LIFECYCLE MODELING LANGUAGE (LML) SPECIFICATION

Document URL: http://www.lifecyclemodeling.org/spec/1.0
Current Document URL: http://www.lifecyclemodeling.org/spec/current

October 17, 2013

Table of Contents

1 Specification Information

With the growth of the Internet and daily changes in Information Technology, systems have become more complex and change more rapidly than ever before. When coupled with ever tightening budgets and schedules it makes for a very challenging future. This is the challenge handed over to the modern Systems Engineer to resolve. Development of these systems requires:

- plans that are nimble and responsive to change,
- designs that are easy to understand by all stakeholders,
- architectures that are easy to modify,
- methodologies that are appropriate to the problem at hand,
- processes that support all stages of the system lifecycle.

Systems engineering approaches, methods and tools have evolved but have not kept pace with the rate of change of modern systems. Model Based Systems Engineering, SysML, DODAF, MODAF, and UAF are tremendous steps forward but do not address the entirety of the problem. Large portions of the development lifecycle are ignored by these approaches. A new approach to analyzing, planning, specifying, designing, building and maintaining modern systems is needed. The Lifecycle Modeling Language (LML) is that approach.

LML was designed with 6 major goals.

1. To be easy to understand,
2. To be easy to extend,
3. To support both functional and object oriented approaches within the same design,
4. To be a language that can be understood by most system stakeholders, not just Systems Engineers,
5. To support systems from cradle to grave,
6. To support both evolutionary and revolutionary changes to system plans and designs over the lifetime of a system.

Most of the systems engineering community recognizes MBSE's ability to evolve, reuse and execute models is a significant improvement over the classic "document-based" approach's static view of a system. These capabilities help to ensure a design of high quality. Good models can bridge the gap between written requirements and bending metal or writing code, the thing that is desired versus the thing that is delivered.

LML takes the principles of MBSE beyond the system development and production stages into the conceptual, utilization, support and retirement stages by providing a robust easy to understand ontology that allows one to model the complex interrelationships not only between system components but between those components and programmatic artifacts such as schedules and risk management plans using clear diagrams to express system information. LML was designed to integrate all lifecycle disciplines, including program management, systems and design engineering, as well as test, deployment and maintenance into a single framework. As a result, LML is a language that can be used throughout the lifecycle. LML uses common, everyday language to define its modeling elements such as entity, attribute, schedule, cost, relationship. It's primary modeling constructs are the box which represents whichever part of the system that is necessary, and the directed arrow which depicts a relationship between modeled elements such as "consists of," "derived from,' "costs." This simplicity allows everyone from the least technical stakeholder to highly skilled end users to model and understand systems of arbitrary complexity. LML becomes the Rosetta Stone that allows easy communications between disparate disciplines across multiple industries.

1.1 Purpose of this Specification

The LML specification's purpose is to provide a reference for users of the language to understand its goals, concepts and structure and to provide vendors a reference for implementation of the language.

1.2 LML Steering Committee

The direction and evolution of this standard is overseen by the LML Steering Committee. It consists of expert systems engineers and program managers from industry and academia. Their charter is to ensure LML evolves in such a way that it continues to meet the needs of its user community. You may provide comments and suggestions on the LML website (www.lifecyclemodeling.org).

1.3 Documentation Conventions & Terminology

When referencing an entity, outside of a heading, the entity name is always in bold with the first letter capitalized, as in **Action**.

When referencing an attribute, outside of a heading, the attribute name is always in italics with the first letter lower case, as in *control*.

When clarification is needed to better identify an attribute, the entity name should be appended to the end of the attribute name as follows: "(**Entity Name**)"; where "**Entity Name**" would be the name of the entity. For example: *units*(**Characteristic**) and *units*(**Cost**), which clarifies the difference between the *units* attribute of the **Characteristic** entity and the *units* attribute of the **Cost** entity.

When referencing a relationship, outside of a heading, the relationship name is always in bold italics with first letter lower case, as in ***traced to***.

When referencing an attribute on a relationship, outside of a heading, the attribute name is always underlined with the first letter lower case, as in trigger.

When referencing the Lifecycle Modeling Language (LML) it is referenced via the full name, Lifecycle Modeling Language, or with the abbreviation, LML.

2 Overview of Lifecycle Modeling Language (LML)

The basis for the LML formulation is the classic entity, relationship, and attribute (ERA) meta-meta model. This formulation modifies the classical approach slightly by including attributes on relationships, to provide the adverb, as well as the noun (entity), relationship (verb), and attribute (adjective) language elements. Since LML was designed to translate to object languages, such as UML/SysML, these language elements correspond to classes (entity), relations (relationship), and properties (attribute).

Current Systems Engineering languages tend to add complexity to already complex problems, thus making it more difficult to communicate the underlying issues and develop effective solutions. LML is designed to be a simpler language, both in terms of its ontology and visual expressions. This feature makes it easy to understand by the entire set of lifecycle stakeholders. Such a simplified language many not include all the "bins" for information a particular domain wants, hence, LML is extensible allowing for discipline and domain specific extensions to support the needs of a specific project, organization or customer. The process for extension submission will be detailed on the lifecyclemodeling.org website.

For example, UML and SysML use the term Actor to define a part of the system that performs actions. The DoDAF Metamodel 2 uses the word Performer for the same purpose. LML uses the term, **Asset**, but also allows the user to extend the language by defining Actors or Performers. However, we recommend modelers use the *type* attribute for **Assets** instead, as a means to differentiate between these different names. New entities or child entities are recommended only when new attributes and/or relationships are needed. Thus modelers using LML will know what entity to put something in immediately and can adjust the *type* as needed. We saw this problem in the DoDAF where modelers were often confused by the difference between an operational node and a systems node. When you have two "bins" to put the same kind of information into confusion often results. Further examples of this comparison appear in Appendix A and B.

This specification of LML also defines common visualizations for information. For example, the *Risk* entity has a standard Risk Matrix as its basic diagram. Each entity has these kinds of common visualizations and they need to be as simple as possible to reduce the complexity of the language and make it more understandable by a broader set of stakeholders. Other visualizations are allowed and encouraged as they aid in expressing the information, which is the real goal of any language visualizations. These can and should be proposed as extensions to the language as well so that others practitioners can benefit from these visualizations.

Ontologies provide a set of defined terms and relationships between the terms to capture the information that describes the physical, functional, performance, and programmatic aspects of the system. By system,[1] we mean the entire set of processes, people and things which operate for the benefit of people. Common ways for describing such ontologies is entity, relationship, and attribute (ERA). ERA is often used to define database schemas. LML uses the ERA approach, but extends it by adding attributes to relationships. The extension reduces the number of relationships needed, just as attributes reduce the number of entities needed.

This section defines the ERAs for LML, thus providing the basic definitions of the data types used to collect information about the system. In addition, we describe how inheritance, extensions, limitations and instantiation can be used by tool developers to remain within the guidelines of this standard.

[1] INCOSE defines a system as a *combination of interacting elements organized to achieve one or more stated purposes.*

Entity, relationship and attribute have equivalent English language elements: noun, verb, and adjective. With the addition of attributes on the relationship, we also have the equivalent of the adverb. These equivalencies have been provided to aid in understanding the semantics of the language.

2.1 Entity (noun)

An entity is something can exist by itself and is uniquely identifiable. In LML, we have defined 12 parent entities (**Action**, **Artifact**, **Asset**, **Characteristic**, **Connection**, **Cost**, **Decision**, **Input/Output**, **Location**, **Risk**, **Statement** and **Time**). The rationale for this set is presented in Section 3. Several child entities have also been defined as they have specific utility in capturing the information needed by the system lifecycle stakeholders. The child entities are **Conduit** (child of **Connection**), **Logical** (child of **Connection**), **Measure** (child of **Characteristic**), **Orbital** (child of **Location**), **Physical** (child of **Location**), **Requirement** (child of **Statement**), **Resource** (child of **Asset**), and **Virtual** (child of **Location**). These child classes have the attributes and relationships of the parents plus additional attributes and relationships that make them unique. More on this "inheritance" of attributes and relationships is discussed in section 2.6. These twenty (20) entities shall be part of any LML-compliant language extension.

Every entity shall have a *name* or *number* or *description* attribute or combination of the three to identify it uniquely. The *name* is a word or small collection of words providing an overview of information about the entity. The *number* provides a numerical way to identify the entity. The *description* provides more detail about that entity.

In terms of the English language, an entity is like a noun.

2.2 Attribute (adjective)

An attribute is an inherent characteristic or quality of an entity. It further describes the entity, enhancing its uniqueness. Every attribute shall have a *name* to identify it uniquely within an entity. The *name* is a word or small collection of words providing an overview of information about the attribute. The attribute data type (see 2.5 below) specifies the data associated with the attribute.

Attributes names shall be name unique with in an entity, but may be used in other entities, such as the example provided earlier: *units*(**Characteristic**) and *units*(**Cost**) to avoid confusion within an entity specification.

In terms of the English language, an attribute is like an adjective.

2.3 Relationship (verb)

A relationship connects entities to each other. In LML, all relations shall be defined in both directions and shall have unique names with the same verb. For example, the standard parent child relationship (used by all LML entities) is ***decomposed by*** and its inverse is ***decomposes***. Each relationship and its inverse shall have unique names. Relationship names also shall be unique across the whole schema. The relationships enable an English reading of the way entities connect. For example, when connecting an **Action** to a **Statement,** LML uses ***traced from*** as the relationship: an **Action** is ***traced from*** a **Statement**. The inverse relation of ***traced from*** is ***traced to***, and thus would be read as: a **Statement** is ***traced to*** an **Action**. Figure 2-1 shows this and the Entity-Relationship Diagram (ERD) for this relationship.

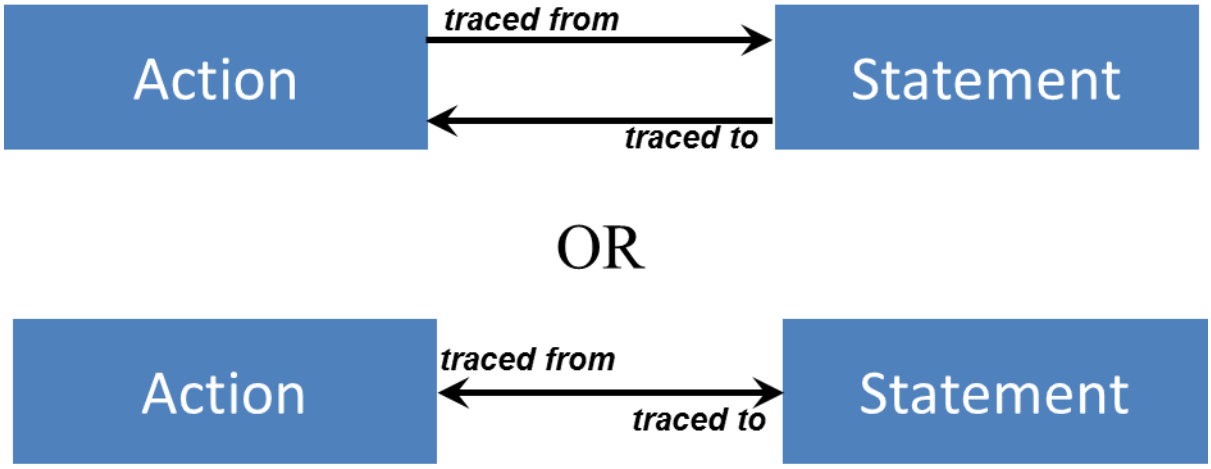

Figure 2-1. Entity Relationship Diagram for the Relationship Between an Action and a Statement.

For relationships within the same entity, such as decomposed by and decomposes can be visualized as an ERD as well (see Figure 2-2).

Figure 2-2. Entity Relationship Diagram for the Relationship Between an Action and Itself.

In terms of the English language, a relationship is like a verb.

2.4 Attributes on Relationships (adverb)

The classic ERA modeling does not include attributes on relationships. However, for LML this addition is very useful.

Figure 2-3 shows an example of how the attribute on a relationship is depicted in an extended version of the ERD. The attribute on a relationship is shown with a dashed line to the relationship. The attribute on a relationship shall have a unique name for that relationship, but can be used in other relationships, if necessary to enhance communications.

