SysML 2 Analysis WG Updates

Updates to SysML 2 RFP / Nov 16, 2016

Team: Manas Bajaj (Intercax), Ahsan Qamar (Ford), George Walley (Ford), Bjorn Cole (JPL), Bill Bailey (Ford)

1. Scenarios (Automotive Example)

Our goal is to develop the workflows for the 3 challenge problems related to HSUV. We will then derive the services required in the System Modeling Environment (SME) to support that workflow. From there, we will derive the analysis concept models and capabilities required in the SME. To summarize, our progression will be as follows:

Analysis Workflows (HSUV Challenge Problem) → SME Services → Concept models and capabilities

1.1. Scenario # 1: Govt. regulation to improve fuel efficiency

#Aerospace - For spacecraft, this could be a need to gather more data per time period, which pushes on efficiencies of downlink, power use, or tasking of instruments.

(1) Locating and gathering the baseline model package — Total System Model

a. Checkout the latest version (baseline) of the Total System Model package. This includes the architecture model (SysML), connected design models (CAD/PLM/ALM), connected analysis models (simulation/FEA/CFD), and connected requirement models of the vehicle (system of interest) and the environment. The Total System Model is a consistent set of connected models that represent the vehicle.
b. Get the latest versions of analysis models used for computing fuel efficiency, the results of the last analyses, and the decisions taken thereof.
c. Where are the latest on-field test results for fuel efficiency?
d. What is the discrepancy between the analysis model results and on-field results?
e. Have we identified factors that contribute to this discrepancy? Are they formalized so that we can account for those in the next set of analyses?

(2) Reviewing the architecture, create new design alternatives, and analyze

a. Understanding factors that influence fuel efficiency at the system level, such as mass, size, engine, powertrain efficiency, driving environment, driver behavior, and other factors.
b. Identify the parametric changes versus the topological changes that can be made to improve fuel efficiency.
    i. Parametric changes (can affect topology beyond a certain range)
    ii. Topological changes

iii. Combination

c. **Parametric changes** and related analyses

   i. Examples - Engine maps, Torque/speed correlation to fuel economy, Gear ratios, and Controls calibration. These could be available as: (1) multivariate functions that need to be evaluated simultaneously, and/or (2) tables from handbook, such as fuel rate = *f* (torque load, requested speed).

   i. Need to perform a trade study of torque, speed and fuel economy

   ii. Multiple control algorithms have to be in sync for the fuel efficiency to be optimum. Loss of synchronization between a shift control algorithm and the engine control algorithm would lead to a fuel efficiency loss. I think this is true for a lot of big system-level optimizations … lots of components and subsystems need to be coordinated.

   #Issue - Analysis models from the suppliers are usually black box models, not parameterized or you do not have control over the knobs that control them.

b. **Architectural / Topology changes** - Identify if the current architecture is capable of meeting fuel efficiency goals, if not, perform an architectural exploration study to find an architectural solution that would meet the fuel efficiency goal.

   i. Example of architectural changes -- Turbocharged engine Vs naturally aspirated Vs Hybrid; Controls Architecture (instead of controlling speed at the wheel, you control torque at the drive shaft)

   ii. What architectural changes will require rebuilding models (beyond the scope of existing analysis models) versus reusing?

   #Issue – How do you know that the changes (parametric or topological) will not break existing analysis models? How do you capture the analysis intent, when model is built, so that we can verify if the assumptions in the model still hold true for the new scenarios? Capture the analysis intent describing the assumptions and goals of the analysis. For example, a particular analysis used a fixed step solver with 2 msec interval and if this is not communicated to domain experts, they may build analysis models that run with a different configuration, leading to issues while integrating**.** Similarly, if a domain expert makes an engine model and does not specify that it is not to be used for cold cycle simulation, could lead to errors. In Simulink, assumptions by the modeler are either not made explicit, or are specified sometimes twelve levels down. ***We need a property-based / math-based description of the assumption and not text-based, e.g. all analysis models in the chain of analyses to compute fuel efficiency are valid from -10C to 45C operating temperature range.***

c. **Analysis planning** – Planning the analyses to compute fuel efficiency for parametric and topological changes.

