Reusable Textual Styles for Domain-Specific Modeling Languages

Patrick Neubauer[1], Robert Bill[2], Dimitris Kolovos[1], Richard Paige[3], Manuel Wimmer[4]

[1] University of York, UK

[2] Technische Universität Wien, Austria

[3] McMaster University, Canada

[4] Johannes Kepler Universität, Austria

19th International Workshop in OCL and Textual Modeling, 2019, Munich, Germany

Roadmap

- Introduction and Background
  - Domain-Specific (Modeling) Languages
  - Grammar-first approach
  - Model-first approach
- Example Scenario
  - Language Structure
  - Language Style
- Approach
  - Style Specification Language
  - Style Modeling
- Conclusion and Future Work

Introduction and Background

- Domain-Specific Languages
  - Context-Free Grammar
  - Extended Backus-Naur Form (structure + representation)
- Domain-Specific Modeling Languages
  - OMG Meta Object Facility
  - Ecore for structural semantics (+ annotations for representation)
- Language Workbenches and Related Work
  - ***Xtext*** framework by Eysholdt et al.
  - ***Spoofax*** by Kats et al.
  - Model-Based Language Engineering with ***EMFText*** by Heidenreich et al.
  - ***Textual Concrete Syntax*** (TCS) by Jounault et al.
  - ***Textual Concrete Syntax Specification Language*** (TCSSL) by Fondement et al.
  - ***Textual Editing Framework*** (TEF) by Scheidgen et al.
  - ***SugarJ***: library-based syntactic language extensibility by Erdweg et al.
  - Classification of Concrete Textual Syntax Mapping Approaches by Goldschmidt et al.

Eysholdt, M., Behrens, H.: Xtext: implement your language faster than the quick and dirty way. SPLASH/OOPSLA 2010.

Kats, L.C.L., Visser, E.: The spoofax language workbench: rules for declarative specification of languages and ides. SPLASH/OOPSLA 2010.

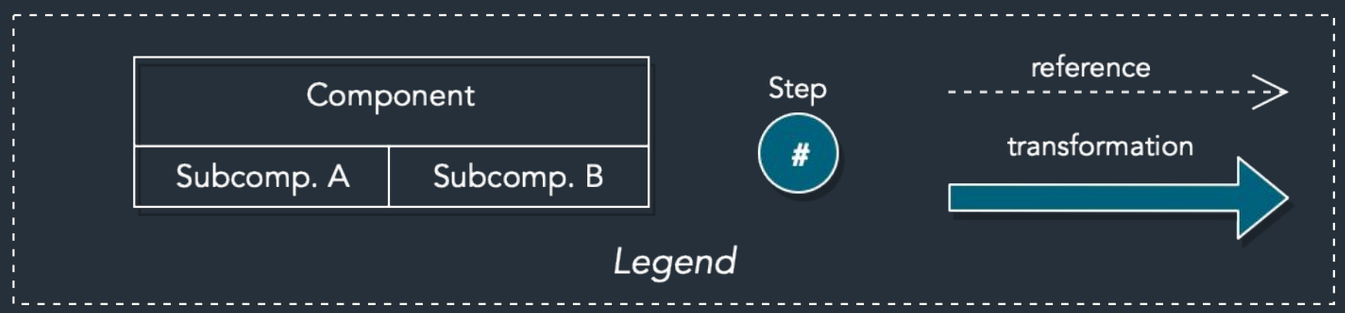
Heidenreich, F., Johannes, J., Karol, S., Seifert, M., Wende, C.: Derivation and refinement of textual syntax for models. ECMDA-FA 2009.

Jouault, F., Bézivin, J., Kurtev, I.: TCS: a DSL for the specification of textual concrete syntaxes in model engineering. GPCE 2006.

Scheidgen, M.: Textual modelling embedded into graphical modelling. ECMDA-FA 2008.

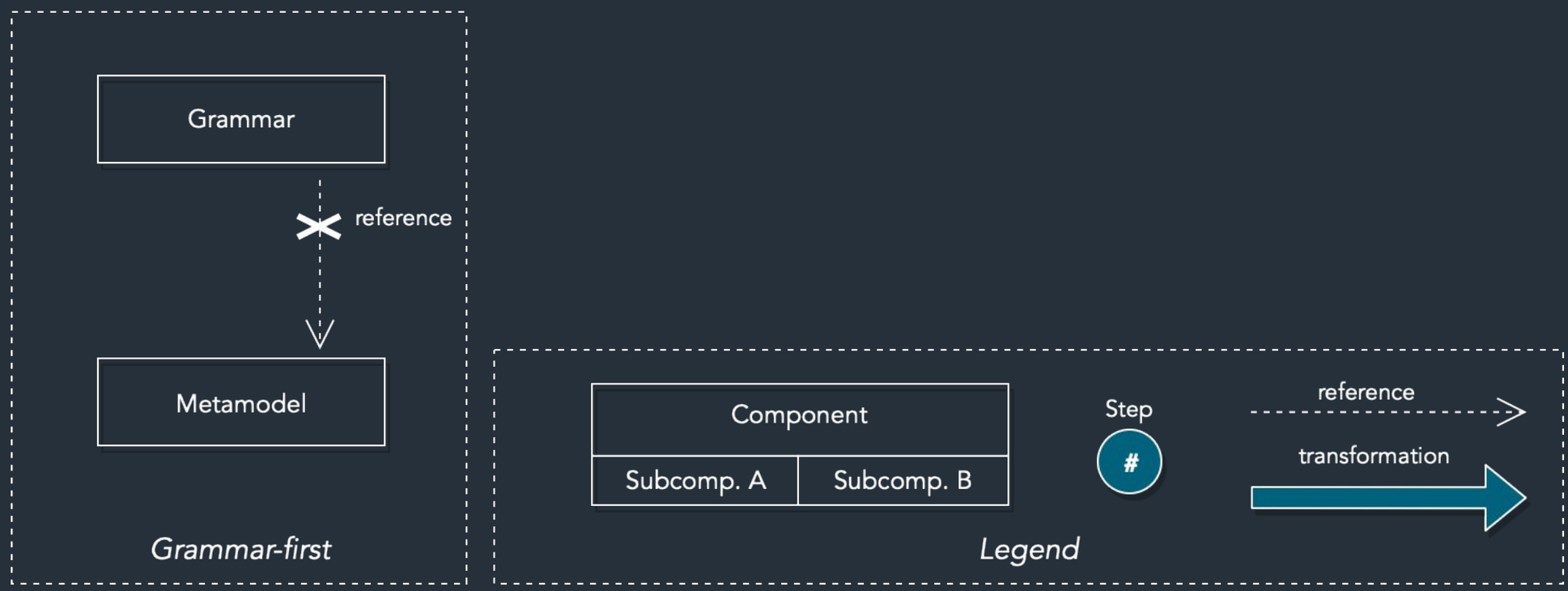
Grammar-first Approach

How to construct DSMLs by applying *grammar-first* approach?