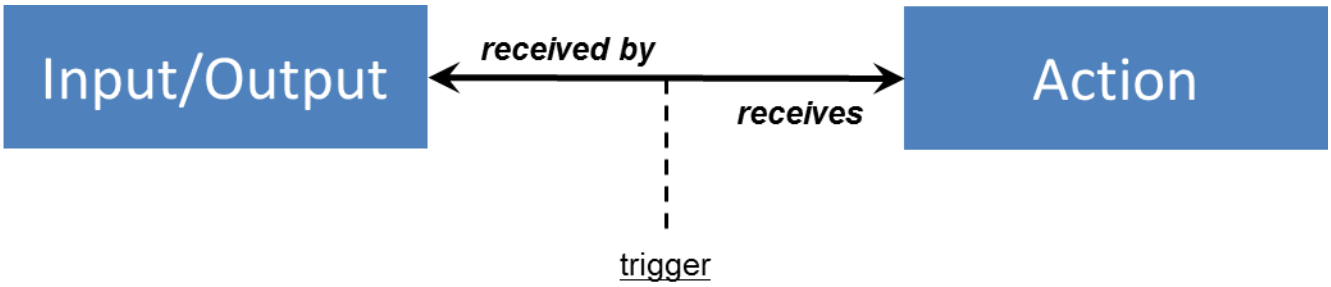

Figure 2-3. Extended Entity Relationship Diagram for the Attribute on the Relationship Between an Action and an Input/Output.

In terms of the English language, an attribute on a relationship is like an adverb.

2.5 Attribute Data Types

For a complete specification, it is essential to define the data types appropriate for the attributes, as they can vary significantly and without this specification interoperability with other schemas (i.e., translations) would be very difficult. The following sub-sections present the current acceptable set of attribute data types for LML. Extensions (see Section 2.7) may also extend this list of data types as well.

2.5.1 Text

A Text data type represents: a single character, single word, or multiple words.

2.5.2 Number

A Number data type represents any individual real number.

2.5.3 Boolean

A Boolean data type represents the two possible logical values “true” and “false”.

2.5.4 Percent

A Percent data type represents a special case of Number where the value is restricted to values between zero and one hundred, inclusive.

2.5.5 DateTime

A DateTime data type represents a specific method of storing a given date and time value. Commonly stored as “YYYY-MM-dd hh:mm:ss”. Where “YYYY” is the four digit year, “MM” is the two digit month, “dd” is the two digit day, “hh” is the two digit hour (in twenty-four hours), “mm” is the two digit minute, and “ss” is the two digit second.

2.5.6 URI

An URI date type represents a special case of Text where the value must be a Uniform Resource Identifier. Examples of URI: “C:/Program Files”, “http://www.google.com”, and “test@test.com”.

2.5.7 Enumeration

An Enumeration data type represents a set of defined choices, where the selection of only one choice is permitted. An enumeration must contain at minimum two choices. All choices within the set must be of the same data type (having a set containing Text data type and Number data type is not permitted). Also, within a set a repeated choice is not permitted (A set cannot have repeated “Apple” as in: “Apple”, “Orange”, “Apple”, and “Plum”).

2.5.8 GeoPoint

A GeoPoint data type represents a longitude and latitude pair on the surface of a body.

2.6 Inheritance

With LML it is possible to create child entities. The child entity inherits all the attributes and relationships from the parent entity with the EXCEPTION of the attribute *type*; this attribute is overridden by the child entity.

An example of a child entity is **Resource** with parent entity **Asset**.

A child entity will also inherit relationships from the parent. Additional relationships may also be defined for the child.

2.7 Extensions

Extensions for domain-specific needs are allowed and encouraged when necessary. However, it is strongly recommended however to avoid simply aliasing the current ontology, but instead to identify aliases as another “*type*” of the entity, which is a common attribute for every entity. Tool developers may want to display the entity *type*, rather than the entity *name* to enhance the communication with these other languages.

The schema user can also apply **Characteristic** entity in many cases to provide ‘attributes” of an **Asset** or **Action**, thus making many other extensions unnecessary. It is recommended to explore this option first, before creating an extension.

All extension must be submitted to the LML Steering Committee for adjudication before they will be recognized as official extensions to LML.

2.8 Instantiation

Actual instantiation of the LML specification will be up to tool vendors. If they find portions of the specification difficult to implement, they can apply to the LML Steering Committee for guidance.

3 LML Ontology

Systems engineers, enterprise architects and program managers have overlapping needs for information and LML provides a basic but comprehensive set of design elements that satisfy them all. For example, the systems engineer is concerned with optimizing cost, schedule and performance. Performance includes form, fit, and function. Enterprise architects often use the "5WH" model (What, Why, When, Where, Who, and How) to capture their information. The Program Manager is primarily concerned with cost, schedule, tasks, resources, and risks. Table 3-1 shows these various information needs and how LML satisfies them. Each of these entities has unique and common attributes and relationships.

Table 3-1. Comparison Between LML and the Information Needs of Key System Lifecycle Stakeholders

Systems Engineering	Architecture	Program Management	Lifecycle Modeling Language
Cost	(How Much)	Cost	Cost
Schedule	When	Schedule	Time/Action
Performance			
Form	Who	Organization	Asset
	What	Resource	Resource
	Where	Location	Location
	Why	Goal, Objective & Decision	Decision & Statement/Requirement
Function	How	Task	Action
Metric (Fit)		Metric	Characteristic/Measure
Interface			Connection (Conduit) & Input/Output
Risk		Risk	Risk
		Artifact	Artifact

This section provides an overview of LML and then subsequently delves into the detailed attributes, relationships, and attributes on relationships for each entity. This information is fundamental for tool developers planning to instantiate LML.

3.1 LML Entities

Table 3-2 summarizes the LML entities, their parent entity, description and examples (where appropriate) of how they might be used. Note that these examples can be part of the *type* attributes used as aliases for the entity itself.

Table 3-2. Summary of the LML Entities

Entity Name	Parent Entity	Description	Examples
Action	None	An **Action** entity specifies the mechanism by which inputs are transformed into outputs.	Activity, Capability, Event, Function, Process, Task
Artifact	None	An **Artifact** entity specifies a document or other source of information that is referenced by or generated in the knowledgebase.	Document, E-mail, Procedure, Specification
Asset	None	An **Asset** entity specifies an object, person, or organization that performs Actions, such as a system, subsystem, component, or element.	Component, Entity, Service, Sub-system, System
Characteristic	None	A **Characteristic** entity specifies properties of an entity.	Attribute, Category, Power, Role, Size, Weight
Conduit	**Connection**	A **Conduit** entity specifies the means for physically transporting **Input/Output** entities between **Asset** entities. It has limitations (attributes) of capability and latency.	Data Bus, Interface, Pipe
Connection	None	A **Connection** entity specifies the means for relating **Asset** instances to each other.	Abstract entity
Cost	None	A **Cost** entity specifies the outlay or expenditure (as of effort or sacrifice) made to achieve an objective associated with another entity.	Earned Value, Work Breakdown Structure, Actual Cost, Planned Cost
Decision	None	A **Decision** entity specifies a challenge and its resolution.	Major Decision, Challenge, Issue, Problem
Input/Output	None	An **Input/Output** entity specifies the information, data, or object input to, trigger, or output from an **Action**.	Item, Trigger, Information, Data, Energy
Location	None	A **Location** entity specifies where an entity resides.	Abstract entity
Logical	**Connection**	A **Logical** entity represents the abstraction of the relationship between two entities (e.g., **Asset** entities with the type “Entity”)	Has, “is a”, “relates to”
Measure	**Characteristic**	A **Measure** entity specifies properties of measurements and measuring methodologies, including metrics.	Key Performance Parameter (KPP), Measure of Effectiveness (MOE), Measure of Performance (MOP), Metric

Entity Name	Parent Entity	Description	Examples
Orbital	**Location**	An **Orbital** entity specifies a location along an orbit around a celestial body.	Orbit
Physical	**Location**	A **Physical** entity specifies a location on, above, or below the surface.	Map Coordinates
Requirement	**Statement**	A **Requirement** entity identifies a capability, characteristic, or quality factor of a system that must exist for the system to have value and utility to the user.	Functional Requirement, Performance Requirement, Safety Requirement
Resource	**Asset**	A **Resource** entity specifies a consumable or producible **Asset**.	Fuel, Bullets, Missiles, People
Risk	None	A **Risk** entity specifies the combined probability and consequence in achieving objectives.	Cost Risk, Schedule Risk, Technical Risk
Statement	None	A **Statement** entity specifies text referenced by the knowledgebase and usually contained in an **Artifact**.	Need, Goal, Objective, Assumption
Time	None	A **Time** entity specifies a point or period when something occurs or during which an action, asset, process, or condition exists or continues.	Milestone, Phase
Virtual	**Location**	A **Virtual** entity specifies a location within a digital network.	URL

3.2 LML Relationships

The relationships between the entities are provided in the Table 3-3. Note that the same verb is used for the inverse relationships.

* - Implies the inverse relation is present, just not shown.

Table 3-3. Summary Table of LML Relationships

	Action	Artifact	Asset (Resource)	Characteristic (Measure)	Connection (Conduit, Logical)	Cost	Decision	Input/Output	Location (Orbital, Physical, Virtual)	Risk	Statement (Requirement)	Time
Action	decomposed by* related to*	references	(consumes) performed by (produces) (seizes)	specified by	-	incurs	enables results in	generates receives	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Artifact	referenced by	decomposed by* related to*	referenced by	referenced by specified by	defines protocol for referenced by	incurs referenced by	enables referenced by results in	referenced by	located at	causes mitigates referenced by resolves	referenced by (satisfies) source of traced from (verifies)	occurs
Asset (Resource)	(consumed by) performs (produced by) (seized by)	references	decomposed by* orbited by* related to*	specified by	connected by	incurs	enables made responds to results in	-	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Characteristic (Measure)	specifies	references specifies	specifies	decomposed by* related to* specified by*	specifies	incurs specifies	enables results in specifies	specifies	located at specifies	causes mitigates resolves specifies	(satisfies) spacifies traced from (verifies)	occurs specifies
Connection (Conduit, Logical)	-	defined protocol by references	connects to	specified by	decomposed by* joined by* related to*	incurs	enables results in	transfers	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Cost	incurred by	incurred by references	incurred by	incurred by specified by	incurred by	decomposed by* related to*	enables incurred by results in	incurred by	located at	causes incurred by mitigates resolves	incurred by (satisfies) traced from (verifies)	occurs
Decision	enabled by result of	enabled by references result of	enabled by made by responded by result of	enabled by result of specified by	enabled by result of	enabled by incurs result of	decomposed by* related to*	enabled by result of	located at	causes enabled by mitigated by result of resolves	alternative enabled by traced from result of	date resolved by decision due occurs
Input/Output	generated by received by	references	-	specified by	transferred by	incurs	enables results in	decomposed by* related to*	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Location (Orbital, Physical, Logical)	locates	locates	locates	locates specified by	locates	locates	locates	locates	decomposed by* related to*	locates mitigates	locates (satisfies) traced from (verifies)	occurs
Risk	caused by mitigated by resolved by	caused by mitigated by references resolved by	caused by mitigated by resolved by	caused by mitigated by resolved by specified by	caused by mitigated by resolved by	caused by incurs mitigated by resolved by	caused by enables mitigated by results in resolved by	caused by mitigated by resolved by	located at mitigated by	caused by* decomposed by* related to* resolved by*	caused by mitigated by resolved by	occurs mitigated by
Statement (Requirement)	(satisfied by) traced to (verified by)	references (satisified by) sourced by traced to (verified by)	(satisified by) traced to (verified by)	(satisified by) specified by traced to (verified by)	(satisified by) traced to (verified by)	incurs (satisified by) traced to (verified by)	alternative of enables traced to results in	(satisified by) traced to (verified by)	located at (satisfied by) traced to (verified by)	causes mitigates resolves	decomposed by* traced to* related to*	occurs (satisified by) (verified by)
Time	occurred by	occurred by	occurred by	occurred by specified by	occurred by	occurred by	date resolves decided by occurred by	occurred by	occurred by	occurred by mitigates	occurred by (satisfies) (verifies)	decomposed by* related to*

3.3 Traceability

Since LML was designed to support traceability of requirements to their implementation mechanisms (Asset class entities), the primary path for traceability is in Figure 3-1. This diagram does not reflect all the relationships shown in Table 3-3.