   i. What is the order in which we will explore the parametric and topological changes?

   ii. What is the level at which changes will be affected – single component, sub-assembly, whole system, hardware-only, software-only, etc.

   iii. What is the order in which we will run the analyses for each change? For example, running analysis at different scales – isolated component versus sub-assembly versus vehicle,

   iv. What additional analyses would we run to check that we don't fail other requirements while we try to analyze the impact of changes to fuel economy.

   v. When are you done?

It would be helpful we had an "**Architectural Distance**" model to give a sense of degree of changes, such as how extensive are they in terms of number of element affected, disciplines affected, and the range of analyses that we have to be re-run. This is similar to what is called platform or program or technology scale based on degree of change to underbody, upper-body, etc. elements of the vehicle. Maybe if computed this could help confirm or alert when changes have been made beyond the intended scale of change allowed, etc.

d. **Executing the analysis models based on the the analysis plan**
   i. Parametric analyses - Perform a sensitivity study to figure out which parameters effect the fuel efficiency, based on a given architecture description, e.g. given a hybrid-electrical vehicles architecture, find the parameters that affect fuel efficiency.
   ii. Trade studies – Perform parametric and architectural trade studies per the analysis plan to compute fuel economy for various scenarios.

e. **Visualization and review of analysis results**
   i. Stack analysis results against each other to view the "analyzed" impact of parametric and topological changes to the fuel economy.
   ii. Compare the gains in fuel economy to the estimated cost and completion time of the vehicle program.

f. **When do you stop the analysis?**
   i. Regulatory requirements vs vehicle program attributes, dictate when the analysis is stopped. A regulatory requirement must be met, vs a market demand, fo which we can be in the nearby/ballpark figure. In short, the analysis is stopped when it is no longer valuable to further continue based on the cost/benefit tradeoff.

(2) Perform field tests

a. Order and install new parts in the test vehicle
b. Perform field tests under same operating conditions as analysis
c. Compare on-field results versus as-analyzed results, and recheck the factor
d. If on-field results agree with as-analyzed results, plan a broader test, and eventual rollout
e. Setup formal test cases, e.g. a specific parameter is within a specific percentage, validated through a particular test case.
f. Comparing test data with the parameters in the analysis, e.g. transient signals.
g. Query a test repository for a test case data, pull down data, but sometimes certain signals that are required to compare the analysis are not in the data. Need to estimate those signals from the ones that are measurable.

1.2. Scenario # 2 -- I&T provides data stating that vehicle fails to meet its fuel efficiency requirement

(1) Locating the baseline results and models

a. Get I&T results, and identify as-built test vehicle, fuel efficiency measurement approach, and test environment (driving conditions, driver behavior,....)
b. Gather as-designed information (BOM / architecture), analysis models and as-analyed fuel efficiency results

(2) Review the results and planning out next steps

a. Do the fuel efficiency measurement approaches on-field and as-analyzed match?
b. What is the different margin between on-field and as-analyzed results? Is this margin normal based on past vehicle designs? If yes, did the design team not account for this?

(3) Updating the design

a. Follow step 2 in Scenario 1

(4) Perform new on-field tests

a. Follow step 3 in Scenario 1

2. Services

SysML 2.0 based SME shall provide the following services. Analysis implies:

1) Quantitative analysis, such as computing KPPs and MoEs to evaluate system alternatives, or quantitative impact analyses to assess if and by how much a system architecture (structure or function) may need to change if a requirement changes.

2) Qualitative analysis, such as done using graph traversals to assess how changes in part of the system "may" affect others parts of the system.

3) Model V&V, such as checking if all requirements are allocated to the system definition, or if there is a test case associated with every requirement, or matching patterns and anti-patterns that may be relevant for a specific type of system or discipline.