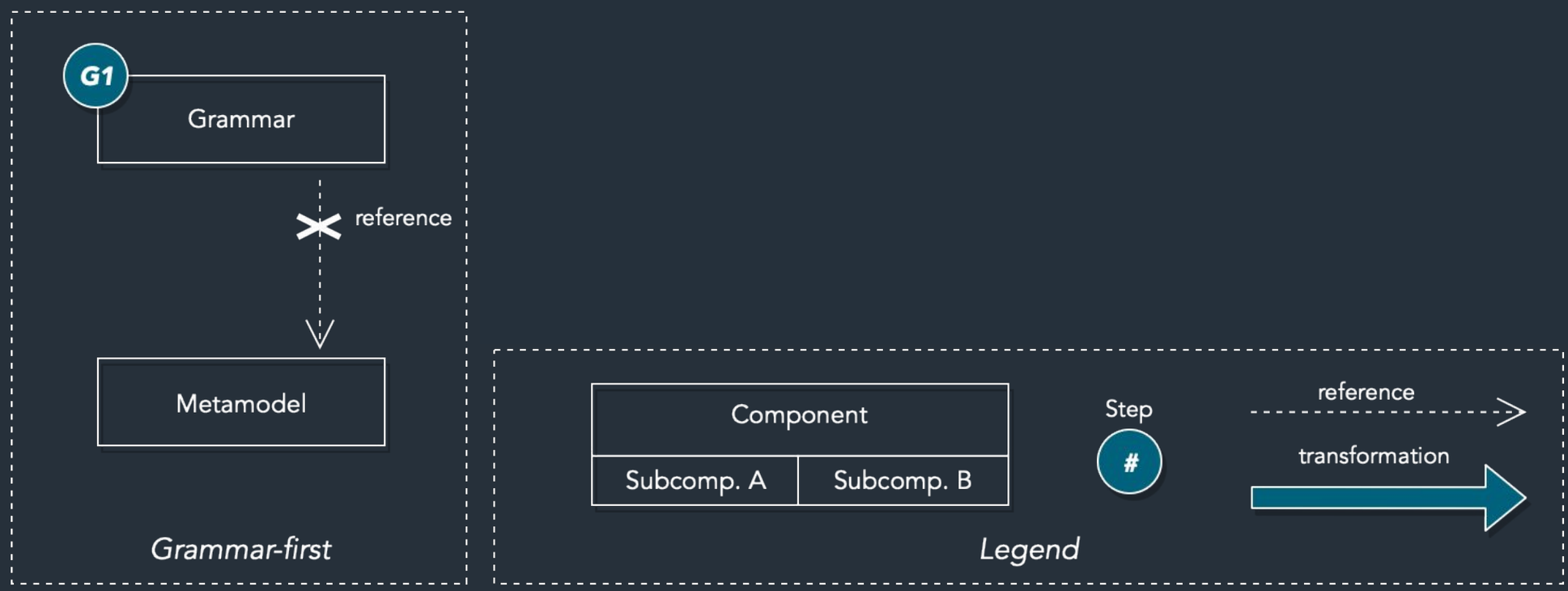
Grammar-first Approach

Grammar does *not* reference metamodel



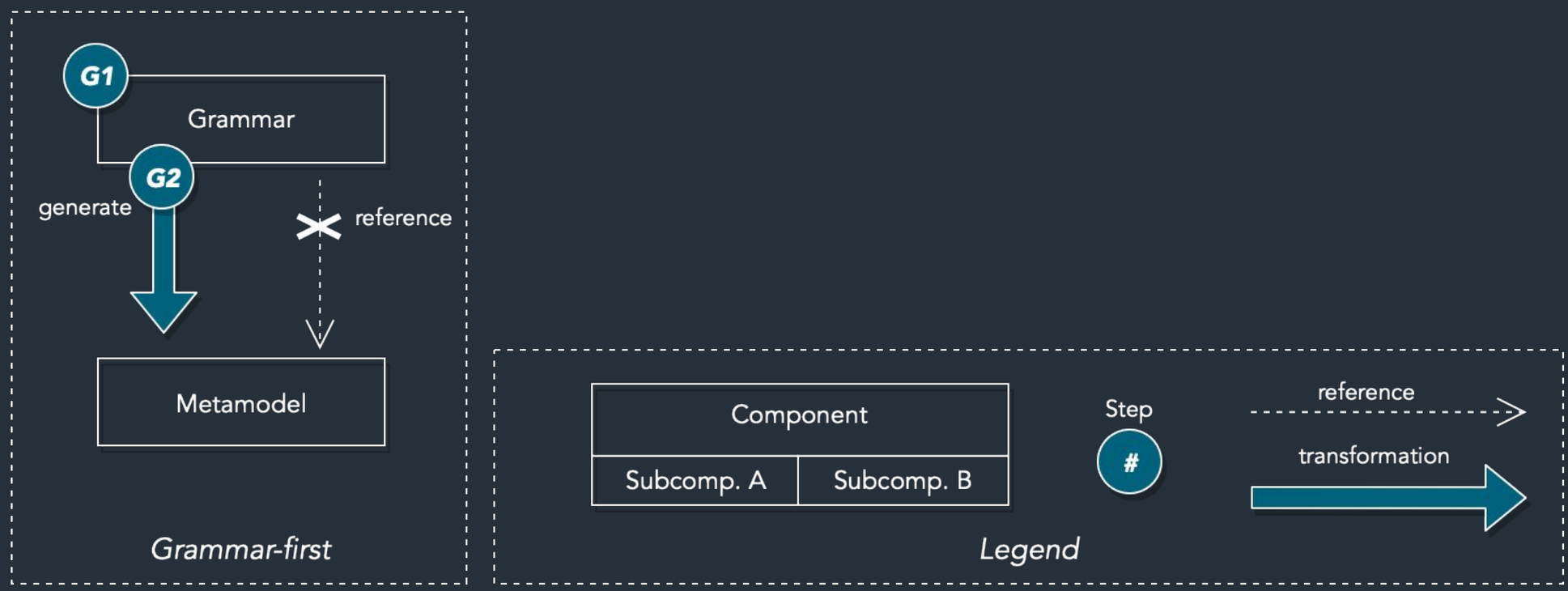
Grammar-first Approach

Create grammar



Grammar-first Approach

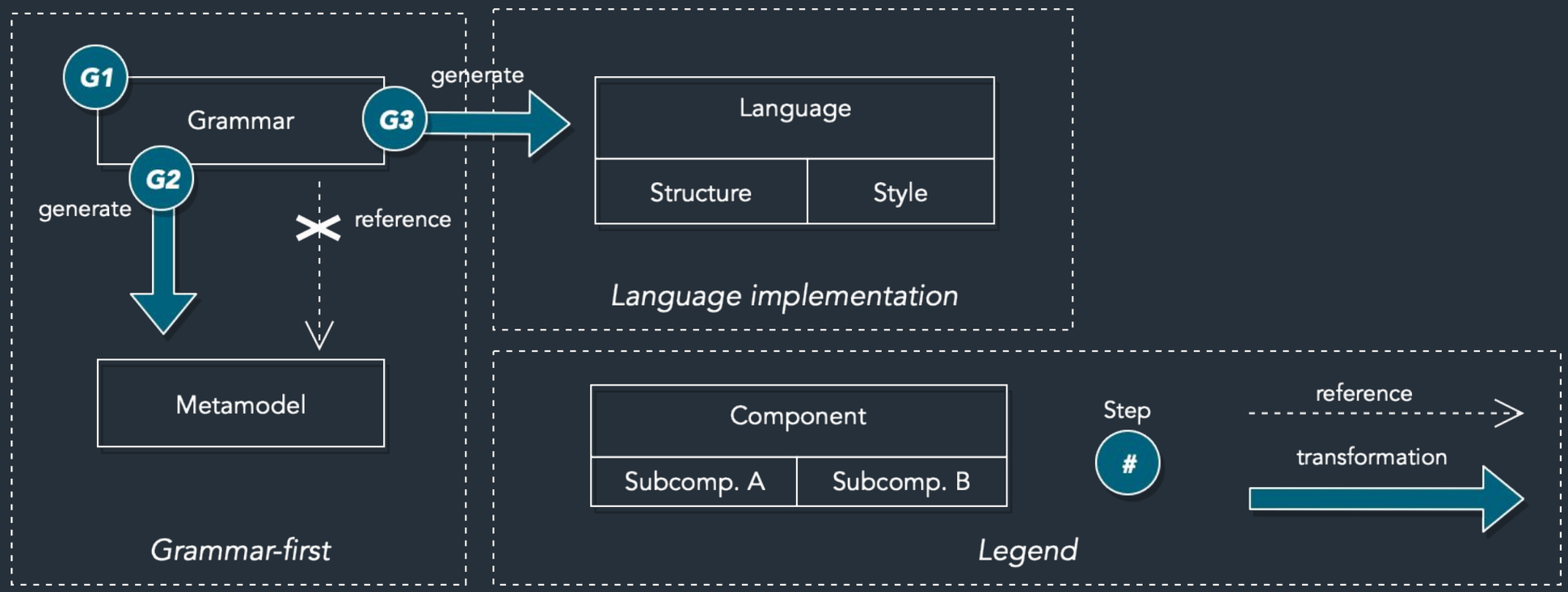
Derive metamodel from grammar



Grammar-first Approach

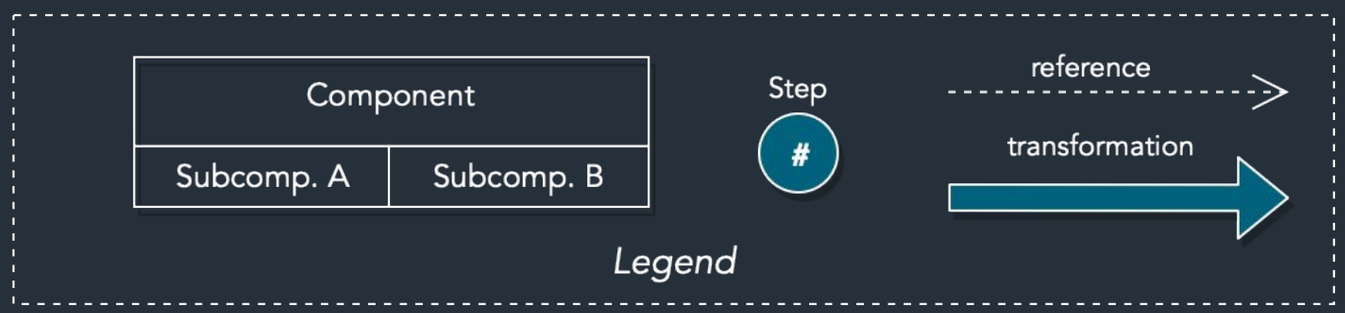
Generate language implementation from grammar

- Generated tooling employs *derived* metamodel



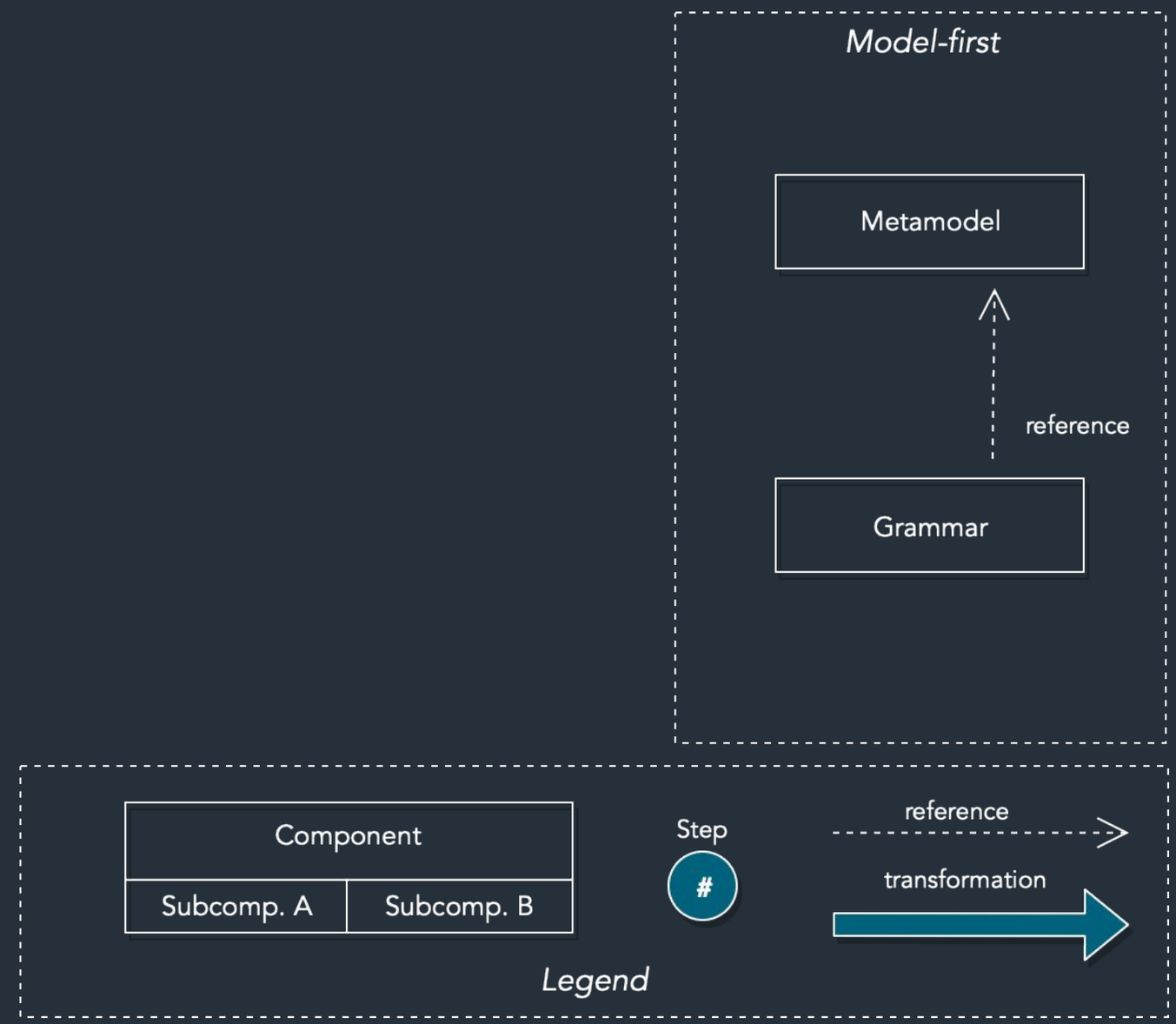
Model-first Approach

How to construct DSMLs by applying *model-first* approach?