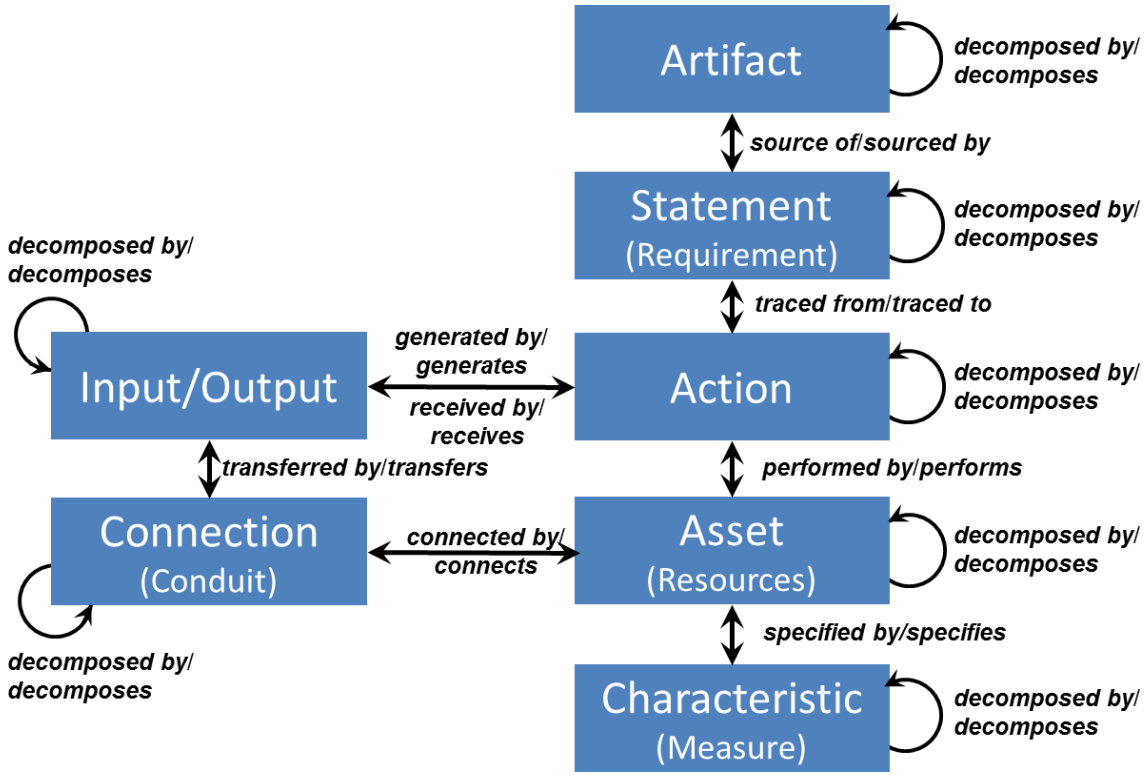

Figure 3-1. Principal Entities and Relationships for Design LML Traceability

3.4 Entity Specifications

The following tables and subsections provide the detailed entity, relationship and attribute specifications for the LML ontology. These specifications shall be used by language users to provide the basis for all LML-related extensions. **Inverse** provides the name of the relationship from the perspective of the target entity. **Target entity(ies)** provides the name of the entity(ies) on the other side of the relationship.

These attributes are common to all entities.

Attribute	Data Type	Description
name	Text	*name* designates the particular instance of an entity called element.
number	Text	*number* provides the element's place in a hierarchy.
description	Text	*description* captures the text needed to describe this element.

3.4.1 Action

Entity	Parent Entity	Child Entities	Description
Action	None	None	An **Action** entity generates effects and may have pre-conditions before it can be executed. This **Action** can include transforming inputs into outputs. Examples: Process, Discover, Calculate.

Attribute	Data Type	Description
duration	Number	*duration* represents the period of time this **Action** occurs.
percent complete	Percent	*percent complete* represents the percentage this **Action** is complete.
start	DateTime	*start* represents the time when this **Action** begins.
type	Text	*type* provides aliases for the entities. For **Action** these can include: Activity, Capability, Event, Function, Mission, Operational Activity, Program, Service Orchestration, Simulation Workflow, Subprocess, System Function, Task, Training, Use Case, Work Process, Workflow

Relationship	Inverse	Target Entity	Description
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
consumes	***consumed by***	**Resource**	***consumes*** identifies the **Resource** that this **Action** uses. After this **Action** is completed the amount consumed is not returned to the **Resource**.
	Attribute	Data Type	Description
	amount	Number	amount represents how much of the resource is ***consumed by*** the **Action**. Units are relative to the units selected for the **Resource**.
decomposed by	***decomposes***	***Action***	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	***Action***	***decomposes*** identifies the parent of this entity.
enables	***enabled by***	***Decision***	***enables*** identifies the *Decision* that is empowered by this entity.
generates	***generated by***	***Input/Output***	***generates*** identifies the **Input/Output** that this **Action** transforms.
incurs	***incurred by***	***Cost***	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	***Location***	***located at*** identifies the **Location** where this entity exists.

Relationship	Inverse	Target Entity	Description
mitigates	***mitigated by***	***Risk***	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	***Time***	***occurs*** identifies the **Time** (or timespan) this entity happens.
performed by	***performs***	**Asset**	***performed by*** identifies the **Asset** that executes this **Action**.
produces	***produced by***	**Resource**	***produces*** identifies the **Resource** that is created by this **Action**. **Resources** are produced when the execution of the **Action** completes.
	Attribute	Data Type	Description
	amount	Number	amount represents how much of the **Resource** is ***produced by*** the **Action**. Units are relative to the units selected for the **Resource**.
receives	***received by***	**Input/Output**	***receives*** identifies the **Input/Output** that is taken in by this **Action**.
	Attribute	Data Type	Description
	trigger	Boolean	trigger represents this relation as an enabling requirement for the **Action**. An **Action** begins execution when it has received control enablement, all of its triggers have arrived, and its necessary resources are available.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Action**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Action**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.

Relationship	Inverse	Target Entity	Description
seizes	***seized by***	**Resource**	***seizes*** identifies the **Resource** that this **Action** uses. After this **Action** has completed the **Resource** is released for use by other **Actions**.
	Attribute	Type	Description
	amount	Number	amount represents how much of the resource is ***captured by*** the **Action**. Units are relative to the units selected for the **Resource**.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.

3.4.2 Artifact

Entity	Parent Class	Child Entities	Description
Artifact	None	None	An **Artifact** entity specifies a document or other source of information that is referenced by or generated in the knowledgebase. Example: Requirements Report.

Attribute	Type	Description
date published	DateTime	*date published* represents the date when this **Artifact** was accessed (webpage), published, or uploaded to the knowledgebase.
file	URI	The *file* that represents this **Artifact**.
type	Text	*type* provides aliases for the entities. For **Artifact** these can include: Briefing, Change Notice, Change Request, Concept of Operations, Directive, Doctrine, Document, E-Mail, Guidance, Instruction, Interface Control Document, Interface Requirements Specification, Manual, Matrix, Meeting Minutes, Memorandum, Mitigation Plan, Model, Operational Concept, Policy, Procedure, Protocol, Proposal, Regulation, Requirements Document, Request for Proposals, Software Requirements Specification, Standard, System Requirements Document, System/Segment Design Document, System/Segment Specification, Test & Evaluation Plan, Test & Evaluation Report, Text Message, Trade Study, White Paper

Relationship	Inverse	Target Class(es)	Description
causes	***caused by***	**Risk**	***causes*** identifies the *Risk* resulting from this entity.
decomposed by	***decomposes***	**Artifact**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Artifact**	***decomposes*** identifies the parent of this entity.
defines protocol for	***defined protocol by***	**Conduit**	***defines protocol for*** identifies the **Conduit** that uses the standard identified in this **Artifact**.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that is empowered by this entity.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.

occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
referenced by	***references***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Decision, Input/Output, Risk, Statement**	***referenced by*** identifies the entity that specified and/or enhanced its definition by this **Artifact**.
related to	***relates***	**Artifact**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Artifact**	*relates* identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
source of	***sourced by***	**Statement**	***source of*** identifies the **Statement** that is contained in this **Artifact**.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.

3.4.3 Asset

Entity	Parent Entity	Child Entities	Description
Asset	None	**Resource**	An *Asset* entity specifies an object, person, or organization that used to create value and perform **Actions**, such as a system, subsystem, component, or element. Example: Infrared Sensor, Accounting Department, Internal Revenue Service

Attribute	Type	Description
type	Text	*type* provides aliases for the entities. For **Asset** these can include: Architecture, Assembly, Component, Context, CSC, CSCI, CSU, Element, Environment, External System, Facility, Hardware, Human, HW Element, HWCI, Infrastructure, LRU, Materiale, Operational Element, Organization, Part, Performer, Personnel, Segment, Service, Software, Subassembly, Subsystem, System, System Instantiation, Test Equipment, Test Software, Unit

Relationship	Inverse	Target Entity(es)	Description
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
connected by	***connects to***	**Connection**	***connected by*** identifies the **Connection** that adjoins this **Asset**.
	Attribute	Type	Description
	origin	Boolean	origin represents if the **Asset** is the origin of the **Connection**. This attribute enables unidirectional (one or the other **Asset**'s relationship origin is false), in additional to the default bi-directional flow (both origins are set to true as a default).
	multiplicity	Text	multiplicity, also called cardinality, represents the number of relationships (one to many, many to one, etc.) that can occur.
decomposed by	***decomposes***	**Asset**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Asset**	***decomposes*** identifies the parent of this entity.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that is empowered by this entity.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.

made	***made by***	**Decision**	***made*** identifies the **Decision** that this **Asset** decided.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
orbited by	***orbits***	**Asset**	***orbited by*** identifies **Asset** entity that acts as a satellite to this **Asset**.
orbits	***orbited by***	**Asset**	***orbits*** identifies the **Asset** that this **Asset** moves around.
performs	***performed by***	**Action**	***performs*** identifies the **Action** that is executed by this *Action*.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Asset**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Asset**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
responds to	***responded by***	**Decision**	***responds to*** identifies the **Decision** that is acted on by this **Asset** (usually a person or organization).
	Attribute	Type	Description
	responsibility	Enumeration [**Primary**, Secondary]	responsibility represents the **Asset** that has the responsibility for resolving the **Decision**.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.

3.4.4 Characteristic

Entity	Parent Entity	Child Entities	Description
Characteristic	None	**Measure**	A **Characteristic** entity specifies properties of an entity. Examples: Blue, no heavier than 2 oz, accurate to within 1%

Attribute	Type	Description
units	Text	*units* represents this **Characteristic**'s units used to measure the value, such as ponds or miles per hour.
value	Text	*value* represents this **Characteristic**'s current value.
type	Text	*type* provides aliases for the entities. For **Characteristic** these can include: Condition, Data Element, Environmental, Functional, Performance, Physical, Scenario, Security, Verification Category

Relationship	Inverse	Target Entity(es)	Description
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
decomposed by	***decomposes***	**Characteristic**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Characteristic**	***decomposes*** identifies the parent of this entity.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that is empowered by this entity.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Characteristic**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.

relates	***related to***	**Characteristic**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	*context* represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
specified by	***specifies***	**Characteristic**	**specified by** identifies a **Characteristic** that provides further information about this entity.
specifies	***specified by***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Decision, Input/Output, Location, Risk, Statement, Time**	***specifies*** identifies an entity that this **Characteristic** provides further information about.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.

3.4.5 Conduit (Connection)

Entity	Parent Entity	Child Entities	Description
Conduit	**Connection**	None	A **Conduit** entity specifies the connection between Assets and has capacity and latency, which carries an Input/Output. Examples: Spacewire, Bluetooth, railway

Attribute	Type	Description
capacity	Number	*capacity* represents the maximum rate supported by the **Conduit**. (Used in simulation as to add time delay by dividing the **Input/Output** size with the *capacity* value.)
latency	Number	*latency* represents the time required to transmit the information or **Input/Output** entity over this **Conduit**. This value does not factor in any delays due to capacity limitations.
units	Text	*units* represents this **Conduit**'s units used to measure the capacity, such as bits per second or gallons per minute.
type	Text	*type* provides aliases for the entities. For **Conduit** these can include: Cable, Downlink, Human-in-the-Loop, Human Machine Interface, Interface, Landline, Link, Needline, Network, Pipe, RF - Satcom, RF - Terrestrial, Roadway, Service Interface, Uplink, Wireless.

Relationship	Inverse	Target Entity(es)	Description
decomposed by	***decomposes***	**Conduit**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Conduit**	***decomposes*** identifies the parent of this entity.
defined protocol by	***defines protocol for***	**Artifact**	***defined protocol by*** identifies the **Artifact** that contains the standard used by this **Conduit**.
transfers	***transferred by***	**Input/Output**	***transfers*** identifies the **Input/Output** that is passed by this **Conduit**.

All attributes and relationships from **Connection** are inherited by the **Conduit** entities.

3.4.6 Connection

Entity	Parent Entity	Child Entities	Description
Connection	None	**Conduit**, **Logical**	A **Connection** entity specifies the mechanism relating Assets. This is an abstract class.