An analysis activity may include evaluation by means of modeling/simulation, inspection, demonstration, test, or a combination of these.

Analysis service bundle includes services related to analysis setup, analysis execution, analysis data and model management, and analysis decision management. These are elaborated below.

1. **Setup Analysis**
   a. Model the types of analyses that need to be performed on the system representation
   b. Model the analysis objectives mathematically
   c. Define the key parameters (KPPs/MoEs) being computed or patterns/anti-patterns to be matched.
   d. Define mapping/transformation from system model to analysis model
   e. Create or Generate analysis model based on mapping/transformation (tool-neutral or tool-dependent)

2. **Execute the analysis**
   a. Define the analysis execution/process – What are the inputs, what is the context/scenario?
   b. Execute the analysis model in a solver tool

3. **Commit models and results to the analysis repository**
   a. Commit the analysis model, run conditions, analysis results, and visualizations together as a set to the analysis repository
   b. Relate the analysis model run to the system representation

4. **Decisions and Change Request**
   a. Model decisions / change requests based on the analysis result
   b. Connect the decision / change request to the analysis model/result set

5. **Query analysis results**
    a. Formulate a query to retrieve the analyses
        i. Based on a given system
        ii. Based on a specific decision
        iii. Based on a specific analysis model
        iv. …and more
    b. Fetch the analysis model set from the repository
    c. Visualize and browse through the analysis model set

3. Concept Model

SysML 2.0 shall provide analysis concepts defined in the Analysis Concept Model.

The *Analysis Concept Model* provides a foundational meta-model for system analyses. It includes concepts relevant for formulating and solving system analysis models, and the inter-relationships between these concepts. We present the core aspects of the Analysis Concept Model here.

Analysis Concept Model

In this section, we present the Analysis Concept Model as a foundational meta-model for computer-based representation of system analyses. Figure 1, Figure 2, and Figure 3 illustrate the SysML model for the Analysis Concept Model, which is also included with this report.

The concept model was developed by a grant project between NIST (Conrad Bock) and Intercax in 2014-2016 timeframe.

System Analysis is the fundamental concept in the Analysis Concept Model, as shown in Figure 1 below. It represents the information required to formulate and solve system analysis problems. The various types of system analyses, as discussed in section are modeled as subtypes of the System Analysis concept. The classification of the System Analysis concept into the subtypes is based on (1) aspect/measure of the system being analyzed, e.g. performance and other MoEs for the Effectiveness Analysis or cost measures for Cost Analysis; and (2) computational process for the analysis, e.g. trade-off analysis requires computation and comparison of multiple MoEs versus cost, and optimization searches for a system design alternative that minimizes/maximizes objective functions defined using MoEs and cost.