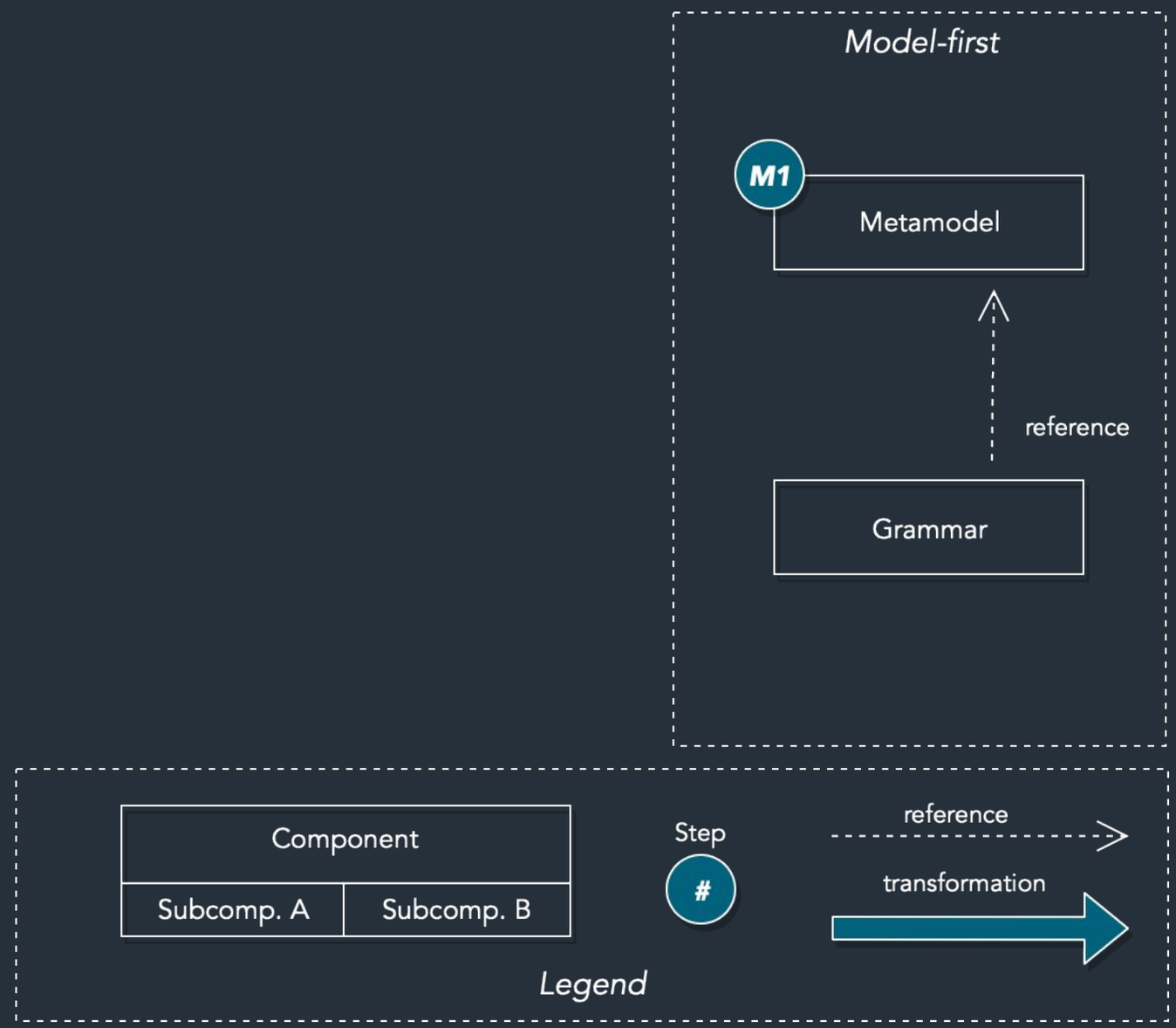
Model-first Approach

Grammar *does* reference metamodel



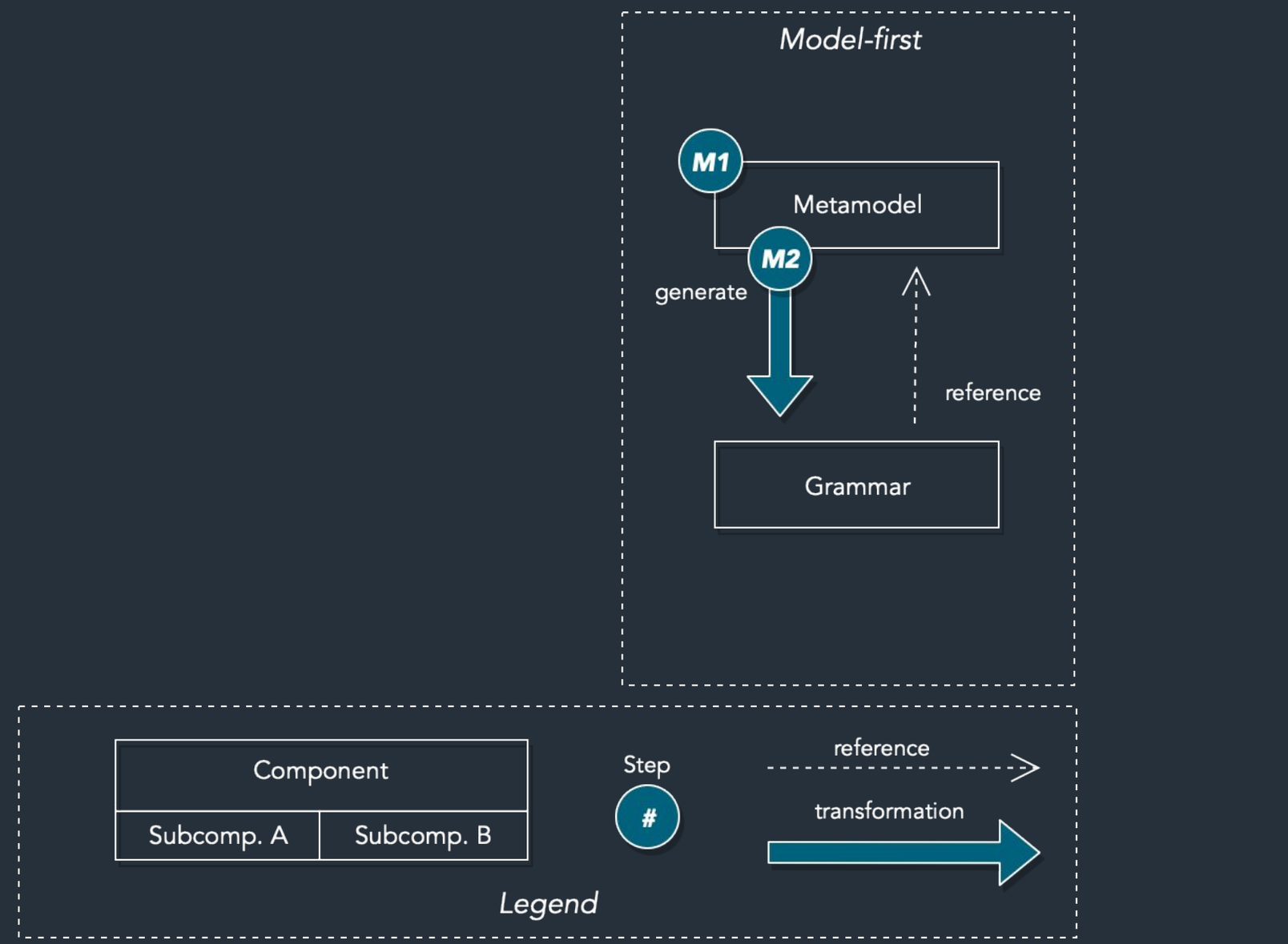
Model-first Approach

Create metamodel



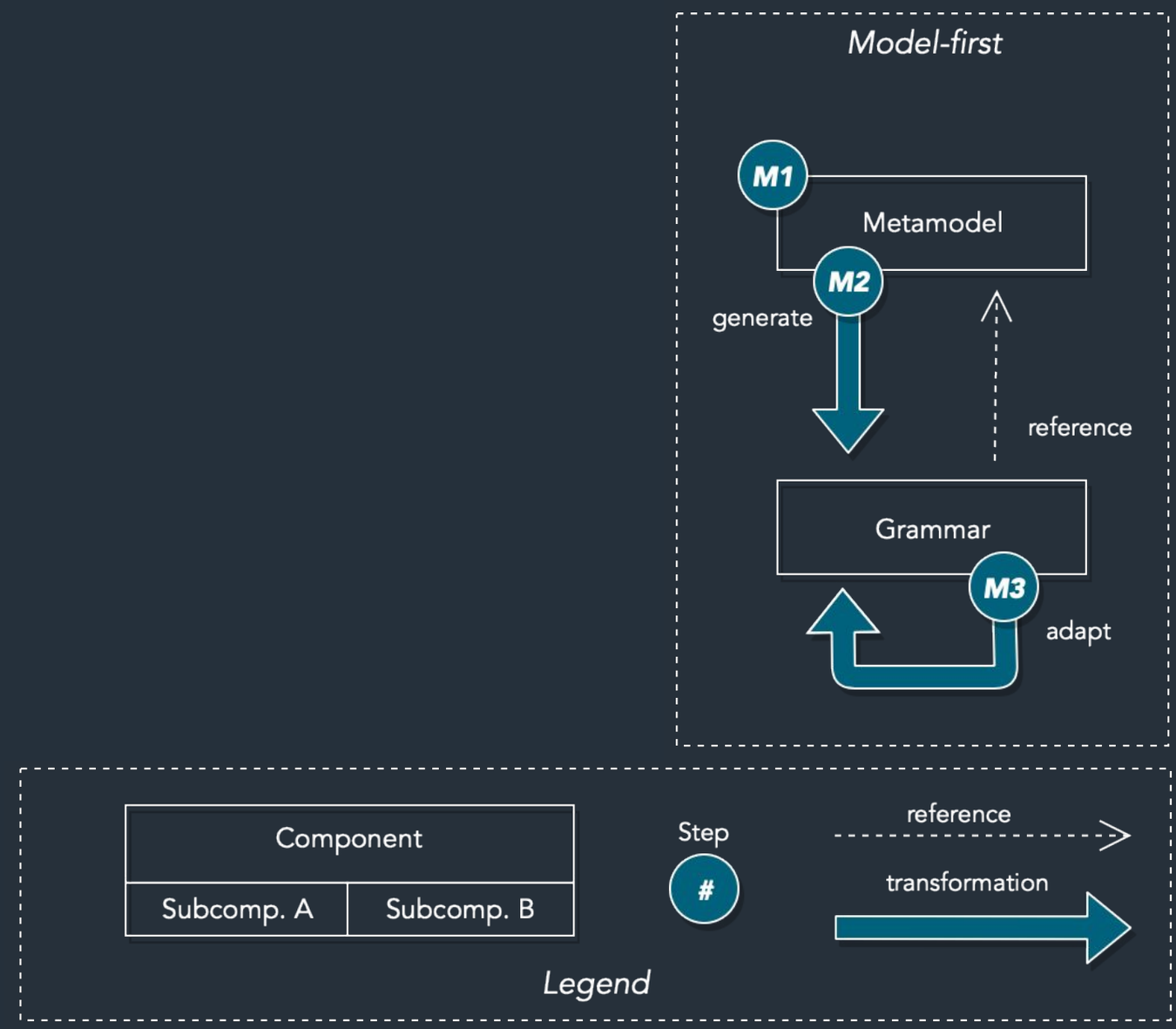
Model-first Approach

Generate grammar from metamodel



Model-first Approach

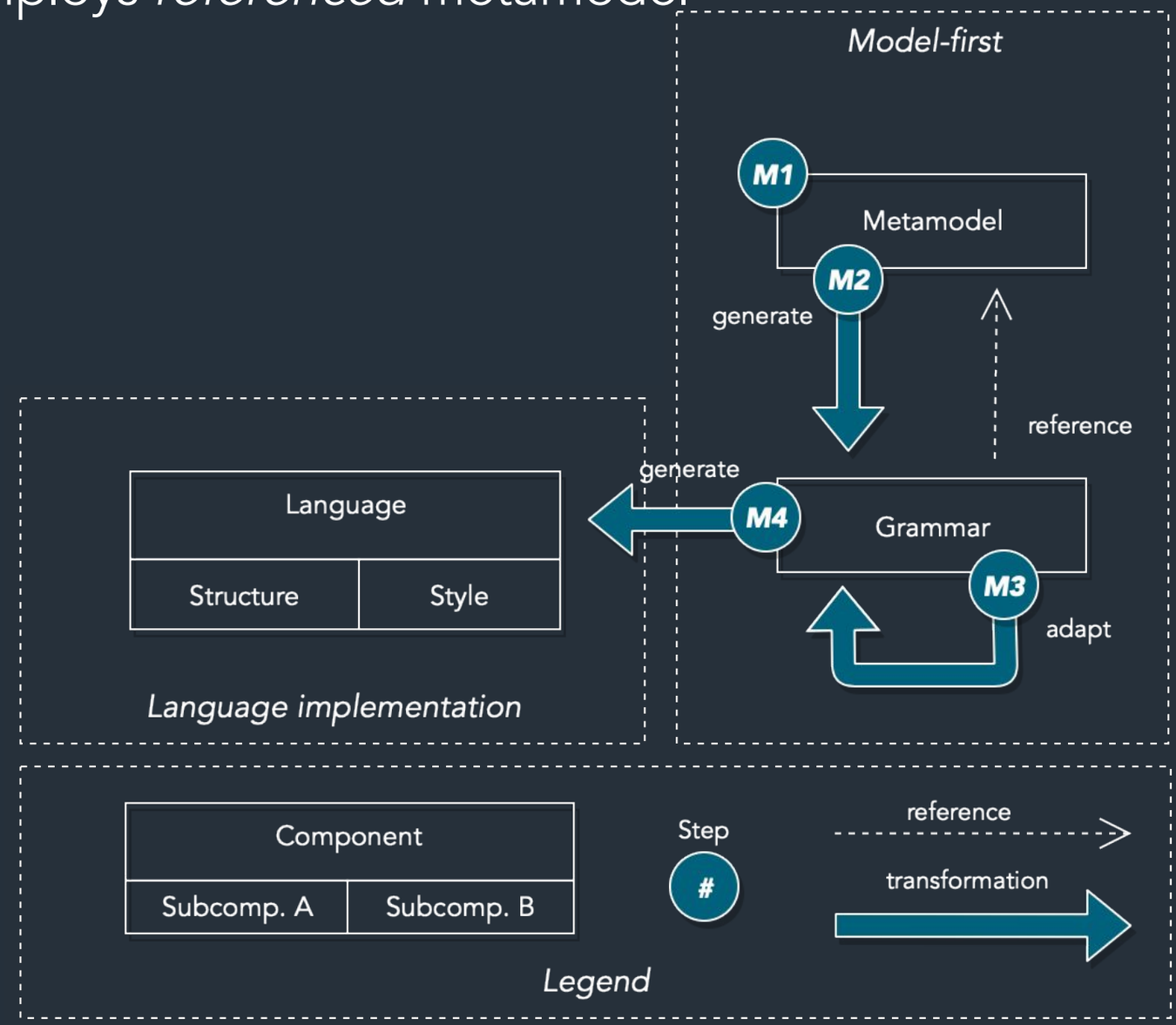
Adapt generated grammar to reflect desired representation



Model-first Approach

Generate language implementation from metamodel-based grammar

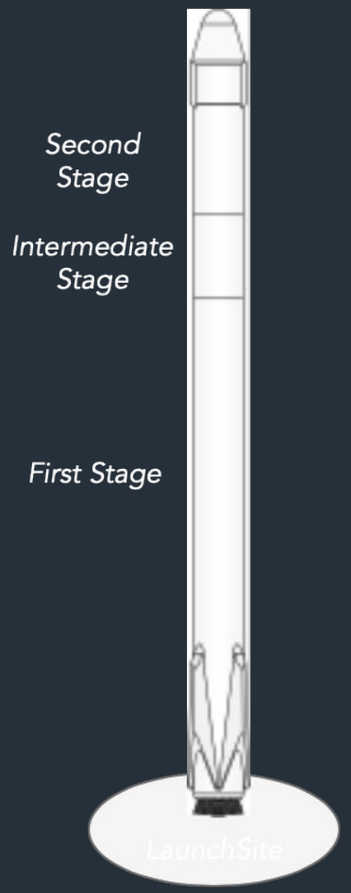
- Generated tooling employs *referenced* metamodel