Relationship	Inverse	Target Entity(es)	Description
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
connects to	***connected by***	**Asset**	***connects to*** identifies the **Asset** that this *Connection* adjoins.
	Attribute	Type	Description
	origin	Boolean	origin represents if the **Asset** is the origin of the *Connection*. This attribute enables unidirectional (one or the other **Asset***'s* relationship origin is False), in additional to the default bi-directional flow (both origins are set to True as a default).
	multiplicity	Text	multiplicity, also called cardinality, represents the number of relationships (one to many, many to one, etc.) that can occur.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that is empowered by this entity.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
joined by	***joins***	**Connection**	***joined by*** identifies the **Connection** that connects to this **Connection**. ***joined by*** implies the end of the relationship.
	Attribute	Type	Description
	bidirectional	Boolean	bidirectional represents if the connection between two **Connections** is bidirectional. If the **Connection** is not bidirectional the *Connection* that has the relation ***joins*** is the start and the Connection with ***joined by*** is the end.

joins	***joined by***	**Connection**	***joins*** identifies the **Connection** that is connected to this **Connection**. *Joins* implies the start of this relationship.
	Attribute	Type	Description
	bidirectional	Boolean	bidirectional represents if the connection between two **Connection**s is bidirectional. If the **Connection** is not bidirectional the **Connection** that has the relation ***joins*** is the start and the Connection with ***joined by*** is the end.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Connection**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Connection**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.

3.4.7 Cost

Entity	Parent Entity	Child Entities	Description
Cost	None	None	A **Cost** entity specifies the outlay or expenditure (as of effort or sacrifice) made to achieve an objective associated with another entity. Examples: $100, 6 man-hours

Attribute	Type	Description
amount	Number	*amount* represents this **Cost**'s value.
category	Enumeration [Procurement, MILCON, MILPERS, O&M, RDT&E, SCN, **N/A**]	*category* represents the part of the lifecycle for which the money is used (commonly called Color of Money). This matches the Federal Government (DoD) cost phases.
contract type	Enumeration [CPFF, CPAF, CPIF, FFP-Completion, FFP-LOE, T&M, **N/A**]	*contract type* represents this **Cost** way to identify the reimbursement structure (i.e. CPFF vs T&M).
rate	Enumeration [**Fixed**, Per Hour]	*rate* represents this **Cost'**s billing rate.
units	Text	*units* represents this **Cost'***s* units used to measure the amount, such as $ or Euro.
type	Text	*type* provides aliases for the entities. For **Cost** these can include: Actual, Earned Value, Estimated, Overrun, Plan

Relationship	Inverse	Target Entity(es)	Description
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
decomposed by	***decomposes***	**Cost**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Cost**	***decomposes*** identifies the parent of this entity.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that is empowered by this entity.
incurred by	***incurs***	**Action, Artifact, Asset, Characteristic, Connection, Decision, input/Output, Risk, Statement**	***incurred by*** identifies the entity that has this **Cost** value.

located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Cost**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Cost**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.

3.4.8 Decision

Entity	Parent Entity	Child Entities	Description
Decision	None	None	A **Decision** entity specifies an opportunity to make a choice. Examples: slip schedule by two months, accept risk, hire ten new employees

Attribute	Type	Description
assumptions	Text	*assumptions* represents facts that are used as a basis for this **Decision**. These assumptions can be taken for granted by the decision maker.
justification	Text	*justification* represents the reason and context for making this **Decision**. For additional justification, the user may want to apply the ***enabled by*** relationship to link it to other **Statements**.
status	Enumeration [**Open**, Closed]	*status* represents the state of the **Decision** (open or closed).
type	Text	*type* provides aliases for the entities. For **Decision** these can include: Challenge, Issue, Problem.

Relationship	Inverse	Target Entity(es)	Description
alternative	***alternative of***	**Statement**	***alternative*** identifies the **Statement** that is a potential choice for this **Decision**.
	Attribute	Type	Description
	choice	Boolean	choice represents if this alternative was the choice selected. If the **Decision** is still open none of these alternatives would be True.
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
date resolved by	***date resolves***	**Time**	***date resolved by*** identifies the **Time** when this **Decision** was closed. For open issues this attribute can be left blank.
decomposed by	***decomposes***	**Decision**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Decision**	***decomposes*** identifies the parent of this entity.
decision due	***decided by***	**Time**	***decision due*** identifies the **Time** when this **Decision** is scheduled to be closed.

enabled by	***enables***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Input/Output, Risk, Statement**	***enabled** by* identifies the entity that empowers this **Decision** to be made.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
made by	***made***	**Asset**	***made by*** identifies the **Asset** that made this *Decision*.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Decision**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Decision**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the *Risk* that is closed by this entity.
responded by	***responds to***	**Asset**	***responded by*** identifies the **Asset** (usually a person or organization) that acts on this **Decision**.
	Attribute	Type	Description
	responsibility	Enumeration [**Primary**, Secondary]	*responsibility* represents the **Asset** that has the responsibility for resolving the **Decision**.

result of	***results in***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Decision, Input/Output, Risk, Statement**	***result of*** identifies the entity that caused this **Decision**.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that this **Decision** comes from.

3.4.9 Input/Output

Entity	Parent Entity	Child Entities	Description
Input/Output	None	None	An **Input/Output** entity specifies the matter, energy, and/or information input to, triggers (controls), or output from an Action. Examples: Nickel/gumball, gasoline/horsepower, investment/return

Attribute	Type	Description
size	Number	*size* represents the amount or proportion of this **Input/Output**, such as 100 as 100 Gigabytes or number of entities (e.g. 10 as in 10 tokens).
units	Text	*units* represents this **Input/Output'***s* units used to measure the size such as Gigabytes or tokens.
type	Text	*type* provides aliases for the entities. For **Input/Output** these can include: Analog, Data, Digital, Event, Information, Item, Mixed, Physical, Product, Verbal.

Relationship	Inverse	Target Entity(es)	Description
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
decomposed by	***decomposes***	**Input/Output**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Input/Output**	***decomposes*** identifies the parent of this entity.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that was empowered by this entity.
generated by	***generates***	**Action**	***generated by*** identifies the **Action** that this **Input/Output** was transformed by.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.

received by	***receives***	**Action**	***received by*** identifies the **Action** that takes in this *Input/Output*.
	Attribute	Type	Description
	trigger	Boolean	trigger represents this relation as an enabling requirement for the **Action**. An **Action** begins execution when it has received control enablement, all of its triggers have arrived, and its necessary resources are available.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Input/Output**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Input/Output**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.
transferred by	***transfers***	**Connection**	***transferred by*** identifies the **Connection** that passes this **Input/Output**.

3.4.10 Location

Entity	Parent Entity	Child Entities	Description
Location	None	**Orbital, Physical, Virtual**	A **Location** entity specifies where an entity resides. Abstract Class.

Relationship	Inverse	Target Entity(es)	Description
decomposed by	***decomposes***	**Location**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Location**	***decomposes*** identifies the parent of this entity.
locates	***located at***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Decision, Input/Output, Statement**	***locates*** identifies an entity that exists at this **Location**.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
related to	***relates***	**Location**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Location**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced from	***traced to***	**Statement**	***traced from*** identifies a **Statement** that accredited to this entity.

3.4.11 Logical (Connection)

Entity	Parent Entity	Child Entities	Description
Logical	**Connection**	None	A **Logical** entity specifies relationship between Assets. It is primarily used in database schema development and entity-relationship diagrams.

Relationship	Inverse	Target Entity(es)	Description
Note for specified by (see description)	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity. This relationship is used to specify an attribute on a relationship in an extended Entity-Relationship-Attribute (ERA) schema.

All attributes and relationships from **Connection** are inherited by the **Logical** entities.

3.4.12 Measure (Characteristic)

Entity	Parent Entity	Child Entities	Description
Measure	**Characteristic**	None	A **Measure** entity specifies the set of measurements used to provide managers, system developers, and systems engineers with insight into the system definition, the analysis of technical solutions with respect to performance, cost, and risk. Examples: 19 inches, 22 grams, 1.21 gigawatts

Attribute	Type	Description
improvement direction	Enumeration [N/A, **Positive**, Negative]	*improvement direction* represents the direction in that metric improvement occurs. It is the direction from the *threshold value* to the *objective value*.
objective value	Text	*objective value* represents the goal for this **Measure**.
projected value	Text	*projected value* represents this **Measure**'s expected value to be achieved with existing resources.
threshold value	Text	*threshold value* represents the minimum acceptable value for this **Measure**.
tolerance	Text	*tolerance* represents the percentage of the value that forms the positive and negative tolerance bands. *tolerance* is used when *value* represents the planned measure value at a given time.
type	Text	*type* provides aliases for the entities. For **Measure** these can include: Critical Operational Issue (COI), Key Performance Parameter (KPP), Mean Time Between Failures (MTBF), Measure of Effectiveness (MOE), Measure of Performance (MOP), Metric, Technical Performance Measure (TPM).

All attributes and relationships from **Characteristic** are inherited by the **Measure** entities.

3.4.13 Orbital (Location)

Entity	Parent Entity	Child Entities	Description
Orbital	**Location**	None	An **Orbital** entity specifies a location (ephemeris) along an orbit around a celestial body. Note that this includes transfer orbits as well. Examples: Mars orbit, transfer to lunar orbit

Attribute	Type	Description
Inclination	Number	*inclination* represents the angle between the orbital plane and a reference plane, such as the equatorial plane for Earth. The *Inclination* must be specified with the *longitude of ascending node* to characterize the orbital plane.
Semimajor Axis	Number	*semimajor axis* represents the one half of the length of the longest diameter of the orbital ellipse. The *semimajor axis* must be specified with the *eccentricity* to characterize the orbital ellipse shape.
Longitude of Ascending Node	Number	*longitude of ascending node* represents the angle from the *origin of longitude* to the ascending node, measured in the *reference plane*. For the Earth, the *origin of longitude* is typically the Prime Meridian. The *longitude of ascending node* must be specified with the *Inclination* to characterize the orbital plane.
Reference Plane	Text	*reference plane* represents the reference plane from where the Inclination angle will be calculated.
Argument of Periapsis	Number	*argument of periapsis* represents the angle between the Periapsis and the ascending node as measured in the orbital plane in the direction of motion.
Origin of Longitude	Text	*origin of longitude* represents the reference meridian from where the *longitude of ascending node* will be calculated.
Mean Anomaly	Number	*mean anomaly* represents the proportion of orbital area swept since the last *periapsis* at the specified time. It is used to define the position of the orbiting **Asset** along the orbital ellipse.
Apoapsis	Number	*apoapsis* represents the point of greatest distance from the celestial body being orbited. For Earth, the term is apogee. For the Sun the term commonly used is aphelion. The *apoapsis* must be specified with the *periapsis* to characterize the orbital ellipse shape.
Periapsis	Number	*periapsis* represents the point of closest distance from the celestial body being orbited. For Earth, the term is perigee. For the Sun the term commonly used is perihelion. The *periapsis* must be specified with the *apoapsis* to characterize the orbital ellipse shape.
Eccentricity	Number	*eccentricity* represents the amount the orbit deviates from a perfect circle (0 being perfectly circular and 1 is a parabola - no longer a closed orbit). The *eccentricity* must be specified with the *semimajor axis* to characterize the orbital ellipse shape.

All attributes and relationships from **Location** are inherited by the **Orbital** entities.

3.4.14 Physical (Location)

Entity	Parent Entity	Child Entities	Description
Physical	**Location**	None	A **Physical** entity specifies a location on, below, or above the surface of a celestial body. Examples: North Pole, Camp Lejeune

Attribute	Type	Description
type	Text	*type* provides aliases for the entities. For **Physical** these can include: Geospatial Location, Map Coordinates

Attribute	Data Type	Description
address	Text	*address* represents this **Location's** complete address.
altitude/depth	Number	*altitude/depth* represents the distance above (positive values) or below (negative values) the surface.
coordinates	GeoPoint	*coordinates* represents the coordinate points for this **Location** (GPS or other system).
units	Text	*units* represents the units used to measure the *altitude/depth* of the **Physical** location.

Note that this entity definition uses Cartesian Coordinates (x, y, z). It may be desirable to establish other coordinate systems (e.g., Cylindrical, Spherical, etc.) for other implementations.

All attributes and relationships from **Location** are inherited by the **Physical** entities.

3.4.15 Requirement

Entity	Parent Entity	Child Entities	Description
Requirement	**Statement**	None	A **Requirement** entity identifies a capability, characteristic, or quality factor of a system that must exist for the system to have value and utility to the user. Example: pump shall weigh no more than 1.2 kilograms.