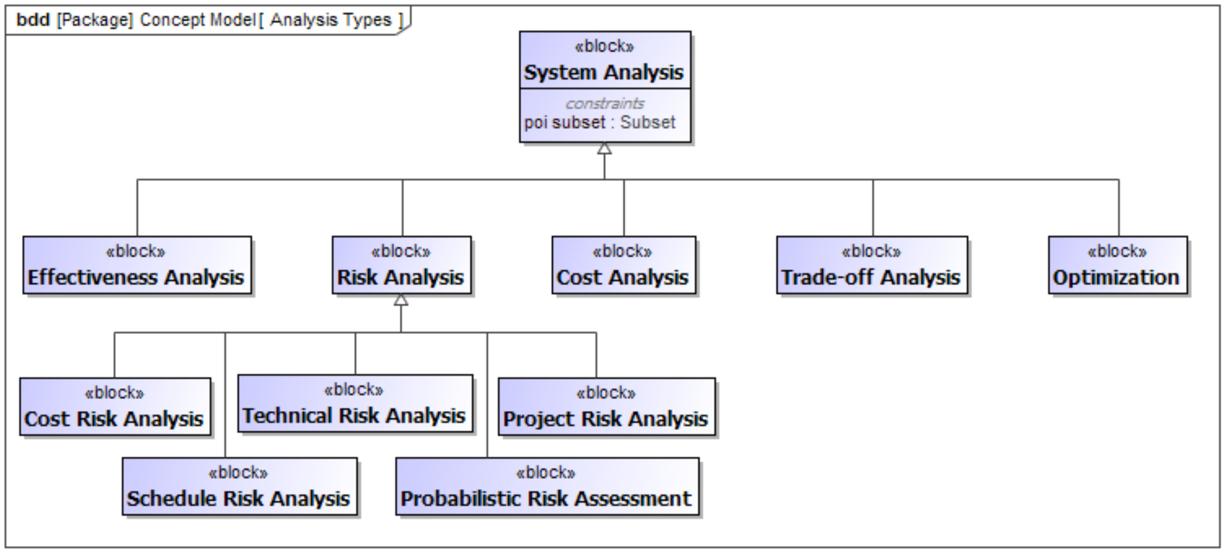
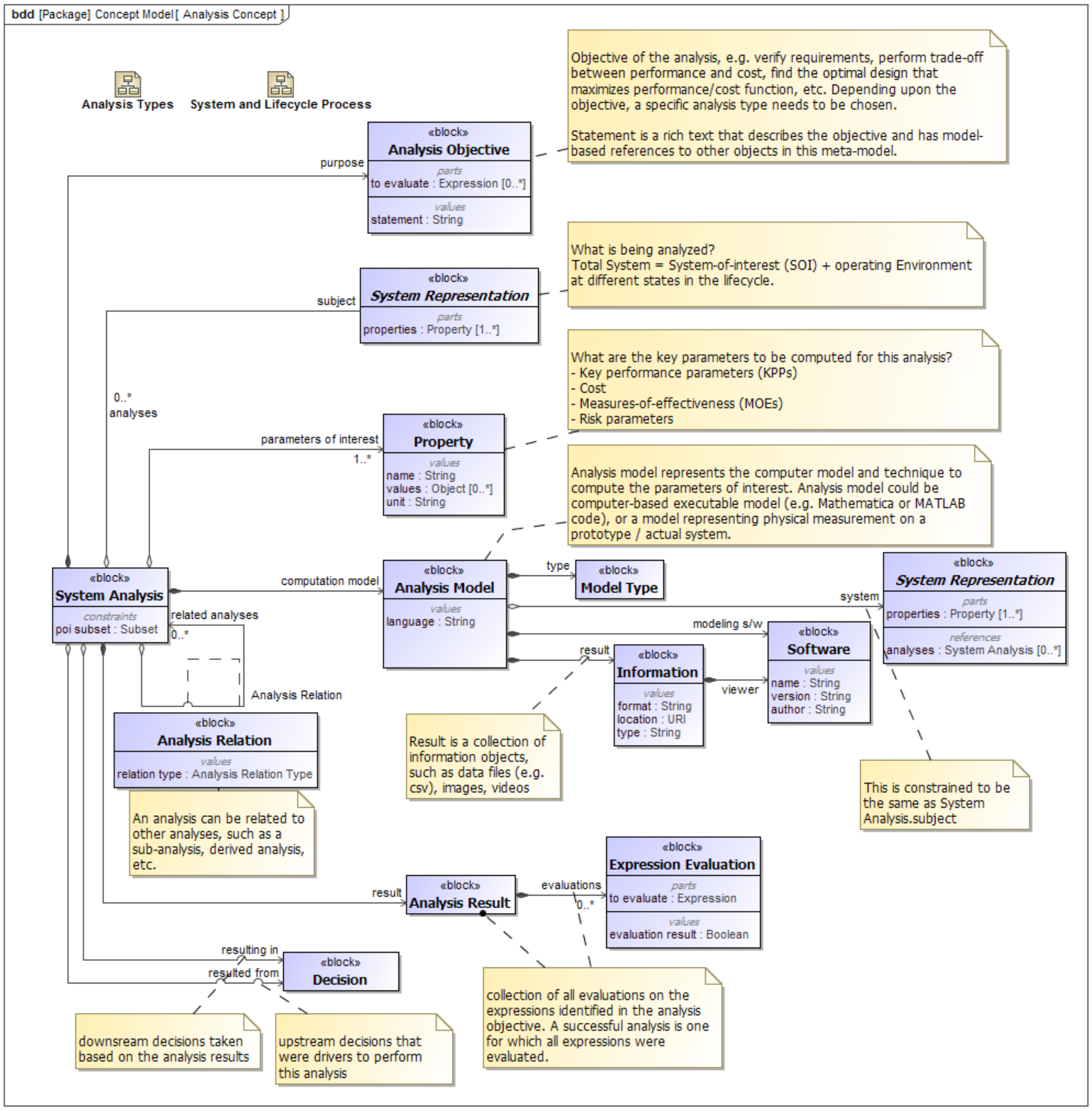