Example Scenario: Structure

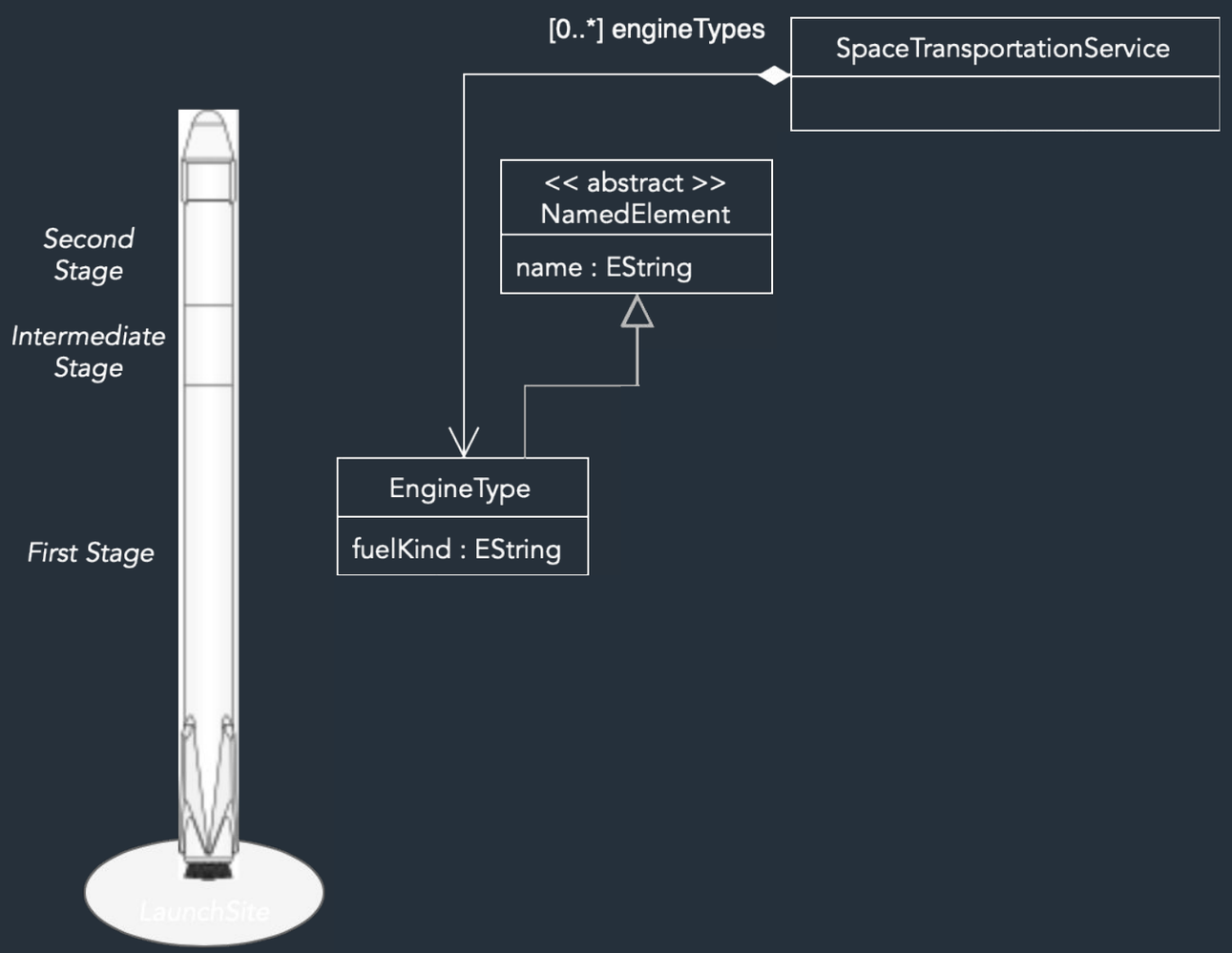
SpaceTransportationService (STS, root element)