Attribute	Type	Description
rationale	Text	*rationale* provides a place to capture the reason(s) behind this **Requirement**.
type	Text	*type* provides aliases for the entities. For **Requirement** these can include: Functional Requirement, Safety Requirement, Support Requirement, Verification Requirement

Note the quality attributes below are optional. Other sets of quality attributes may be provided by the tool developer or these may be user-definable. However, some form of quality attributes are recommended.

Attribute	Type	Description
clear	Boolean	*clear* represents if this **Requirement** is unambiguous and not confusing.
complete	Boolean	*complete* represents if this **Requirement** expresses a whole idea.
consistent	Boolean	*consistent* represents if this **Requirement** is not in conflict with other requirements.
correct	Boolean	*correct* represents if this **Requirement** describes the user's true intent and is legally possible.
design	Boolean	*design* represents if this **Requirement** does not impose a specific solution on design; says "what", not "how".
feasible	Boolean	*feasible* represents if this **Requirement** is able to be implemented with existing technology, and within cost and schedule.
modular	Boolean	*modular* represents if this **Requirement** can be changed without excessive impact on other requirements.
traceable	Boolean	*traceable* represents if this **Requirement** is uniquely identified, and able to be tracked to predecessor and successor lifecycle items/objects.
verifiable	Boolean	*verifiable* represents if this **Requirement** is provable (within realistic cost and schedule) that the system meets the requirement.

All attributes and relationships from **Statement** are inherited by the **Requirement** entities.

3.4.16 Resource

Entity	Parent Entity	Child Entities	Description
Resource	**Asset**	None	A **Resource** entity specifies a consumable or producible **Asset**. Example: $5, 2 kilowatts, natural gas

Attribute	Type	Description
initial amount	Number	*initial amount* represents this **Resource**'s starting amount.
maximum amount	Number	*maximum amount* represents this **Resource**'s maximum amount allowed.
minimum amount	Number	*minimum amount* represents this **Resource**'s minimum amount allowed.
units	Text	*units* represents this **Resource**'s units used to measure the amounts, such as each or tons.
type	Text	*type* provides aliases for the entities. For **Resource** these can include: Computer Resource, Human Resource, Fuel

Relationship	Inverse	Target Entity(es)	Description
consumed by	***consumes***	**Action**	***consumed by*** identifies the **Action** that uses this **Resource**. After the **Action** is completed the amount consumed is not returned to the **Resource**.
	Attribute	Type	Description
	amount	Number	amount represents how much of the resource is ***consumed by*** the ***Action***. Units are relative to the units selected for the **Resource**.
produced by	***produces***	**Action**	***produced by*** identifies the **Action** that creates this **Resource**. **Resources** are produced when the execution of the action completes.
	Attribute	Type	Description
	amount	Number	*amount* represents how much of the *Resource* is *produced by* the *Action*. Units are relative to the units selected for the *Resource*.

seized by	***seizes***	**Action**	***seized by*** identifies the **Action** that uses this **Resource**. After the **Action** has completed this **Resource** is released for use by other **Actions**.
	Attribute	Type	Description
	amount	Number	amount represents how much of the resource is ***captured by*** the **Action**. Units are relative to the units selected for the **Resource**.

All attributes and relationships from **Asset** are inherited by the **Resource** entities.

3.4.17 Risk

Entity	Parent Entity	Child Entities	Description
Risk	None	None	A **Risk** entity specifies the combined probability and consequence in achieving objectives. Example: the risk of a large meteorite hitting the earth in the next 100 years is low but that could cause the extinction of life as we know it.

Attribute	Type	Description
type	Text	*type* provides aliases for the entities. For **Risk** these can include: Cost Risk, Schedule Risk, Technical Risk

Attribute	Type	Description
consequence	Percent	*consequence* represents the level of effect from this **Risk** occurring.
consequence description	Text	*consequence description* represents the result of this **Risk** occurring.
mitigation status	Enumeration [**Accept**, Avoid, Mitigate, Transfer]	*mitigation status* represents the status of the mitigation technique for this **Risk**.
probability	Percent	*probability* or likelihood represents the chance that this **Risk** will occur.
status	Enumeration [**Open**, Duplicate, Declined, Resolved]	*status* represents the current state of this **Risk**.
trend	Enumeration [Decreasing, Increasing, New, **Unchanged**]	*trend indicates the change in the Risk over time as to whether it is increasing or decreasing or staying the same.*

Relationship	Inverse	Target Entity(es)	Description
caused by	***causes***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Decision, Input/Ouput, Risk, Statement**	***caused by*** identifies the entity that this **Risk** results.

causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
decomposed by	***decomposes***	**Risk**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Risk**	***decomposes*** identifies the parent of this entity.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that is empowered by this entity.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
mitigated by	***mitigates***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Decision, Location, Statement, Time**	***Mitigated by*** identifies the entity that contains the plan to alleviate this **Risk**.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Risk**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
relates	***related to***	**Risk**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolved by	***resolves***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Decision, Input/Output, Risk, Statement**	***resolved by*** identifies the entity that closes this **Risk**.

resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.

3.4.18 Statement

Entity	Parent Entity	Child Entities	Description
Statement	None	**Requirement**	A **Statement** entity specifies text referenced by the knowledgebase and usually contain in an Artifact. Example: Elvis is king!, Our goal is to be the first on Mars

Attribute	Type	Description
type	Text	*type* provides aliases for the entities. For **Statement** these can include: Acronym, Assumption, Constraint, Definition, Directive, Doctrine, Goal, Need, Objective, Plan, Policy, Question, Rule, Scope, Standard, Vision

Relationship	Inverse	Target Entity(es)	Description
alternative of	***alternative***	**Decision**	*alternative* of identifies the *Decision* that has this *Statement* as a potential choice.
	Attribute	Type	Description
	choice	Boolean	choice represents if this alternative was the choice selected.
causes	***caused by***	**Risk**	***causes*** identifies the **Risk** resulting from this entity.
decomposed by	***decomposes***	**Statement**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Statement**	***decomposes*** identifies the parent of this entity.
enables	***enabled by***	**Decision**	***enables*** identifies the **Decision** that is empowered by this entity.
incurs	***incurred by***	**Cost**	***incurs*** identifies the **Cost** value for this entity.
located at	***locates***	**Location**	***located at*** identifies the **Location** where this entity exists.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the **Risk** that this entity plan alleviates.
occurs	***occurred by***	**Time**	***occurs*** identifies the **Time** (or timespan) this entity happens.
references	***referenced by***	**Artifact**	***references*** identifies the **Artifact** that specifies and/or enhances the definition of this entity.
related to	***relates***	**Statement**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.

relates	***related to***	**Statement**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.
resolves	***resolved by***	**Risk**	***resolves*** identifies the **Risk** that is closed by this entity.
results in	***result of***	**Decision**	***results in*** identifies the **Decision** that is caused by this entity.
sourced by	***source of***	**Artifact**	***sourced by*** identifies the **Artifact** that contains this *Statement*.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.
traced to	***traced from***	**Action, Artifact, Asset, Characteristic, Connection, Cost, Input/Output, Location**	***traced to*** identifies an entity that is accredited to this **Statement**.

3.4.19 Time

Entity	Parent Entity	Child Entities	Description
Time	None	None	A **Time** entity specifies a point or period when an action, asset, process, or condition exists, finishes, starts, or continues. Example: Milestone A, Phase 2

Attribute	Type	Description
duration	Number	*duration* represents the period of time this **Time** occurs. A zero (0) duration indicates a milestone.
start	DateTime	*start* represents the time when this **Time** begins.
end	DateTime	*end* represents the time when this **Time** finishes. It can be computed from the *start* and *duration* attributes.
type	Text	*type* provides aliases for the entities. For **Time** these can include: Duration, Milestone, Point In Time, Time Frame

Relationship	Inverse	Target Entity(es)	Description
date resolves	***date resolved by***	**Decision**	***date resolves*** identifies the **Decision** that is closed at this **Time**.
decided by	***decision due***	**Decision**	***decided by*** identifies the **Decision** scheduled for closure at this **Time**.
decomposed by	***decomposes***	**Time**	***decomposed by*** identifies the children of this entity.
decomposes	***decomposed by***	**Time**	***decomposes*** identifies the parent of this entity.
mitigates	***mitigated by***	**Risk**	***mitigates*** identifies the *Risk* that this entity plan alleviates.
occurred by	***occurs***	**Action, Artifact, Asset, Characteristic, Conduit, Cost, Decision, Input/Output, Location, Risk, Statement**	***occurred by*** identifies the entity that happens at this **Time**.
related to	***relates***	**Time**	***related to*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	context represents a description of this relation.

relates	***related to***	**Time**	***relates*** identifies the entity that ties in a peer-to-peer way with this entity.
	Attribute	Type	Description
	context	Text	***context*** represents a description of this relation.
specified by	***specifies***	**Characteristic**	***specified by*** identifies a **Characteristic** that provides further information about this entity.

3.4.20 *Virtual (Location)*

Entity	Parent Entity	Child Entities	Description
Virtual	**Location**	None	A **Virtual** entity specifies a location within a digital network. Example: http://www.google.com

Attribute	Type	Description
address	URI	*address* represents the identification address using the Uniform Resource Identifier (URI) protocols.

All attributes and relationships from **Location** are inherited by the **Virtual** entities.

4 Visualizations

The following diagrams represent the common forms of visualization of information. They do not attempt to encompass all the possible visualizations. Only one is unique to LML: the Action Diagram. A number of similar models have been developed over the years to express functional sequencing, such as the Flow Charts, Activity Diagram in UML/SysML, Business Process Modeling Notation (BPMN), and others. Although these various notations are accurate, they use many different symbols, which often confuse the non-expert viewers or recipients of the visualization. As seen below, the visualization of this functional sequencing in LML is much simpler, but appears to have all the necessary information for language execution. The usual constructs are replaced by special cases of Actions to denote decision points.

The other visualizations should be considered standard diagrams, used over many years by different techniques. We also provide suggested diagrams that LML users may want consider as well. A few do not appear to require visualization beyond a hierarchy diagram. We denote which diagrams are appropriate for which classes.

4.1 Action Diagram (Mandatory for *Action* entities with children)

The Action Diagram (see Figure 4-1) represents the functional sequencing of **Actions** along with the data flow provided by the **Input/Output** entities. This combination of **Actions** with **Input/Outputs** is similar to the SREM "Behavior Diagram" or UML Activity Diagram. Without the **Input/Output** entities, the Action Diagram would be the equivalent of the classic Function Flow Block Diagram (FFBD) or IDEF 3. The main difference between LML and these other diagrams is the use of special kinds of **Actions** to replace the constructs used in these other languages. The construct set is shown below.

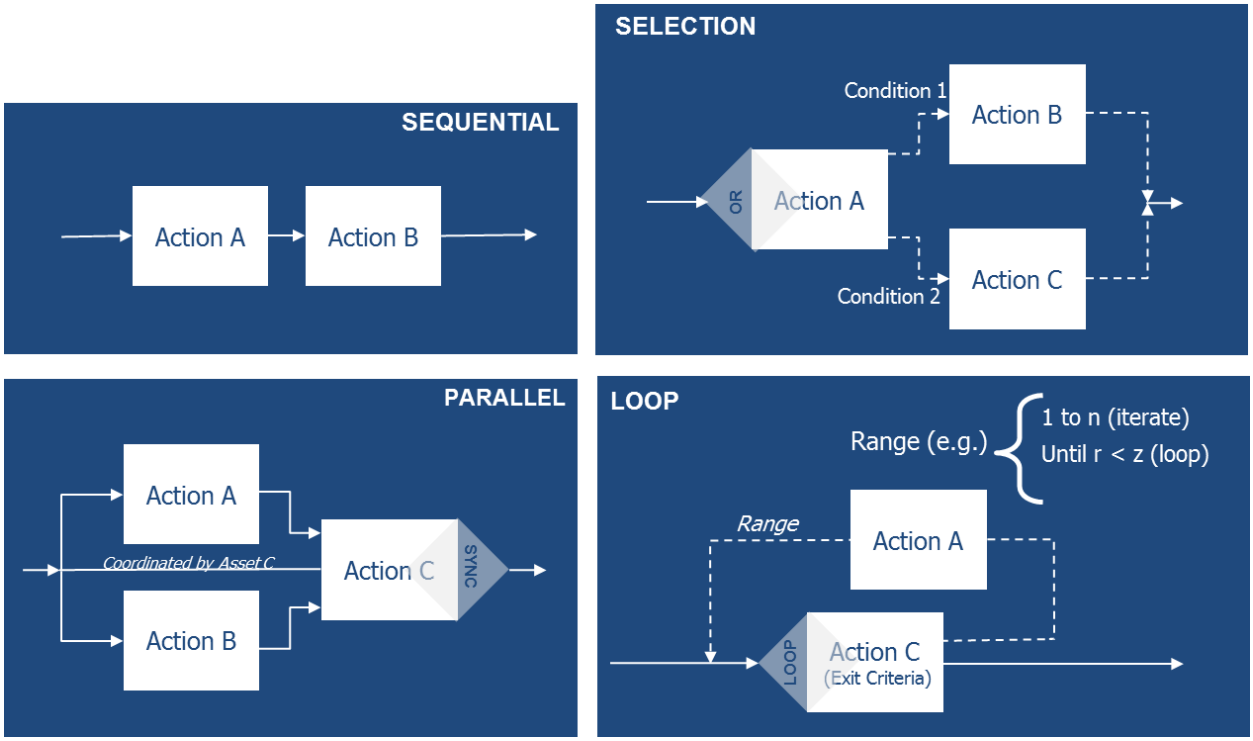

Figure 4-1. Special Actions for the Action Diagram to Capture Decision Points.