Figure 1: Types of system analyses (SysML BDD with taxonomy of analysis types)

The *System Analysis* concept is modeled in detail, as shown in Figure 2 below. The following concepts are defined.

Analysis Objective concept represents the objective of the analysis. The objective is modeled using a set of math expressions that can be formally evaluated and a description. The objective of an analysis is met if the expressions can be successfully evaluated by the information generated during the analysis. For example, if the analysis is being performed to verify a performance or mass requirement, then the math representation of that requirement will be used for the expressions that that need to be evaluated, such as *acceleration > 10* m/s^2 or *mass < 1000 kg*.



Objective of the analysis, e.g. verify requirements, perform trade-off between performance and cost, find the optimal design that maximizes performance/cost function, etc. Depending upon the objective, a specific analysis type needs to be chosen.

Statement is a rich text that describes the objective and has model-based references to other objects in this meta-model.

What is being analyzed?
Total System = System-of-interest (SOI) + operating Environment at different states in the lifecycle.

What are the key parameters to be computed for this analysis?
- Key performance parameters (KPPs)
- Cost
- Measures-of-effectiveness (MOEs)
- Risk parameters

Analysis model represents the computer model and technique to compute the parameters of interest. Analysis model could be computer-based executable model (e.g. Mathematica or MATLAB code), or a model representing physical measurement on a prototype / actual system.

Result is a collection of information objects, such as data files (e.g. csv), images, videos

This is constrained to be the same as System Analysis.subject

An analysis can be related to other analyses, such as a sub-analysis, derived analysis, etc.

collection of all evaluations on the expressions identified in the analysis objective. A successful analysis is one for which all expressions were evaluated.

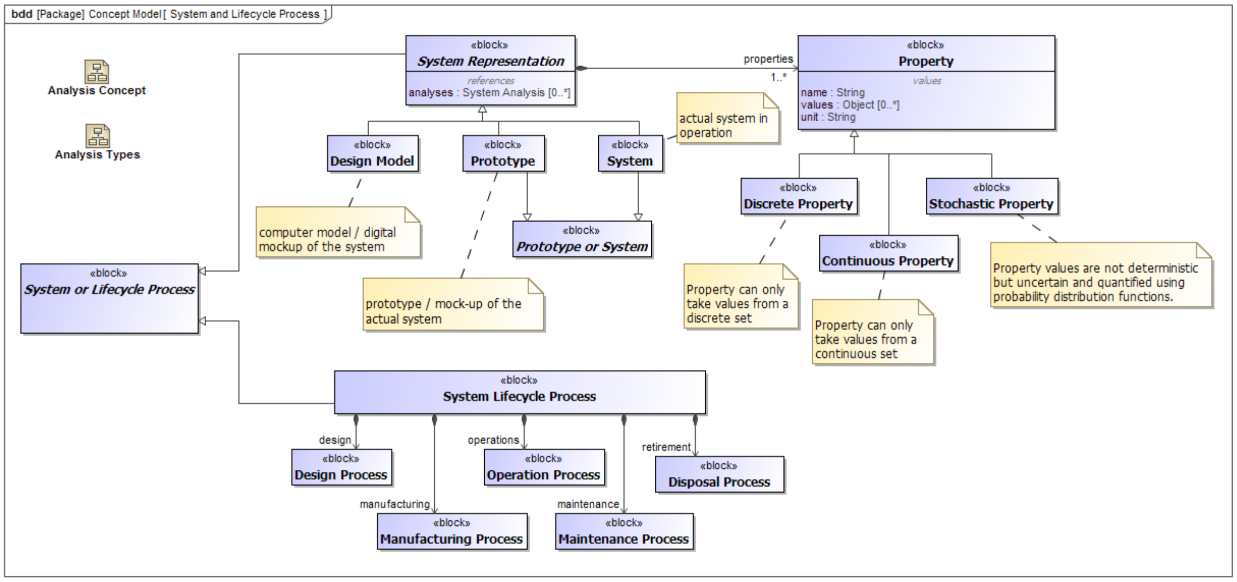
downsream decisions taken based on the analysis results