SpaceTransportationService



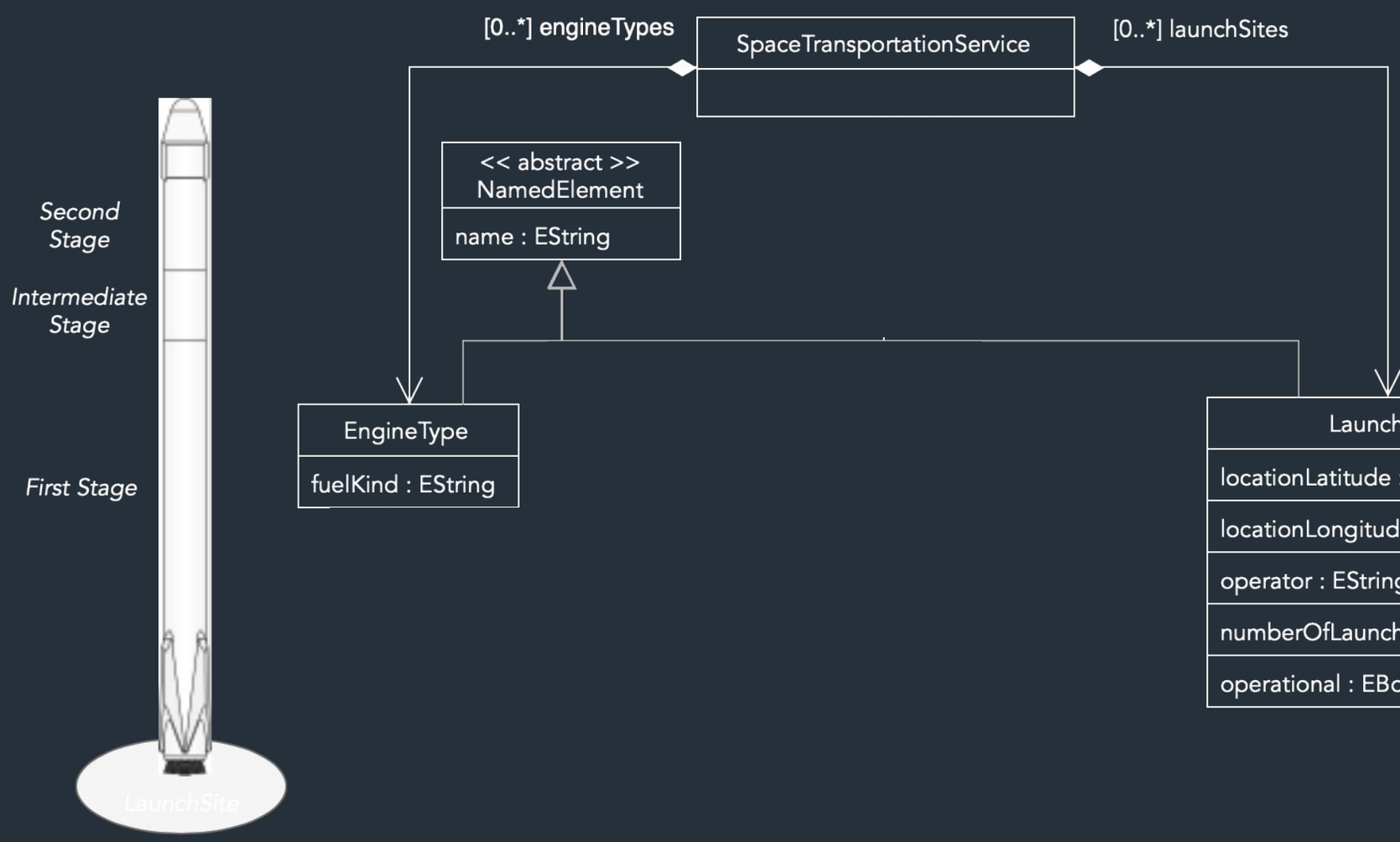
Example Scenario: Structure

STS *engineTypes*



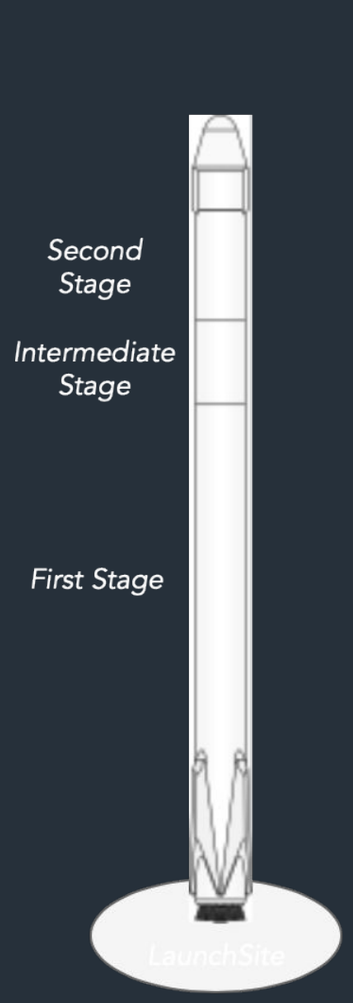
Example Scenario: Structure

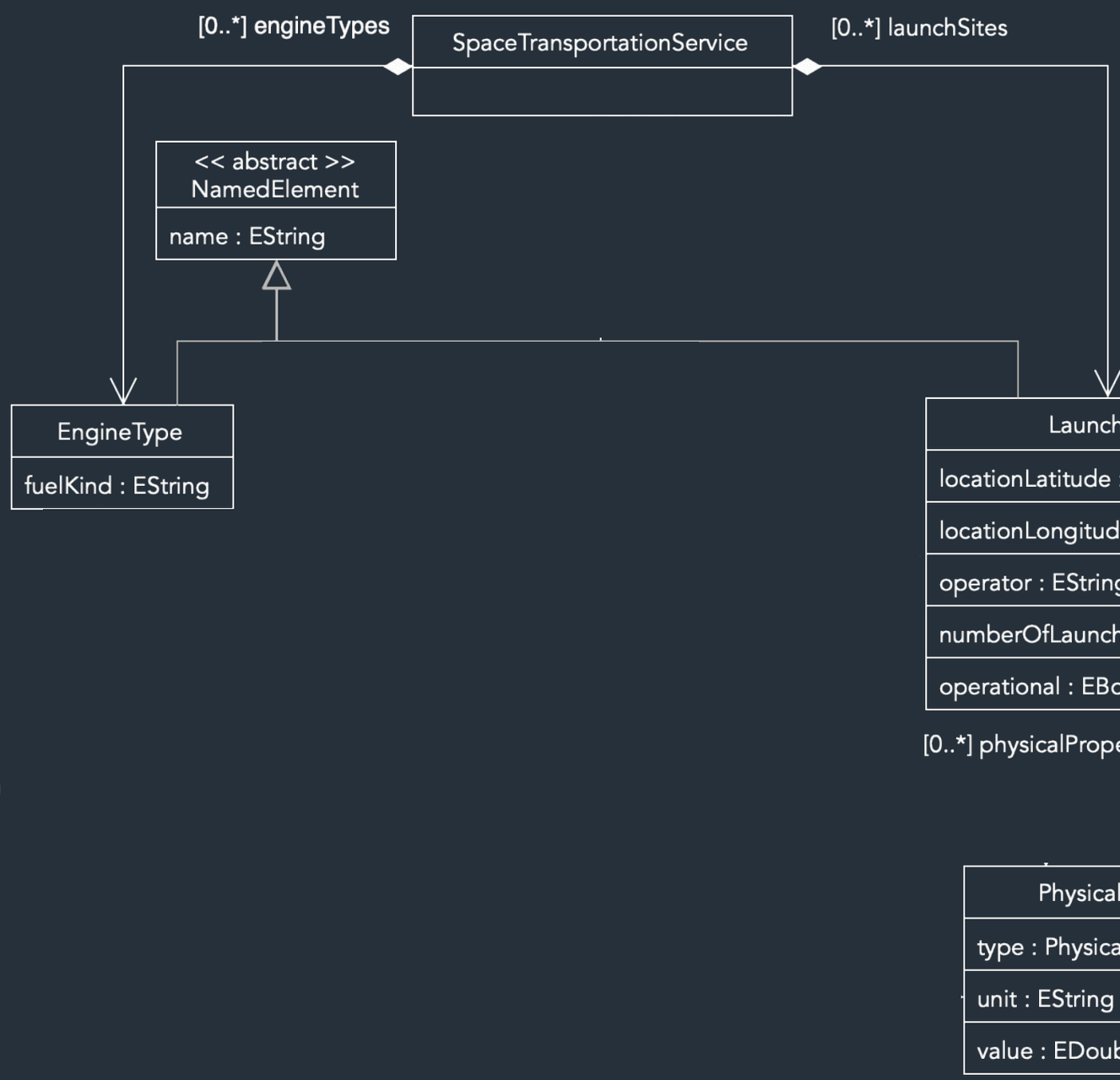
STS *launchSites*



Example Scenario: Structure

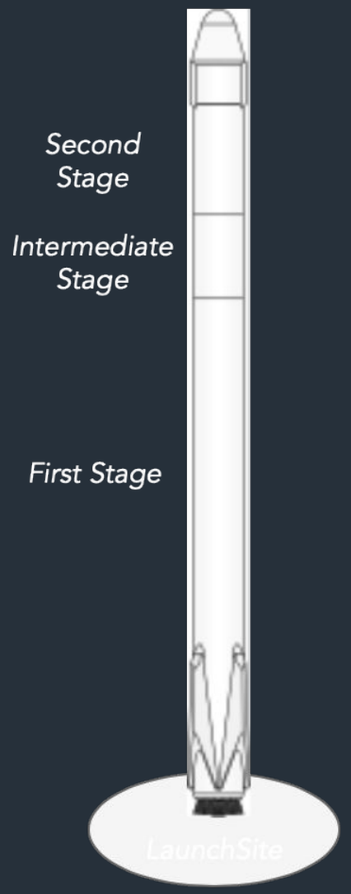
LaunchSite *physicalProperties*

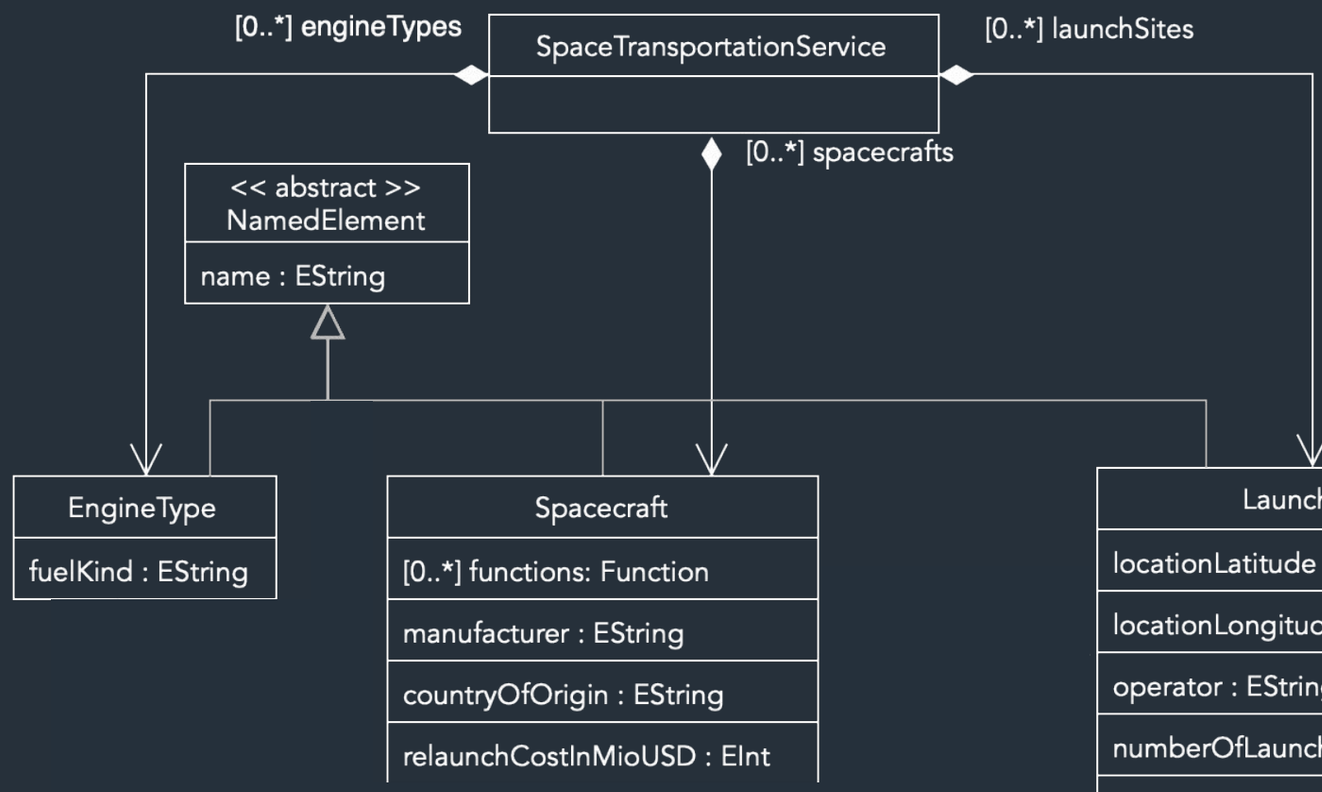




Example Scenario: Structure

STS *spacecrafts*



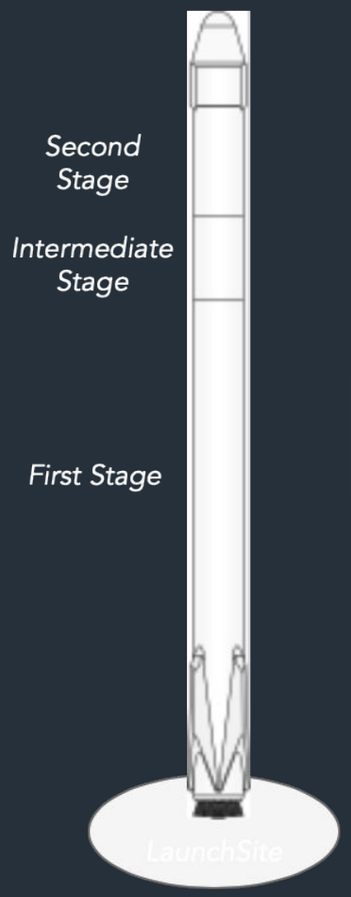


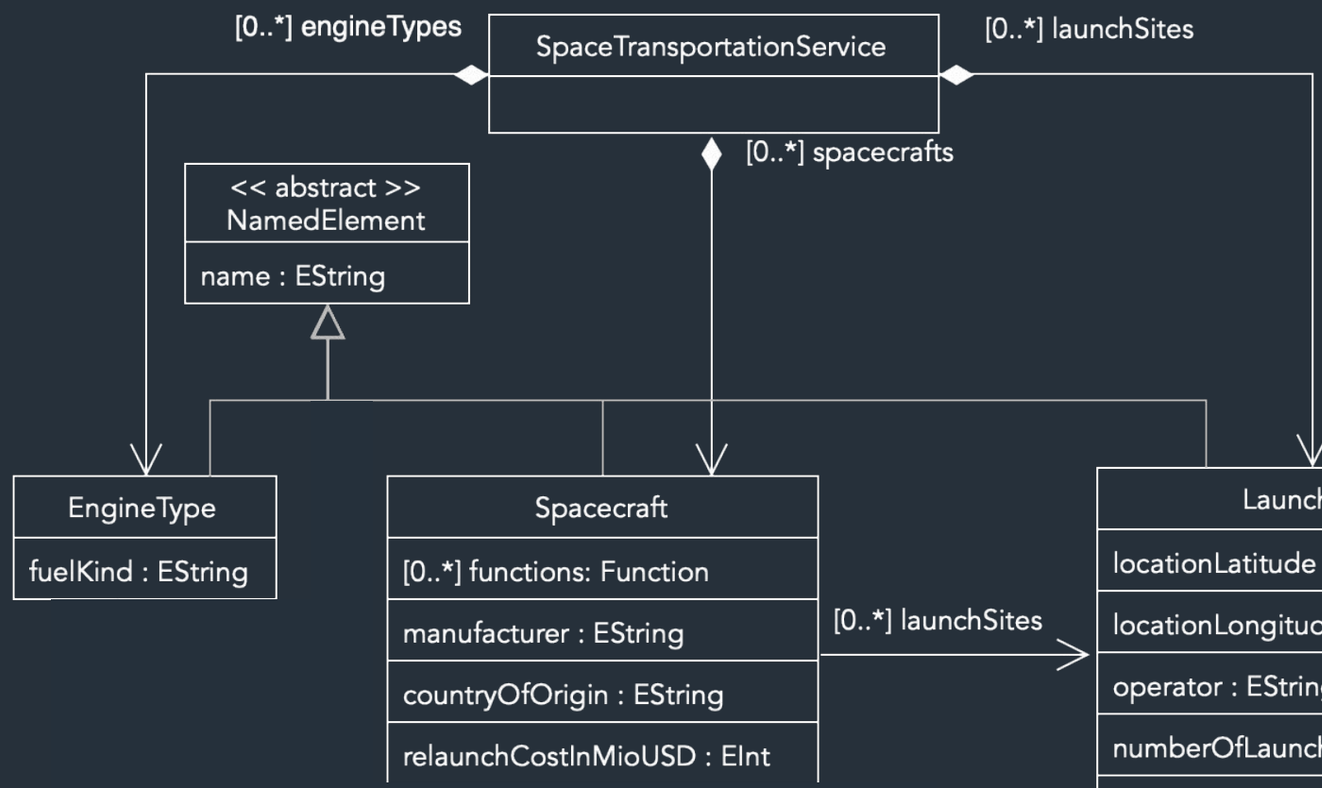
<< enumeration >> Function
MARS_COLONIZATION
EARTH_LUNAR_TRANSPORT
MULTIPLANETARY_TRANSPORT
INTERCONTINENTAL_TRANSPORT
ORBITAL_LAUNCHER

<< enumeration >> PhysicalPropertyType
LENGTH
WIDTH
PERIMETER
AREA
VOLUME
MASS

Example Scenario: Structure

Spacecraft → *launchSites*



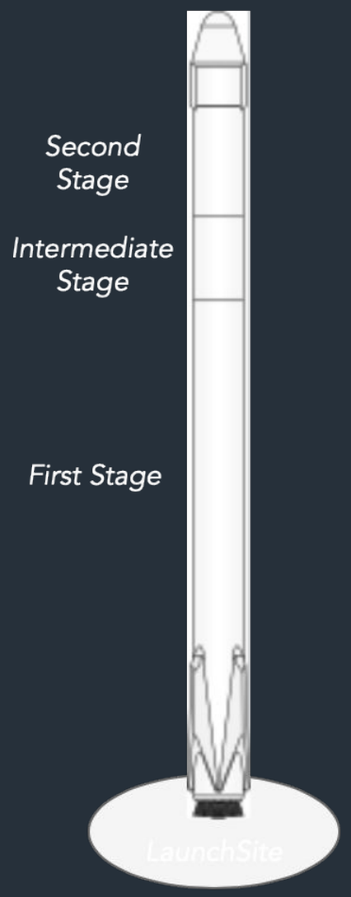


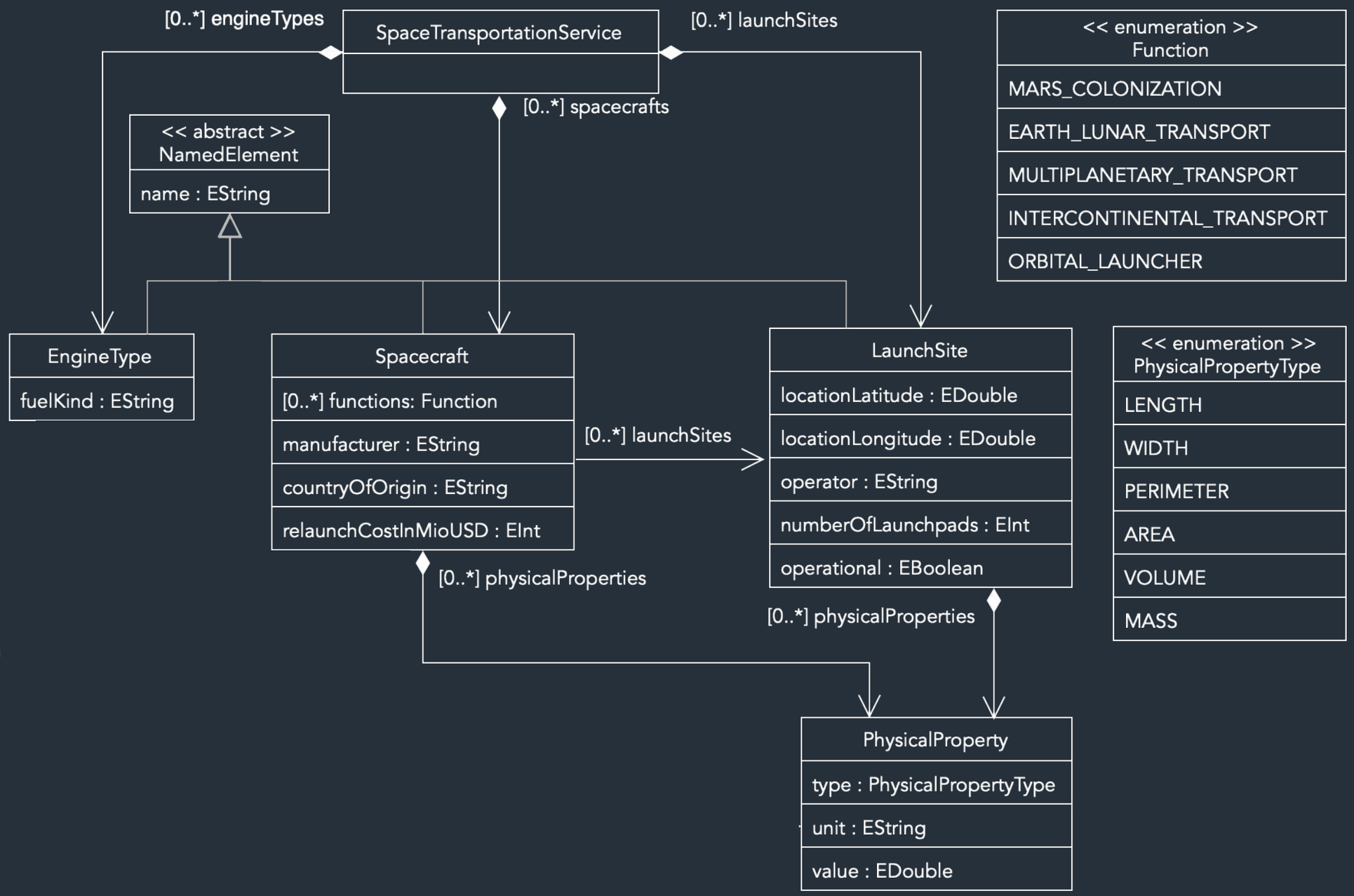
<< enumeration >> Function
MARS_COLONIZATION
EARTH_LUNAR_TRANSPORT
MULTIPLANETARY_TRANSPORT
INTERCONTINENTAL_TRANSPORT
ORBITAL_LAUNCHER