The special cases of **Actions**, denoted by the rectangle with a diamond embedded in it (showing ½ the diamond as a point on the rectangle), represent decision points. For example, in a loop the key decision is the exit criteria for the loop. This criterion can be as simple as the number of iterations of the loop or more complex logic that determines when the loop must stop.

The "OR" decision point represents and exclusive selection of one path or the other. The decision point in the case can be a probability or a specific criterion for path selection.

The final decision point type is the "SYNC." The SYNC provides the functional rationale for ending parallel branches. Note that in the physical view, two separate entities can exist without any synchronization between them. However, in the functional view between two physical entities that are interacting, it is necessary decide how to terminate that interaction. We see this in software, when two parallel processes are spawned these threads must later be synchronized to complete the program.

The Action Diagram can include **Input/Output** entities as well. An example of this inclusion is shown in Figure 4-2.

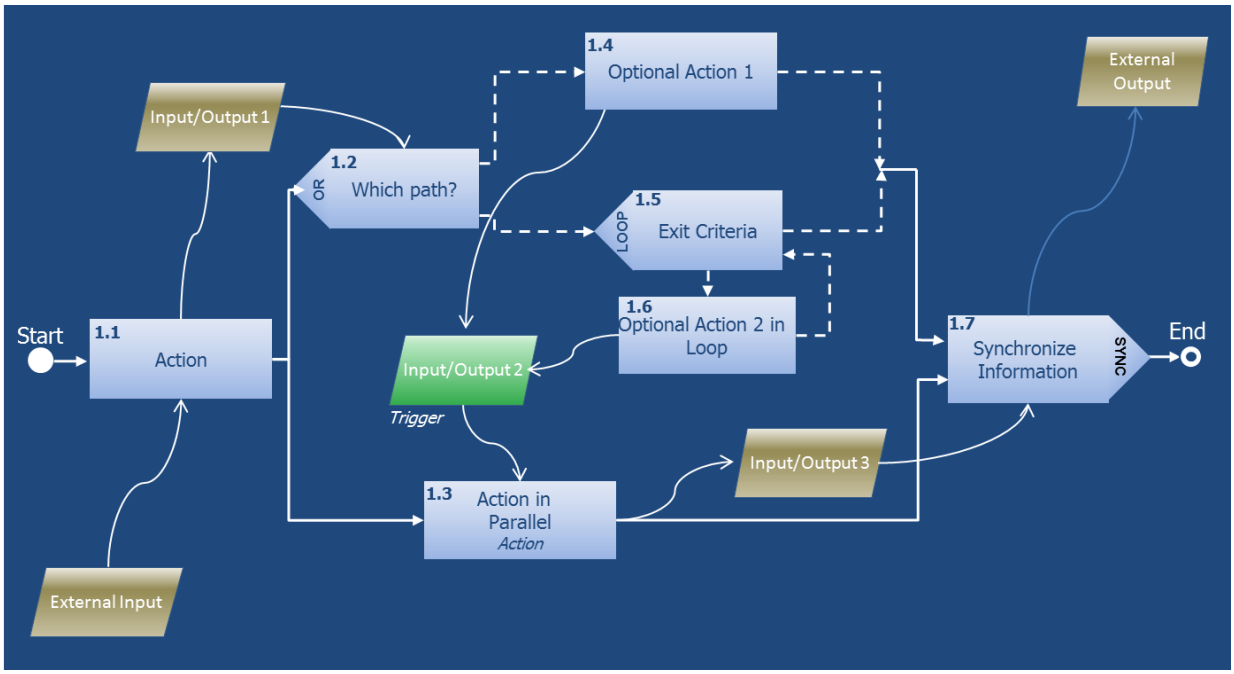

Figure 4-2. An Example Action Diagram with Inputs/Outputs.

The **Input/Output** entities are shown as parallelograms, reminiscent of the classical flow charting symbols used in the 1950s and 1960s. Two colors (or some other mechanism) are used to distinguish triggering **Input/Outputs** from optional **Input/Outputs**. It is recommended that a different type of line be used to show the **Input/Output** flow, thus making it easy to distinguish between the data flow and functional sequencing lines. The diagram should also contain a way to show the **Start** and **End** of the functional sequencing.

No other constructs have been determined to be necessary. Other languages include a “Replicate,” however the research done by the LML developers indicated that was a way identify a physical instantiation of the functional entity in more than one physical entity at a time. This representation appears more appropriately in the physical diagram (see Asset Diagram in next section).

While this specification does not directly specify or standardize the Action Diagram, the constructs above are to be used as guidance for the overall look of an Action Diagram. The key takeaway is the inclusion of logic flow into the Action entities themselves, instead of relying on separate logic flow diagram elements.

4.2 Asset Diagram (Mandatory for *Asset* entities with children)

LML must have a physical representation of design elements as well as the functional one provided by the Action Diagram. The Asset Diagram provides this kind of information. Figure 4-3 shows one way of providing this information. The **Assets** are shown as rectangles with **Conduits** displayed as lines connecting the **Assets**. Since **Resource** is a subclass of **Asset**, they could also by displayed by this diagram. Other information, such as the capacity and latency attributes of the **Conduit**, may be overlaid on this diagram as well. Also, since the **Asset** *type* may be of interest to most users it should be considered desirable to include that attribute on this diagram as well.

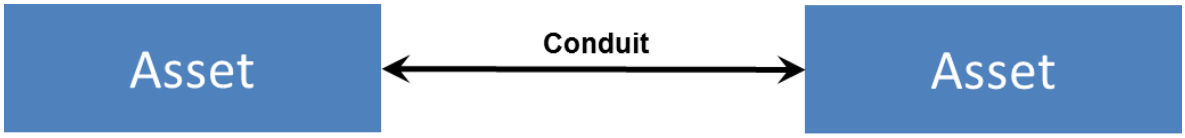

Figure 4-3. A Simple Asset Diagram.

Nodes and connections could be replaced with pictures as shown in Figure 4-4.

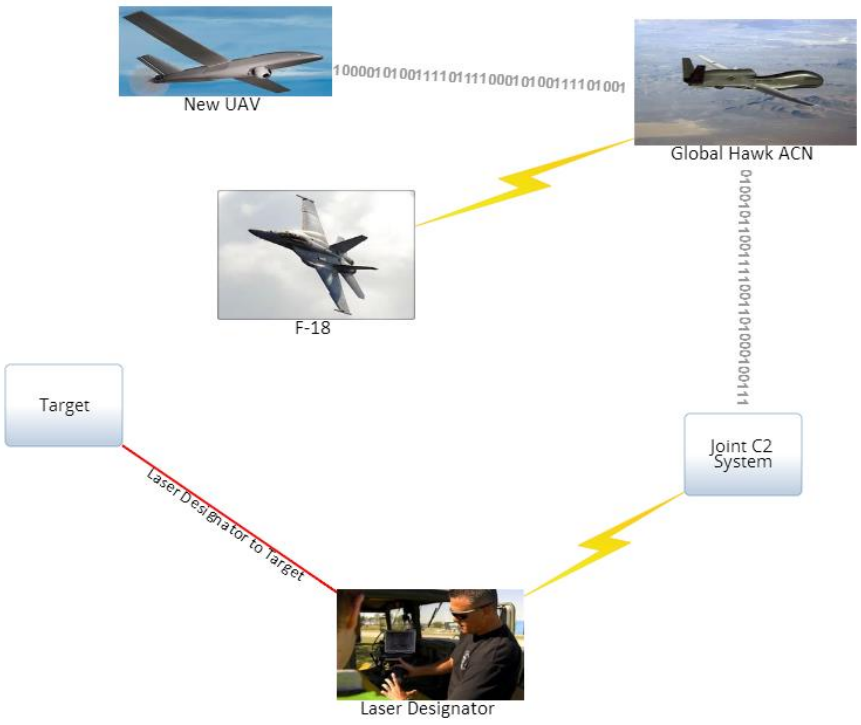

Figure 4-4. Asset Diagram with Pictures and Special Lines for Conduits.

The Asset Diagram is mandatory only for those Assets with children.

4.3 Spider Diagram (Mandatory for Traceability)

The spider diagram shows how entities are related to one another. This diagram is similar to the ERD shown above, but reflects LML's schema, not an abstract schema. This diagram (see Figure 4-5) shows traceability of LML entities with their relationships.

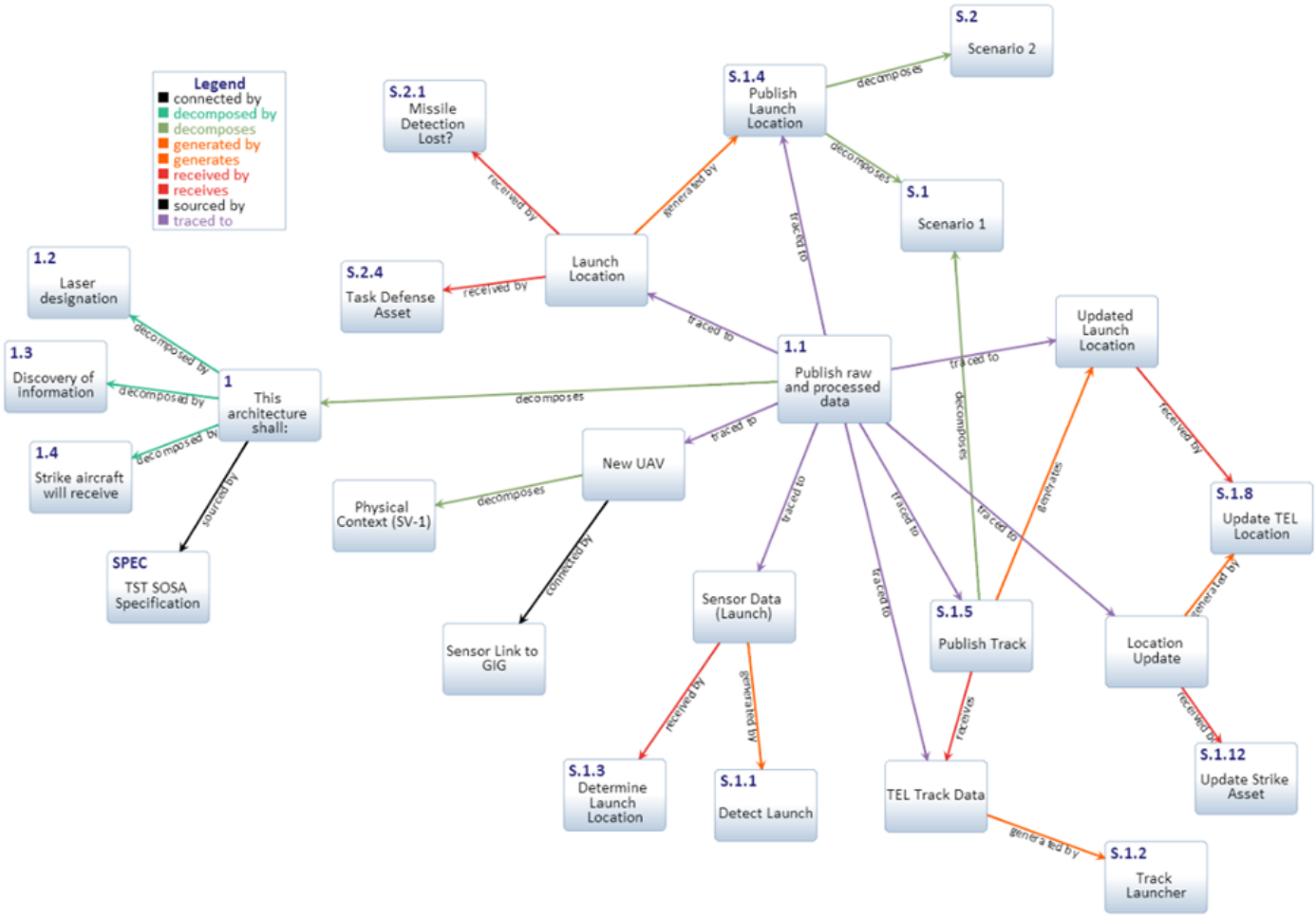

Figure 4-5. Example Spider Diagram to Show Traceability between Entities Using the LML Relationships.