upstream decisions that were drivers to perform this analysis

Figure 2: System Analysis concept (SysML BDD)

System Representation concept represents the subject of the analysis being performed. Since the scope of system analysis spans across the lifecycle, the subject of the analysis could be either of the following, as shown in Figure 3.

(1) design representation of the system, such as a digital mockup of a spacecraft being developed
(2) prototype of the system, such as a scaled or real prototype of the spacecraft
(3) deployed system, such as the actual spacecraft deployed in orbit



bdd [Package] Concept Model [System and Lifecycle Process]
Analysis Concept
Analysis Types
«block» System Representation
references
analyses : System Analysis [0..*]
properties
1..*
«block» Property
values
name : String
values : Object [0..*]
unit : String
actual system in operation
«block» Design Model
«block» Prototype
«block» System
«block» Discrete Property
«block» Stochastic Property
computer model / digital mockup of the system
«block» Prototype or System
«block» Continuous Property
«block» System or Lifecycle Process
prototype / mock-up of the actual system
Property can only take values from a discrete set
Property can only take values from a continuous set
Property values are not deterministic but uncertain and quantified using probability distribution functions.
«block» System Lifecycle Process
design
operations
retirement
«block» Design Process
«block» Operation Process
«block» Disposal Process
manufacturing
maintenance
«block» Manufacturing Process
«block» Maintenance Process

Figure 3: System representation (SysML BDD)

Property concept represents attributes of a system, in any of its representations. Every property has a name, set of values, and a unit if the property represents a quantity such as mass. A property may be discrete or continuous—take values from a discrete or continuous set of values. A property may have deterministic or stochastic values (expressed with probabilities). A system representation may have one or more properties, as shown in Figure 3. For a given analysis, some or all of these properties may be of interest, as shown in Figure 2.

Analysis Model concept represents the computation model used to calculate the system properties (relevant to the analysis) to meet the analysis objectives. Analysis model could be computer-based executable model (e.g. Mathematica/MATLAB code or FEA/CFD model), or a model representing physical measurement on a prototype or actual system. For every analysis model, the following characteristics are modeled:

(1) Language in which the model is formulated
(2) Software used to formulate the model
(3) Type of model
(4) Result data from executing the model
(5) Relationship to the system representation, e.g. design model. This relationship embodies the model transformations required to generate or update the analysis model from the system representation

Analysis Relation concept represents the relationships between analyses. A given analysis may be related to multiple analyses, such as in the following scenarios:

(1) An analyst may perform the same type of analysis (same objective) with varying degrees of fidelity, such as one-, two-, or three-dimensional analyses. For each analysis, a new analysis and corresponding analysis model would be created, and all analyses of the same type can also be grouped together using the analysis relation concept.

(2) Based on the results of an analysis (say A1), new analyses (say A2, A3, and A4) may be formulated for the system. In this scenario, the analysis relationship is used to relate A2, A3, and A4 as being derived from A1.