<< enumeration >> PhysicalPropertyType
LENGTH
WIDTH
PERIMETER
AREA
VOLUME
MASS

Example Scenario: Structure

Spacecraft *physicalProperties*



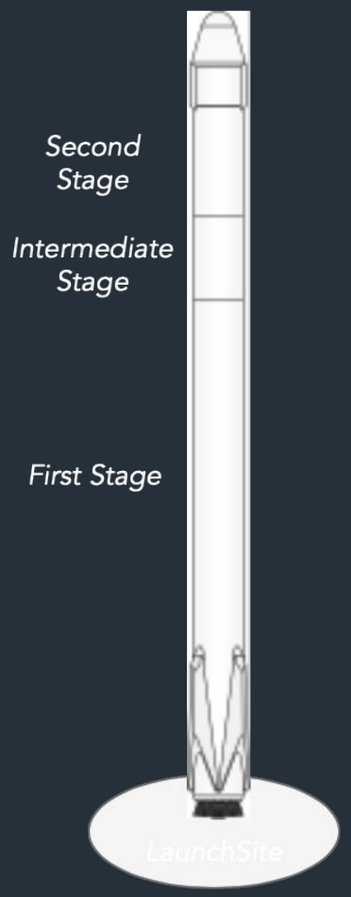


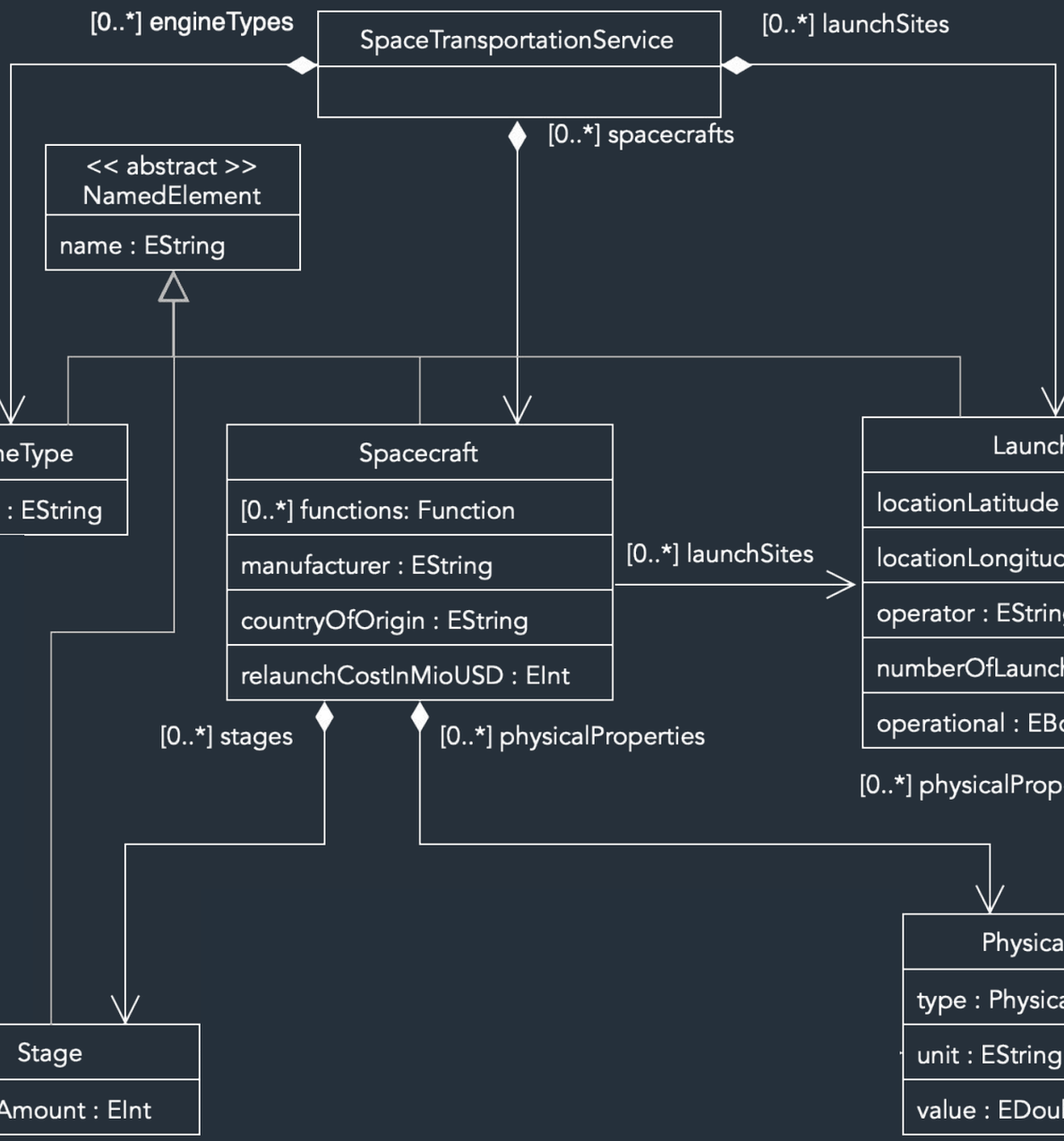
<< enumeration >> Function
MARS_COLONIZATION
EARTH_LUNAR_TRANSPORT
MULTIPLANETARY_TRANSPORT
INTERCONTINENTAL_TRANSPORT
ORBITAL_LAUNCHER

<< enumeration >> PhysicalPropertyType
LENGTH
WIDTH
PERIMETER
AREA
VOLUME
MASS

Example Scenario: Structure

Spacecraft *stages*



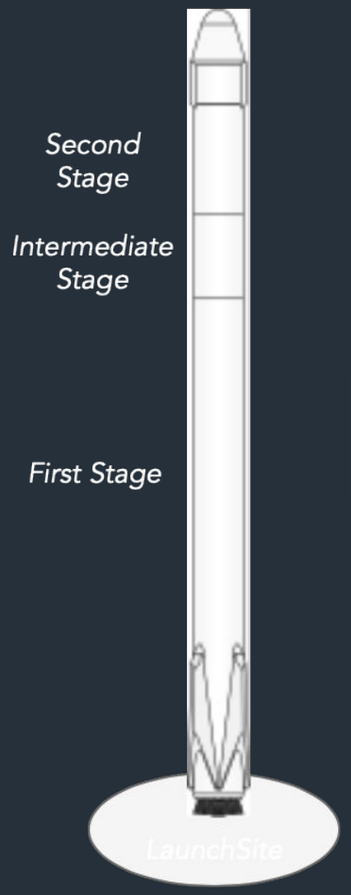


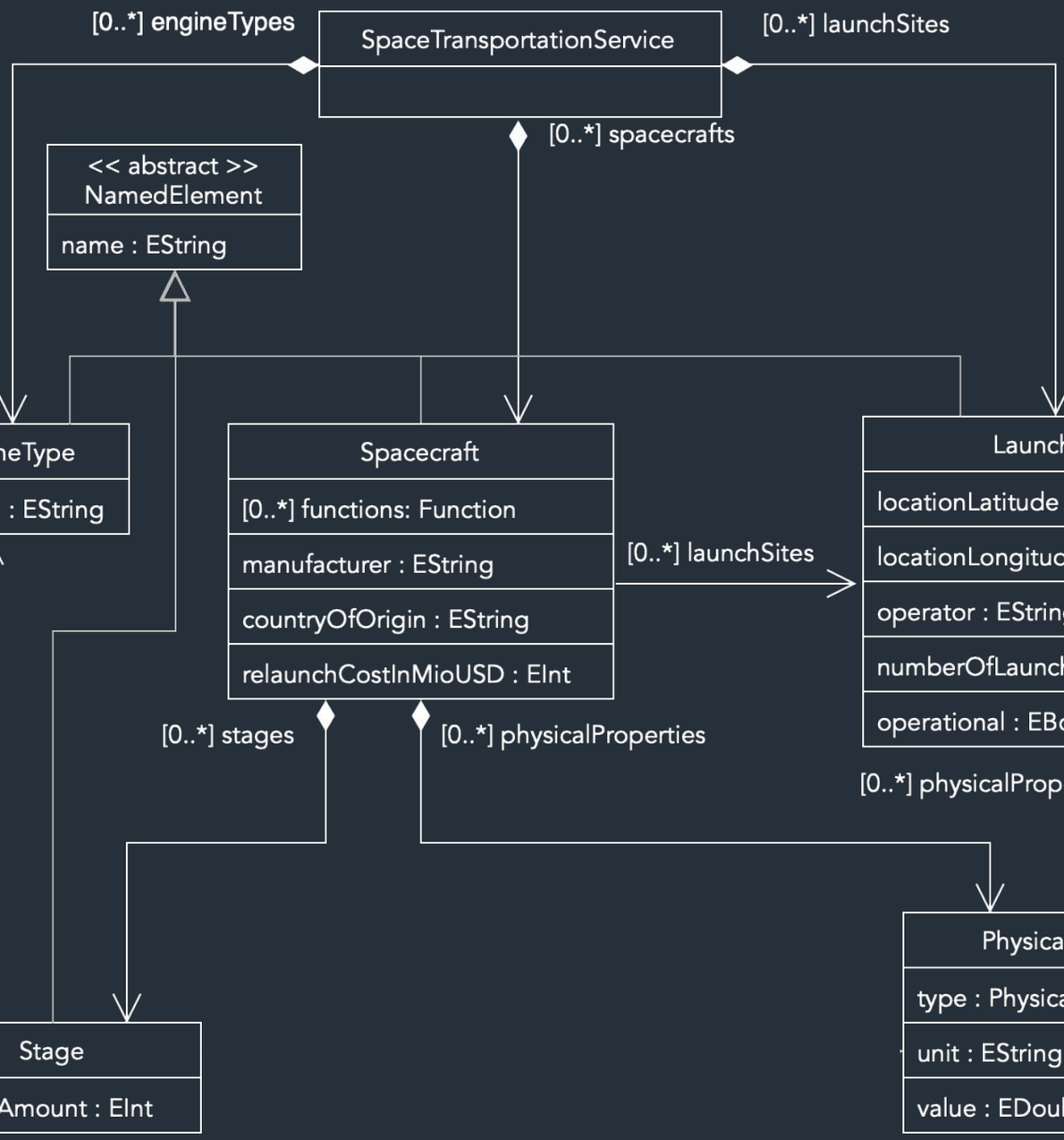
<< enumeration >> Function
MARS_COLONIZATION
EARTH_LUNAR_TRANSPORT
MULTIPLANETARY_TRANSPORT
INTERCONTINENTAL_TRANSPORT
ORBITAL_LAUNCHER