4.4 Example Views for Other LML Entities

LML was designed to support the full lifecycle and all of its entities can support the common visualizations used by architects, systems engineers, software engineers, test engineers, operators, logisticians, and program managers. The examples below are suggestions of how to implement these common visualizations using LML.

4.4.1 Class Diagram

The diagram below shows how to implement the UML Class Diagram using LML classes. Since the Class Diagram has become a standard for software development, it seems a good candidate for inclusion in LML's approach. In this diagram, the LML entities that match the diagram elements include: **Asset** with the *type* "object"; attributes are captured as **Characteristics**; methods are **Actions**; and relationships are **Logical** connections.

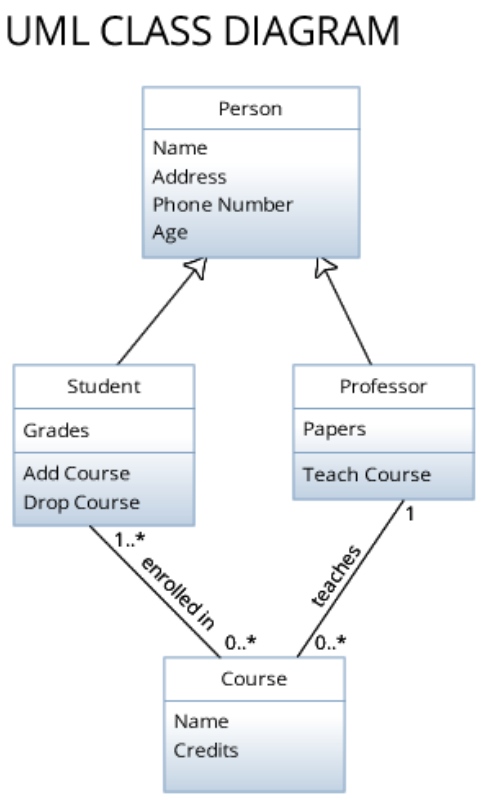

Figure 4-6. LML Encourages Use of the Class Diagram and Provides the Schema Entities to Support Its Creation.

4.4.2 Entity-Relationship Diagram

Another way to model information has been the classic entity-relationship diagram (ERD). An example of this is shown below for the LML schema. The relationships are expressed using the **Logical Connection**. Entities are represented by **Assets**. You can also capture attributes as **Characteristics**.

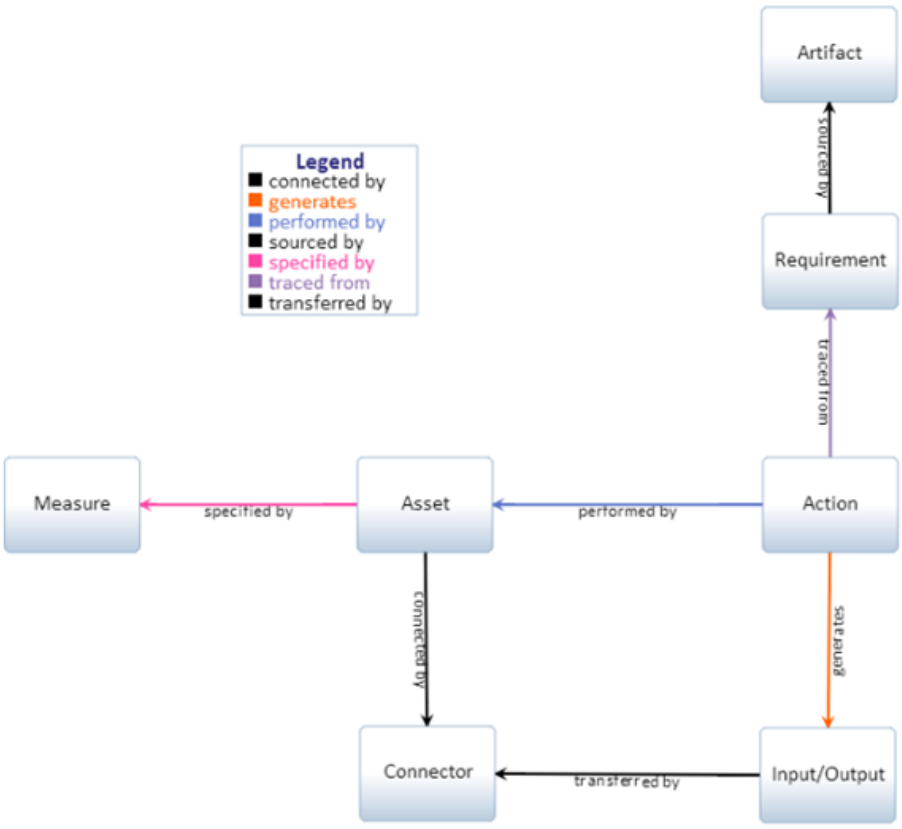

Figure 4-7. Entity-Relationship Diagram Developed Using the Asset Diagram.

4.4.3 Timelines

The **Actions** and **Times** can be displayed using a classic Gantt Chart shows how functional elements execute over time. An example of this is shown below.

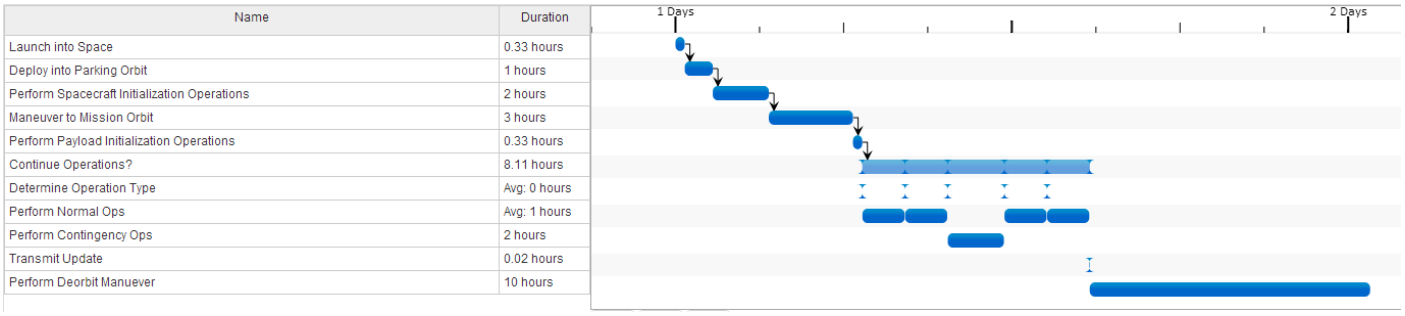

Figure 4-8. Use of Gantt Chart for Displaying Actions and Durations.

Time with **Actions** or **Assets** can be also be visualized in many other ways. One of the most useful is shown below.

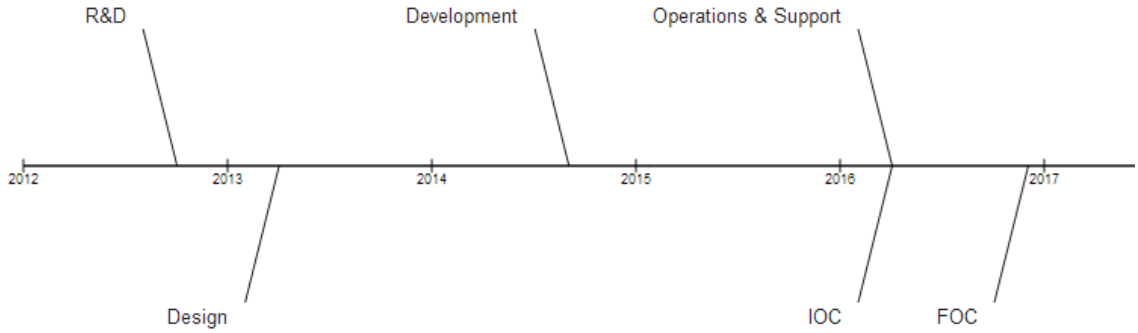

Figure 4-9. Timeline Diagram Showing Actions at Specific Times.

4.4.4 Hierarchy Diagram

A hierarchy chart is used in LML to show decomposition of elements. The figure below shows an example of requirements decomposition. This chart uses the ***decomposed by*** relationship.

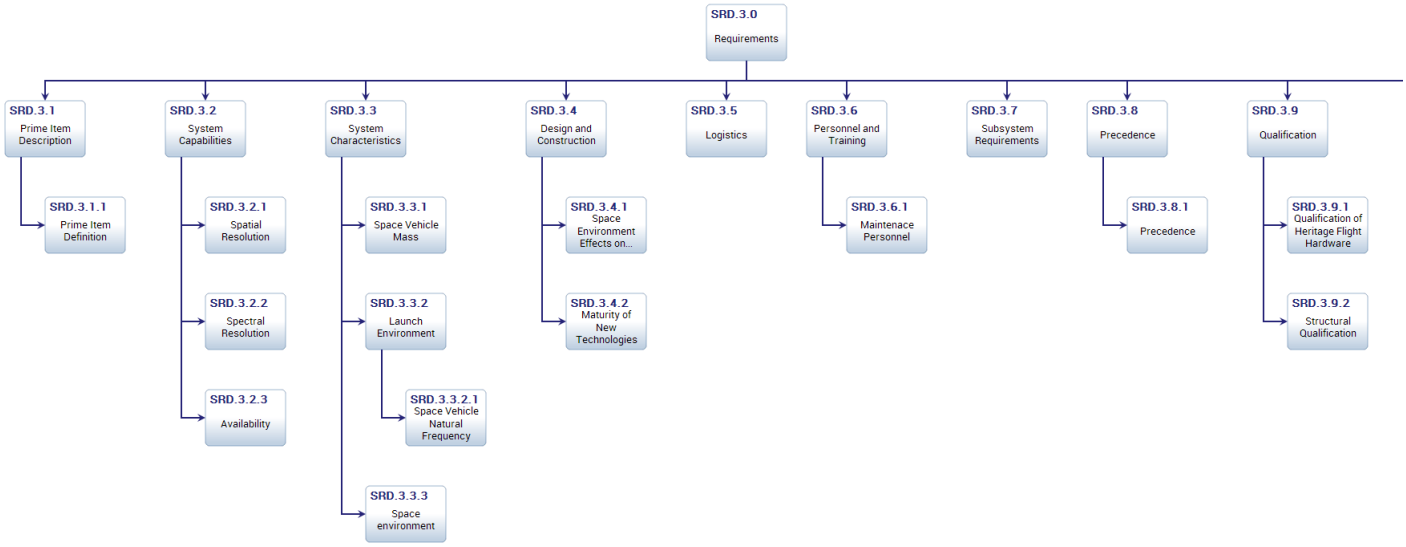

Figure 4-10. A Hierarchy Diagram Is a Good Way to Show Decomposition.

4.4.5 Risk Matrix

A standard DoD risk matrix (shown below) or other form can be used to display **Risk** entity information. Another type of risk analysis uses probabilities to create a fault-tree. A fault tree is often shown as a hierarchy diagram with the probabilities shown for each branch.