(3) A single analysis may be decomposed into multiple analyses. The analysis relationship concept can also be used to represent this decomposition.

Analysis Result concept represents the result of the analysis in terms of the evaluations of all the expressions in the **Analysis Objective**. An analysis is successful if its objective has been met. Since the objective is represented as a set of expressions, an analysis is successful if all the expressions can be evaluated. The analysis should generate enough information to be able to evaluate the expressions in the analysis objective. For example, if the expressions associated with the analysis objective were (1) mass < 1000 kg, and (2) power generated > 1200 hp, then the analysis result would contain the evaluations of these expressions: (1) true, and (2) false.

Decision concept represents the decisions that are taken related to the analysis. This includes downstream decisions—based on the result of the analysis, and the upstream decisions that led to performing the analysis. Modeling the decision is not in the scope of this project. We will leverage the OMG Decision Modeling Notation [DMN 1.0, 2015] for this purpose. It is important to capture the decisions taken during a system engineering process such that they can be traced downstream, especially for design reviews and problem resolutions. It is also important to capture the rationale and the follow-up actions for each decision. This need is addressed by the relationships between the **System Analysis** concept and the **Decision** concept—*resulted from* for upstream decisions, and *resulting in* for downstream decisions.

Generic to SysML 2.0 – to be merged above

1 Universal UUID
2 Versioning of elements and configuration control
3 Provide a way to specify model transformations to generate analysis model from system representation – equation-based model, state-based model, flow-based model
4 Representation of datatypes
    a. Arrays, Lists (ordered/unordered), Sets, …
    b. Matrices (mxn)
    c. Map (key-value pairs)
    d. Tensors and Vectors
    e. Mutable and Immutable objects (constants)
    f. Date and Time
    g. Temporal model for representing time-based representation of properties, behavior, and structure.
    h. Geographic map
    i. Probability Distributions
5 Units
    a. QUDV – no representation of FPS system
    b. Automated unit conversions

6 Operators and Functions
    a. Differential
    b. Integral
    c. …
7 Libraries of functions
8 Built in solvers
9 Geometry
    a. Coordinate systems
    b. Primitive geometric shapes
        i. Primitive shape definitions in SysML
        ii. Shape as a characteristic property of a hardware element typed by an external AP242 model.

4. Requirements

These are the requirements for analysis that SysML 2.0 should satisfy.

1. Explicit representation of analysis concept

2. Concepts for modeling quantitative analyses

3. Concepts for modeling qualitative analyses

4. Concepts for managing analysis inputs and context

5. Concepts for representing MoE, KPP, and key things being targeted for computation during analyses

6. Concepts for representing and executing analysis workflow

7. Concepts for managing, archiving, querying, and visualizing analysis results

8. Concepts for representing decisions and linking them to analyses

9. Concepts for representing and executing model transformations, generating analysis models from design/system description, and vice versa

10. Support for analysis models in multiple formalisms – equation-based, state-based, function flow-based, graph-based, and more.

11. Improved representation and libraries for units (e.g. QUDV++)

12. Libraries of higher-level data structures and units, such as matrices, arrays (ordered, unordered, list, set), tensors, immutable objects, date, time, geographic map, time, probability distributions.

13. Concepts for representing and executing transformations between units and data types

14. Concepts to represent a rich set of mathematical functions and algebra, e.g. calculus (differentials, integrals), trigonometric functions, coordinate geometry functions, and more.

15. Concepts for representing primitive geometry for analysis requirements, formulation, and results (alignment with ISO STEP Part 42 and AP 242).

16. Concepts for representing material properties (homogeneous and heterogeneous)

17. Concepts for representing co-ordinate system and reference frames

18. Common concepts
    - 18.1. Representation of unique ID for all concepts (UUID)
    - 18.2. Versioning of concepts
    - 18.3. Configuration management of concepts (organization, user/role, read/write privileges, etc.)
    - 18.4.