<< enumeration >> PhysicalPropertyType
LENGTH
WIDTH
PERIMETER
AREA
VOLUME
MASS

Example Scenario: Structure

Stage → *engineType*



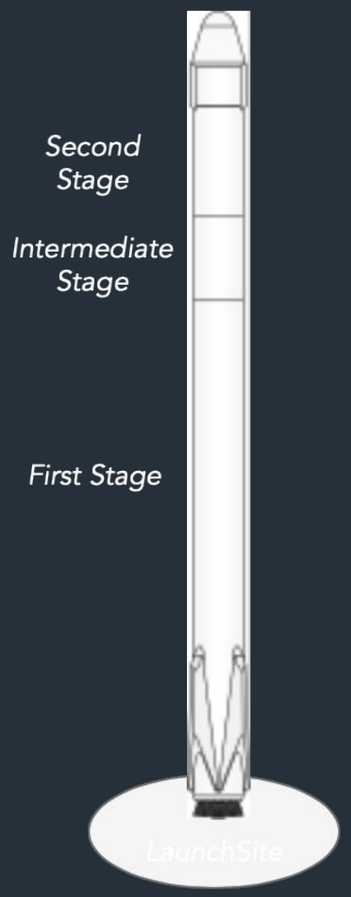


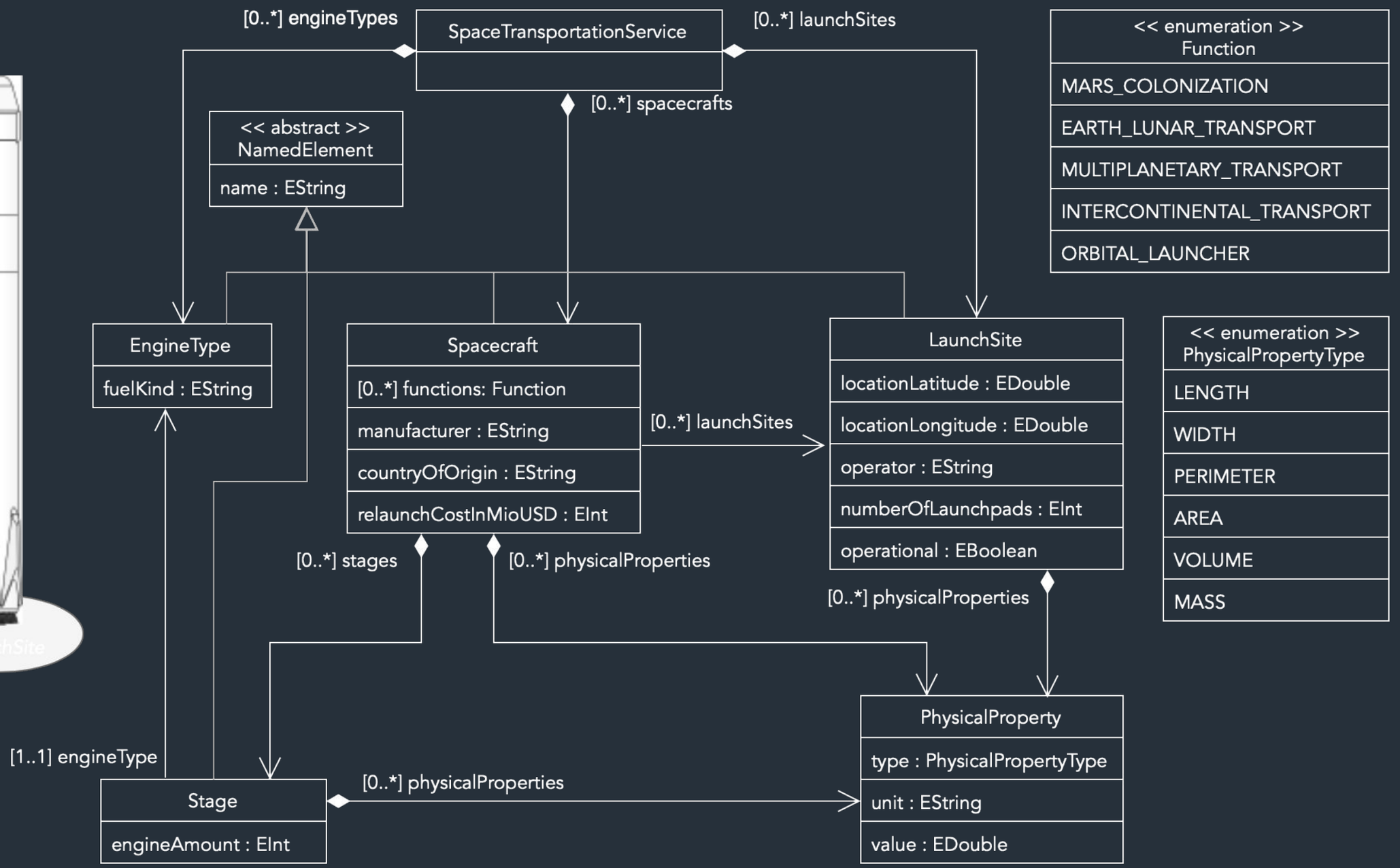
<< enumeration >> Function
MARS_COLONIZATION
EARTH_LUNAR_TRANSPORT
MULTIPLANETARY_TRANSPORT
INTERCONTINENTAL_TRANSPORT
ORBITAL_LAUNCHER

<< enumeration >> PhysicalPropertyType
LENGTH
WIDTH
PERIMETER
AREA
VOLUME
MASS

Example Scenario: Structure

Stage *physicalProperties*





<< enumeration >> Function
MARS_COLONIZATION
EARTH_LUNAR_TRANSPORT
MULTIPLANETARY_TRANSPORT
INTERCONTINENTAL_TRANSPORT
ORBITAL_LAUNCHER

<< enumeration >> PhysicalPropertyType
LENGTH
WIDTH
PERIMETER
AREA
VOLUME
MASS

Example Scenario: Style

Example Scenario: Style

Curly bracket-based layout

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Example Scenario: Style

Curly bracket-based layout

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Default layout

Example Scenario: Style

Curly bracket-based layout

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Single order of declaration

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Example Scenario: Style

Curly bracket-based layout

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Single order of declaration

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Default order of declaration

Example Scenario: Style

Default Grammar (excerpt)

```
grammar spacetransportationservice.MyDefaultSts  with org.eclipse.xtext.common.Terminals

import "http://cs.york.ac.uk/ecss/examples/spacetransportationservice"
import "http://www.eclipse.org/emf/2002/Ecore"  as ecore

...

LaunchSite returns LaunchSite:
    operational?='operational'
    'LaunchSite'
    name=EString
    '{'
        'locationLatitude' locationLatitude=EDouble
        'locationLongitude' locationLongitude=EDouble
        ('operator' operator=EString)?
        'numberOfLaunchpads' numberOfLaunchpads=EInt
        ('physicalProperties' '{' physicalProperties+=PhysicalProperty (  "," 
physicalProperties+=PhysicalProperty)*  '}' )?
    '}';

...
```

Single order of declaration

Curly bracket-based layout

Example Scenario: Style

Curly bracket-based layout (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Indentation-based layout

Single order of declaration (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Example Scenario: Style

Curly bracket-based layout (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Indentation-based layout

```
SpaceTransportationService
  launchSites
    operational LaunchSite KennedySpaceCenter
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    ...
  ...
```

Single order of declaration (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Example Scenario: Style

Curly bracket-based layout (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Indentation-based layout

```
SpaceTransportationService
  launchSites
    operational LaunchSite KennedySpaceCenter
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    ...
  ...
```

Single order of declaration (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Arbitrary order of declaration

Example Scenario: Style

Curly bracket-based layout (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Indentation-based layout

```
SpaceTransportationService
  launchSites
    operational LaunchSite KennedySpaceCenter
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    ...
  ...
```

Single order of declaration (default)

```
SpaceTransportationService {
  launchSites {
    operational LaunchSite KennedySpaceCenter {
      locationLatitude 28.524058
      locationLongitude -80.65085
      operator NASA
      numberOfLaunchpads 3
    }
    ...
  }
  ...
}
```

Arbitrary order of declaration

```
SpaceTransportationService
  ...
  launchSites
    operational LaunchSite KennedySpaceCenter
      operator NASA
      numberOfLaunchpads 3
      locationLatitude 28.524058
      locationLongitude -80.65085
    ...
```

Example Scenario: Style

```
SpaceTransportationService
  ...
  launchSites
    operational  LaunchSite KennedySpaceCenter
      operator NASA
      numberOfLaunchpads 3
      locationLatitude 28.524058
      locationLongitude -80.65085
...
```

Indentation-based layout

Arbitrary order of declaration

Example Scenario: Style

```
SpaceTransportationService
  ...
  launchSites
    operational  LaunchSite KennedySpaceCenter
      operator NASA,
      numberOfLaunchpads 3,
      locationLatitude 28.524058,
      locationLongitude -80.65085
...
```

Indentation-based layout

Arbitrary order of declaration

Features separated by comma

Example Scenario: Style

```
SpaceTransportationService
  ...
  launchSites
    operational  LaunchSite KennedySpaceCenter
      operator NASA,
      numberOfLaunchpads 3,
      locationLatitude 28.524058,
      locationLongitude -80.65085
...
```

Indentation-based layout

Arbitrary order of declaration

Features separated by comma

Metamodel-agnostic

Example Scenario: Style

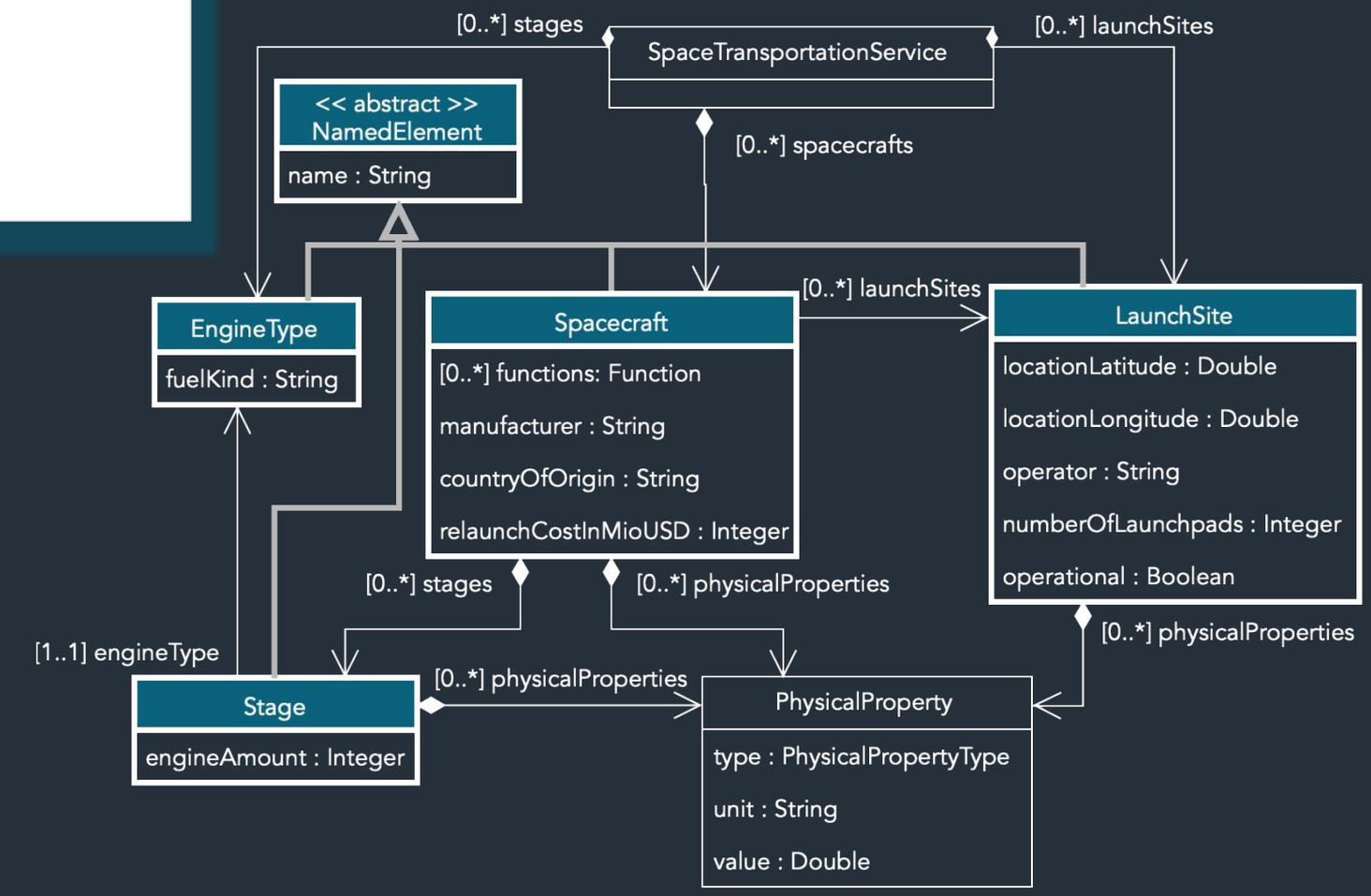
```
SpaceTransportationService
  ...
  launchSites
    operational LAUNCHSITE KennedySpaceCenter
      operator NASA,
      numberOfLaunchpads 3,
      locationLatitude 28.524058,
      locationLongitude -80.65085
    ...
```

Indentation-based layout

Arbitrary order of declaration

Features separated by comma

Keyword of classes that extend class ‘NamedElement’ in uppercase



Example Scenario: Style

```
SpaceTransportationService
  ...
  launchSites
    operational  LAUNCHSITE KennedySpaceCenter
      operator NASA,
      numberOfLaunchpads 3,
      locationLatitude 28.524058,
      locationLongitude -80.65085
...
```

Indentation-based layout

Arbitrary order of declaration

Features separated by comma

Keyword of classes that extend class ‘NamedElement’ in uppercase

Metamodel-dependent

Ecore Concrete Syntax Specification (ECSS)

Design principles and goals

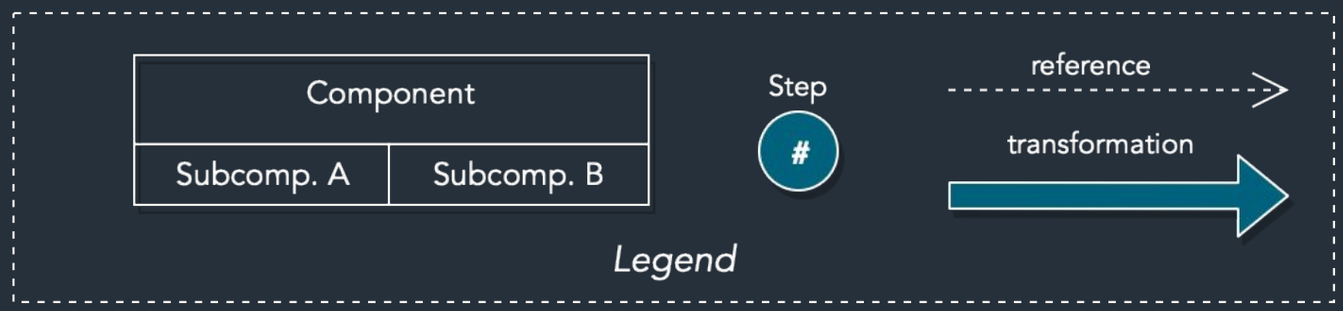
- Language aims to enable expression of common textual representations
  - For example: YAML, JSON, and XML

- Inspired by Cascading Style Sheets (CSS)
  - Composition (enable reuse and extension) and ability of parameterization

- Framework targets the ability to generate *styled* grammar for combinations of metamodel and style model
  - Based on grammar rule templates and injection-based property selection