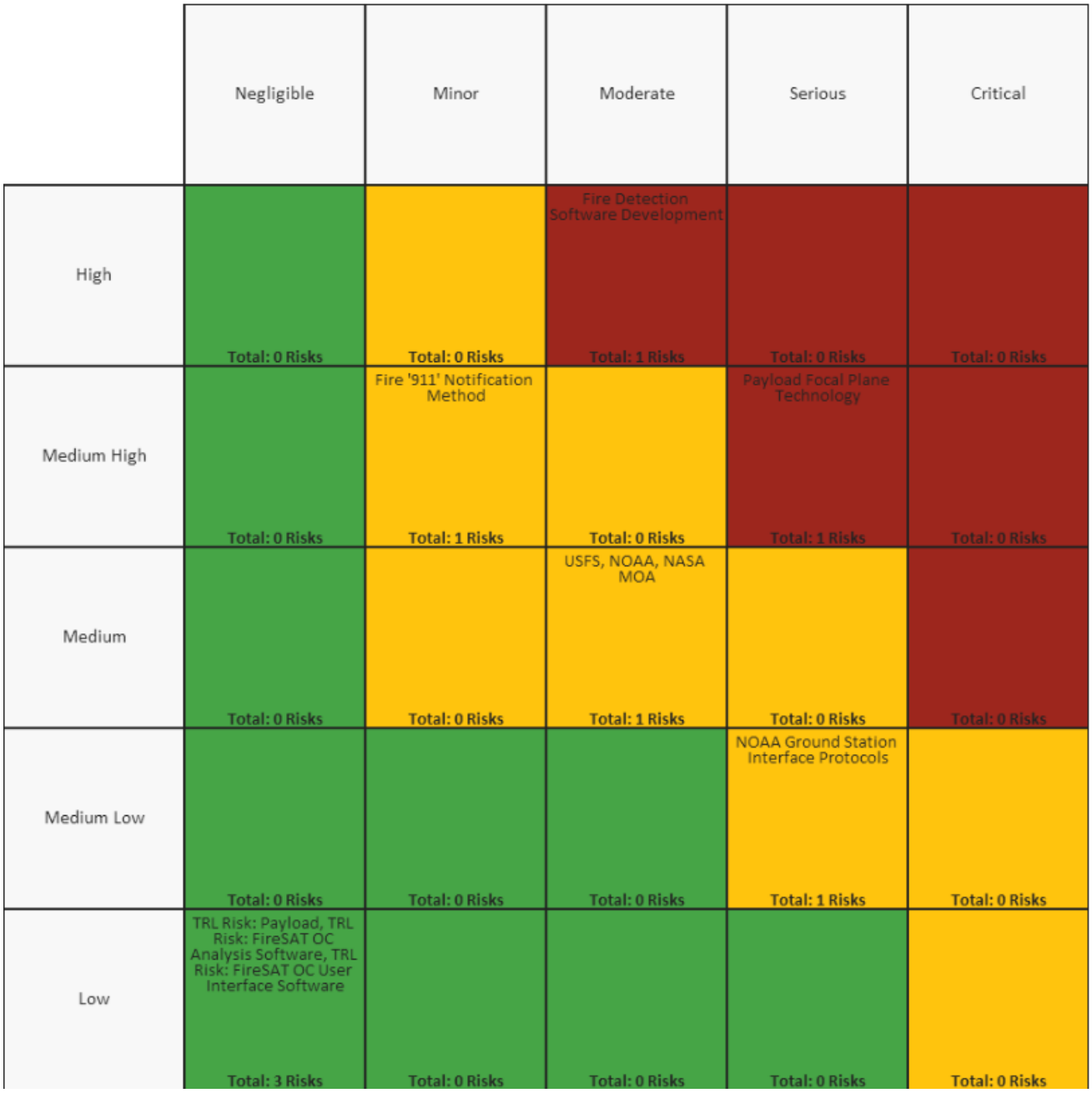

Figure 4-11. Typical Risk Matrix.

4.4.6 State Machine Diagram

The state machine diagram expresses how an **Asset** transitions from one state to another. In the diagram below, the state (or **Characteristic**) transition occurs when the **Action** entity event causes the transition to the other state.

Name	Class	Description
State	**Characteristic**	Means that it's a state of the system
Initial State	**Characteristic**	Means that it's the initial state
Final State	**Characteristic**	Means that it's the final state.

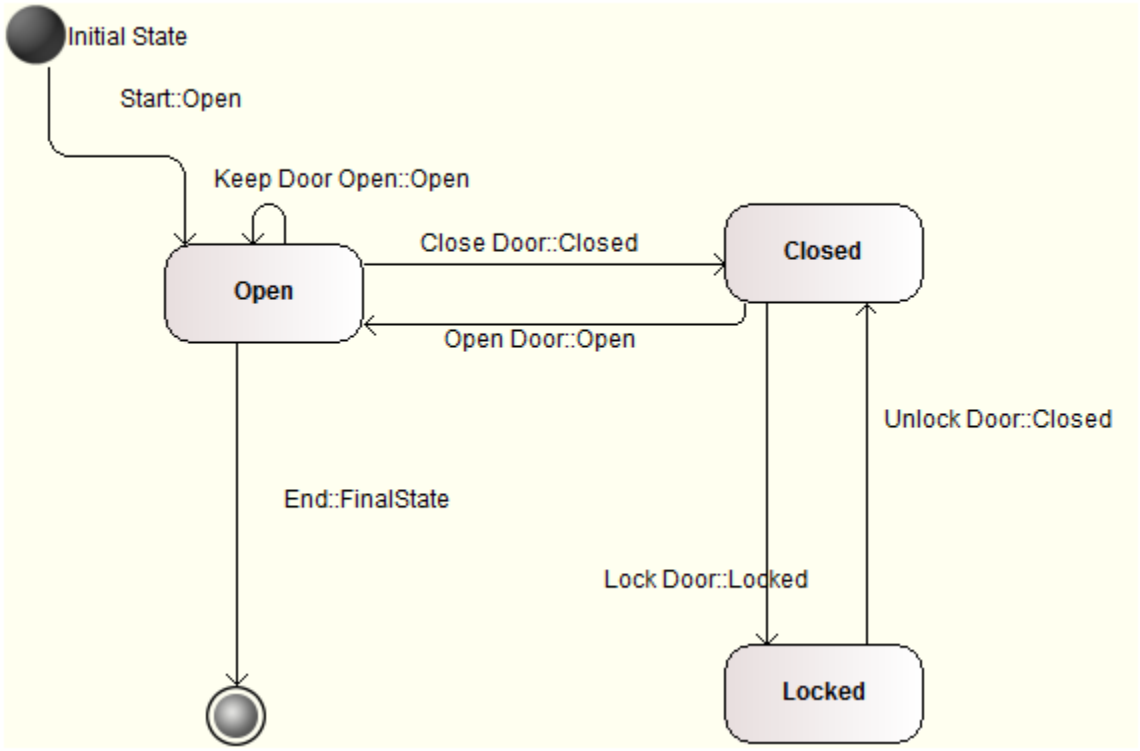

Figure 4-12. The State Machine Diagram Has Proven Useful and LML Supports It.

Appendix A. SysML Mapping to LML

SysML focuses mainly on diagrams, with an underlying ontology embedded in the XMI that each diagram represents. Currently (October 2013) a complete ontology is under development. The table below shows the various SysML diagrams and the LML equivalent diagram and associated classes.

Table A-1. SysML Diagram Mapping to LML Diagrams and Ontology

SysML Diagram	LML Diagram	LML Entities
Activity	Action Diagram	**Action**, **Input/Output**
Sequence	Sequence	**Action**, **Asset**
State Machine	State Machine	**Characteristic** (State), **Action** (Event)
Use Case	Asset Diagram	**Asset**, **Connection**
Block Definition	Class Diagram, Hierarchy Chart	**Input/Output** (Data Class), **Action** (Method), **Characteristic** (Property)
Internal Block	Asset Diagram	**Asset**, **Connection**
Package	Asset Diagram	**Asset**, **Connection**
Parametric	Hierarchy, Spider, Radar	**Characteristic**
Requirement	Hierarchy, Spider	**Requirement** and related entities

Appendix B. DoDAF MetaModel 2.0 (DM2) Mapping to LML

The Department of Defense has developed a schema called the DoDAF (DoD Architecture Framework) MetaModel 2.0 (DM2). The Conceptual Data Model is shown below. The physical data model contains over 500 entries. As can be seen this is a specialized model that focuses mainly on the DoD nomenclature and may be less useful in other domains. For example, since the DoD has developed their acquisition process around the concept of "Capability" that becomes a critical item in the top level of this schema, thus driving additional relationships. In LML we identified that capability could be a type of **Action**, **Asset**, **Characteristic** or even a **Statement**. However, LML does not preclude the user or tool vendor from adding "Capability" as an entity class. Schema extensions are actually encouraged for different domains. As such extensions are made and standardized; the LML Steering Committee will consider adding them as extensions to LML.

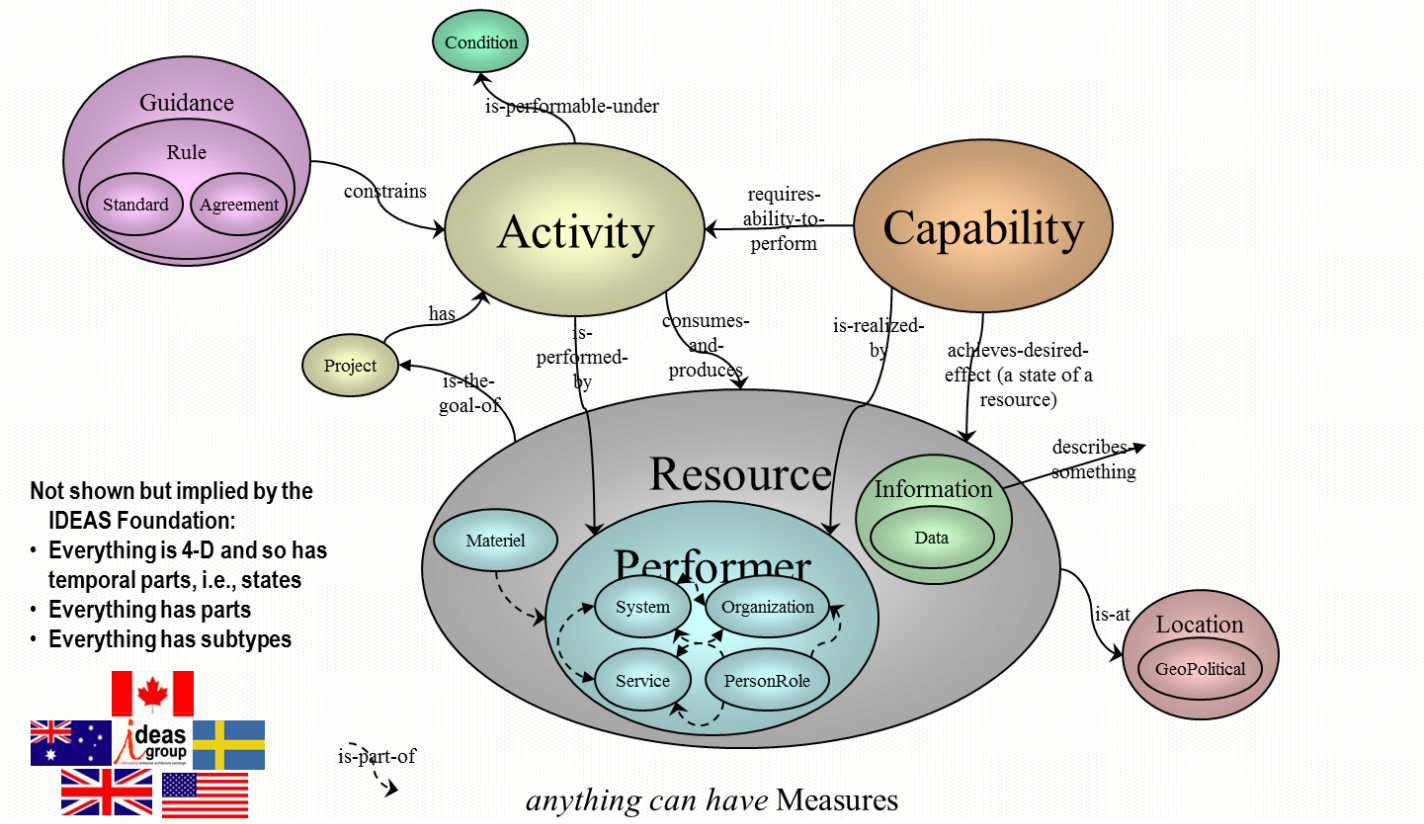

Figure B-1. DM2 Conceptual Data Model.

For a quick guide from the DM2 schema to LML, please see the table below.

Table B-1. DM2 Conceptual Data Model Mapping to LML

DM2 Schema Element (Conceptual)	LML Equivalent
Activity	Action
Capability	Action with “Capability” label
Condition	Characteristic with “Condition” label
Information/Data	Input/Output
Desired Effect	Statement with “Desired Effect” label
Guidance	Statement with “Guidance” label
Measure	Measure
Measure Type	Measure labels
Location	Location
Project	Action with “Project” label
Resource	Asset with labels for “Materiel,” “Organization,” etc.
Skill	Characteristic with label “Skill”
Vision	Statement with label “Vision”