:
			by parts having attributes with <<resourceRequest_>> stereotype from resource supplied by part equipped with this attribute.

Table 2-11: Stereotypes related to connections between layers

Stereotypes related to coverage algebra:

Coverage algebra is used when we need provide some coverage constraints based on some predefined coverage data. I.e. we have data that for each instance of coverage provider placed in each possible geometrical location defines coverage of all possible areas that require such coverage (coverage requesters).

Stereotype prefix:	Applied to:	Applied with:	Meaning:
coverageRequester_	attribute	<<optimized>>	Applied to attribute of technical part that allocated to some geometrical location which can be "covered".
coverageProvider_	part	<<technical>> and <<optimized>>	Applied to part that provides coverage.
coverageDataProvider_	attribute	<<inventory>>	Attribute of <<coverageData_>> part which contains id of specific type of coverage data provider or id of geometrical location to which this provider can be allocated. Two attributes with these stereotypes are needed for each <<coverageData_>> part (one defines type of provider and second defines geometrical location for this type of provider).
coverageDataRequester_	attribute	<<inventory>>	Attribute of <<coverageData_>> part which contain id of specific geometrical location which can be covered by specific type of coverage provider (given by <<coverageDataProvider_>> attribute) allocated to specific geometrical location (also given by <<coverageDataProvider_>> attribute).
coverageDataValue_	attribute	<<inventory>>	Attribute of <<coverageData_>> part which defines relative coverage of specific geometrical location (given by <<coverageDataRequester_>> attribute) which covered by specific type of coverage provider

Stereotype prefix:	Applied to:	Applied with:	Meaning:
			(given by <<coverageDataProvider_>> attribute) allocated to specific geometrical location (given by <<coverageDataProvider_>> attribute).
coverageData_	part	<<technical>> and <<inventory>>	Applied to part that is database table which provides coverage information. This part must contain at least 4 attributes: 2 with <<coverageDataProvider_>>, 1 with <<coverageDataRequester_>> and 1 with <<coverageDataValue_>> stereotypes correspondingly.
coverageAggregator_	attribute	<<optimized>>	Must be applied to attribute which used to calculate total relative coverage.

Table 2-12: Stereotypes related to coverage algebra

Stereotypes related to interface capacity algebra:

Interface algebra is used when some technical element have limitations on number of possible connections of specific type and information about these limits stored in database.

Stereotype prefix:	Applied to:	Applied with:	Meaning:
interfaceRequest_	connector	<<technical>> and <<optimized>>	Connector equipped with this stereotype is subject of interface capacity constraint. Connector must be connected to part which contains attribute equipped with <<interfaceMaxCapacity_>> or <<interfaceUsedCapacity_>> stereotype having same suffix.
interfaceMaxCapacity_	attribute	<<catalog>>	Set maximum number of connections of specific type (having << interfaceRequest_>> stereotype with same suffix) allowed for the part containing attribute equipped with this stereotype.
interfaceUsedCapacity_	attribute	<<optimized>>	Attribute equipped with this stereotype used to calculate number of connections of specific type connected to part containing this attribute.

Table 2-13: Stereotypes related to interface capacity algebra

Stereotypes related to geometrical allocation algebra:

Geometrical allocation algebra used when some geometrical locations allow placing of multiply technical parts and information about maximal numbers of technical parts that can be placed in the specific geometrical locations stored in database.

Stereotype prefix:	Applied to:	Applied with:	Meaning:
geometricalMaxAllocation_	attribute	<<inventory>>	Attribute equipped with this stereotype set maximum number of parts from technical layer that can be allocated to specific place (part containing this attribute on geometrical layer).

Table 2-14: Stereotypes related to geometrical allocation algebra

Stereotypes related to energy flow algebra:

Energy flow algebra used when we need add constraints on energy flow in the network. In this case we have parts that request specific quantity of energy, parts that can supply specific maximal quantity of energy and various parts (including typed connectors) which participate in energy transmission process. The information about required demands, maximal supplies and maximal transmission rates for various types of various parts are stored in the database.

Stereotype prefix:	Applied to:	Applied with:	Meaning:
energyDemand_	attribute	<<catalog>>	Must be applied to attributes of parts that are **consuming** the distributed energy. Attribute with this stereotype define demand rate of energy.
energyDistribution_	part	<<technical>>	Must be applied to all the parts that participate in energy distribution.
energyCapacity_	attribute	<<catalog>>	Must be applied to attributes of "main" energy sources. Attribute with this stereotype defines the maximal ability to provide energy (similar to <<energyMaxOut_>>).
energySupply_	attribute	<<optimized>>	Must be applied to attributes of "main" energy sources. Attribute with this stereotype calculates how much energy is actually used.
energyIn_	attribute	<<optimized>>	Attribute with this stereotype calculates the actual inflow of energy.
energyOut_	attribute	<<optimized>>	Attribute with this stereotype calculates the actual

Stereotype prefix:	Applied to:	Applied with:	Meaning:
			outflow of energy.
energyMaxIn_	attribute	<<catalog>>	Attribute with this stereotype defines the Maximal inflow of energy
energyMaxOut_	attribute	<<catalog>>	Attribute with this stereotype defines the Maximal outflow of energy
energyPassed12_	attribute	<<optimized>>	Must be applied to attribute of the block that "implements" the connector (block which type set for some connector having <<TypedConnector>> stereotype, for example "Power_Cable") Used for calculation of energy that flows through this connector. It is advised to use only one direction: energyPassed12_. It is very important to note that the connections in the technical diagram must all be in the same direction in order to make it work. The correct direction will be obtained if while creating a connector, you first click on the source element and then on a target element. Quite Counter-intuitively, Rhapsody will then name the source element "End2", and the target element "End1", however, this is the correct direction you need to use with stereotype <<energyPassed12_...>>.
energyPassed21_			

Table 2-15: Stereotypes related to geometrical allocation algebra

Stereotypes related to energy drop algebra:

Energy transmission in the real systems often causes energy drop on elements of energy distribution system (technical parts and typed connectors). Energy drop algebra used to add constraints on possible energy drop in the system. This algebra is fully independent from energy flow algebra and must be provided separately. At current state this algebra is experimental and can be subject of additional changes. We recommend avoiding usage of this algebra before stable version of it will be released.

Stereotype prefix:	Applied to:	Applied with:	Meaning:
energyDropDelta_	attribute	<<catalog>>	Must be applied to attribute that characterize energy drop per length of the energy carrier (connector).

Stereotype prefix:	Applied to:	Applied with:	Meaning:
energyDropNode_	part	<<technical>>	Must be applied to all parts where energy drop occurs.
energyDrop_	attribute	<<catalog>>	Must be applied to attribute that characterize total energy drop on specific part (and independent on part dimensions).
energyMaxDrop_	attribute		Not used. Reserved for future extension.
energyDropAggrigator_	attribute	<<optimized>>	Must be applied to attribute that aggregates total energy drop in the system.
energyDropMaxAggrigator_	attribute	<<catalog>>	Must be applied to attribute that represents maximum energy drop in the system.
energyDrop12_	attribute	<<optimized>>	Must be applied to attribute of the block that "implements" the connector (block which type set for some connector having <<TypedConnector>> stereotype, for example "Power_Cable") Used for calculation of energy drop that occurs on this connector.
energyDrop21_			

Table 2-16: Stereotypes related to energy drop algebra

3 Conclusion

This extension of the UPDM profile in Rhapsody enables the simulation of models interconnected according to UPDM models and therefore the application of the DANSE modeling methodology. The extension contains additionally means to annotate goals and contracts, modeling refinements to support concise modeling and means to describe architectural changes.