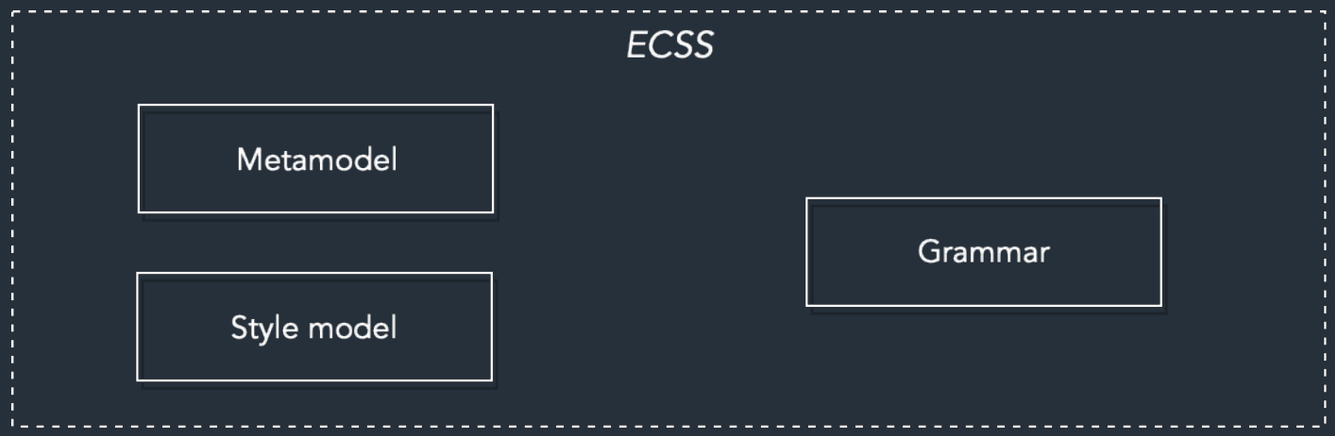
ECSS Approach

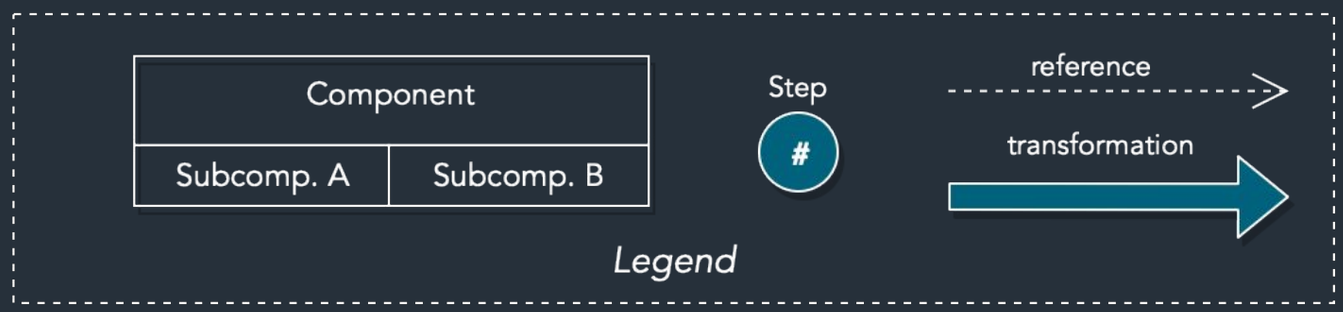
How to construct DSMLs by applying *ECSS* approach?



ECSS Approach

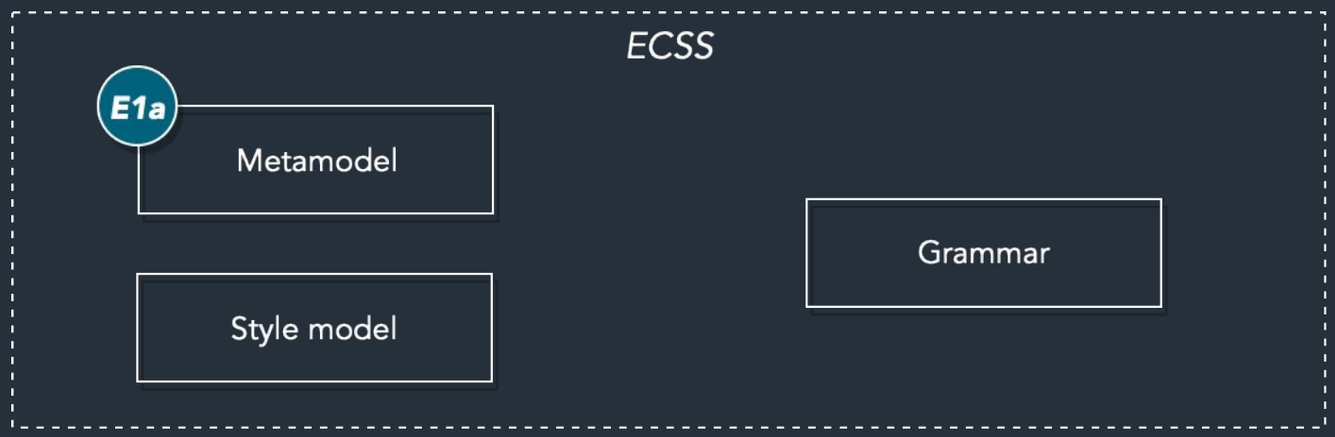
Based on model-first approach

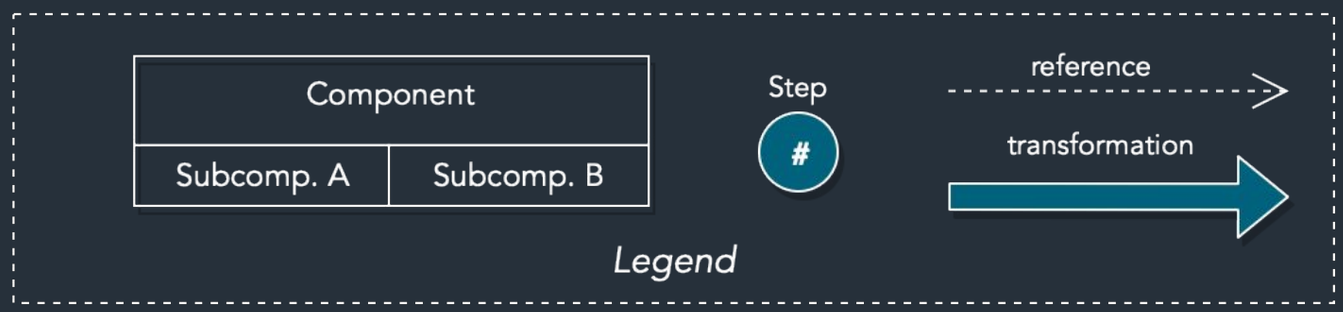




ECSS Approach

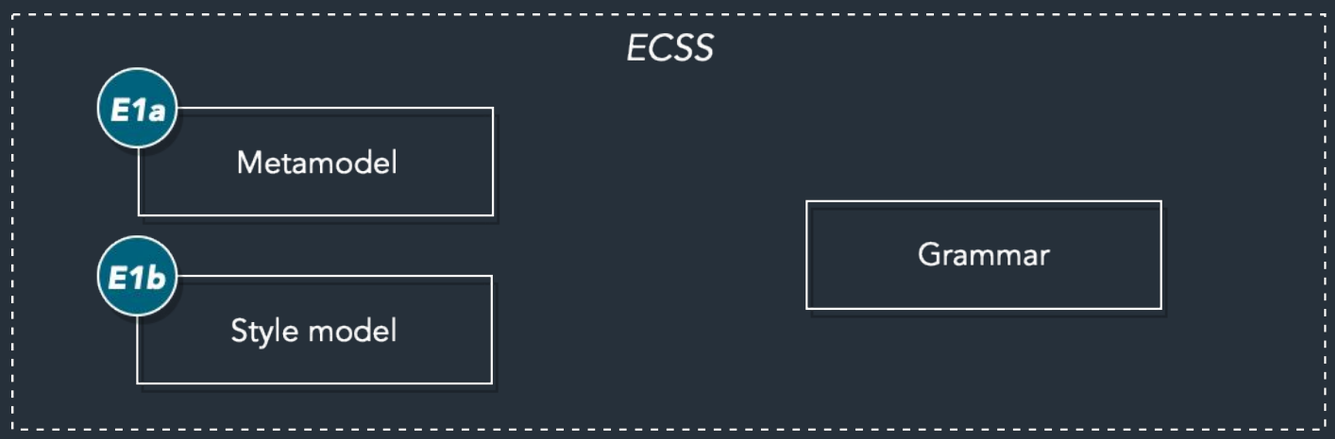
Create metamodel

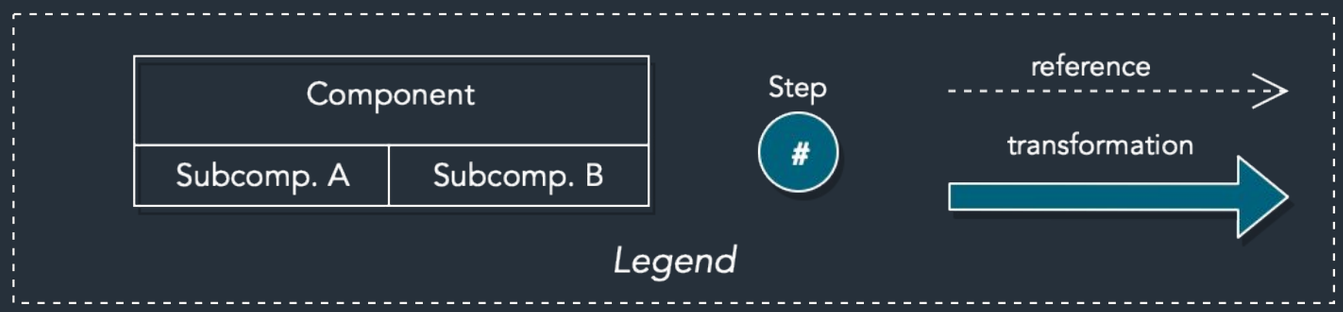




ECSS Approach

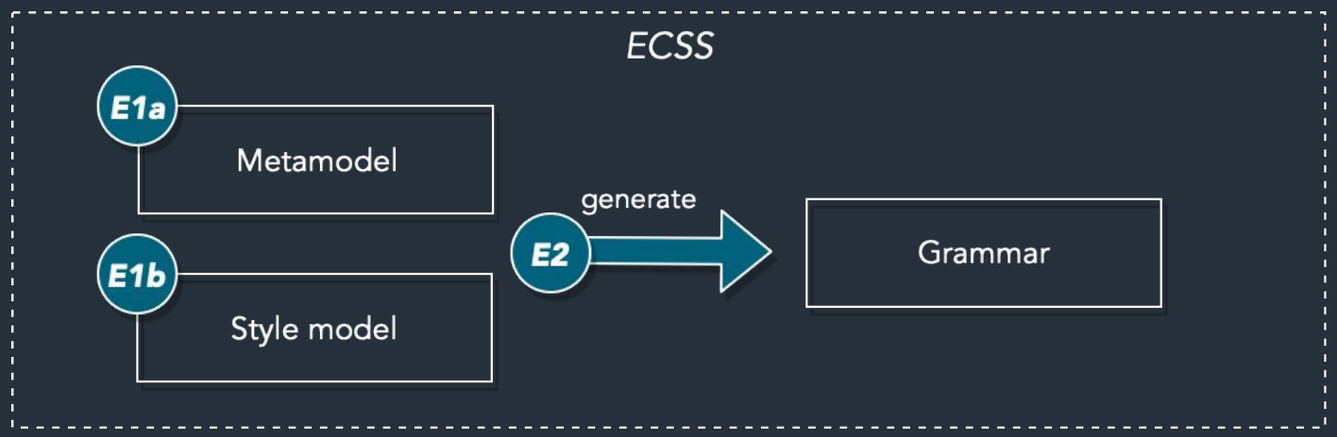
Create style model

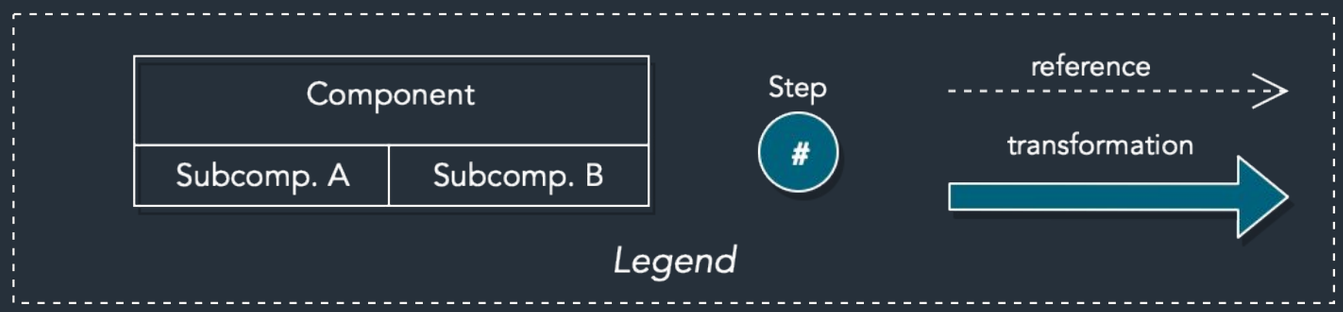




ECSS Approach

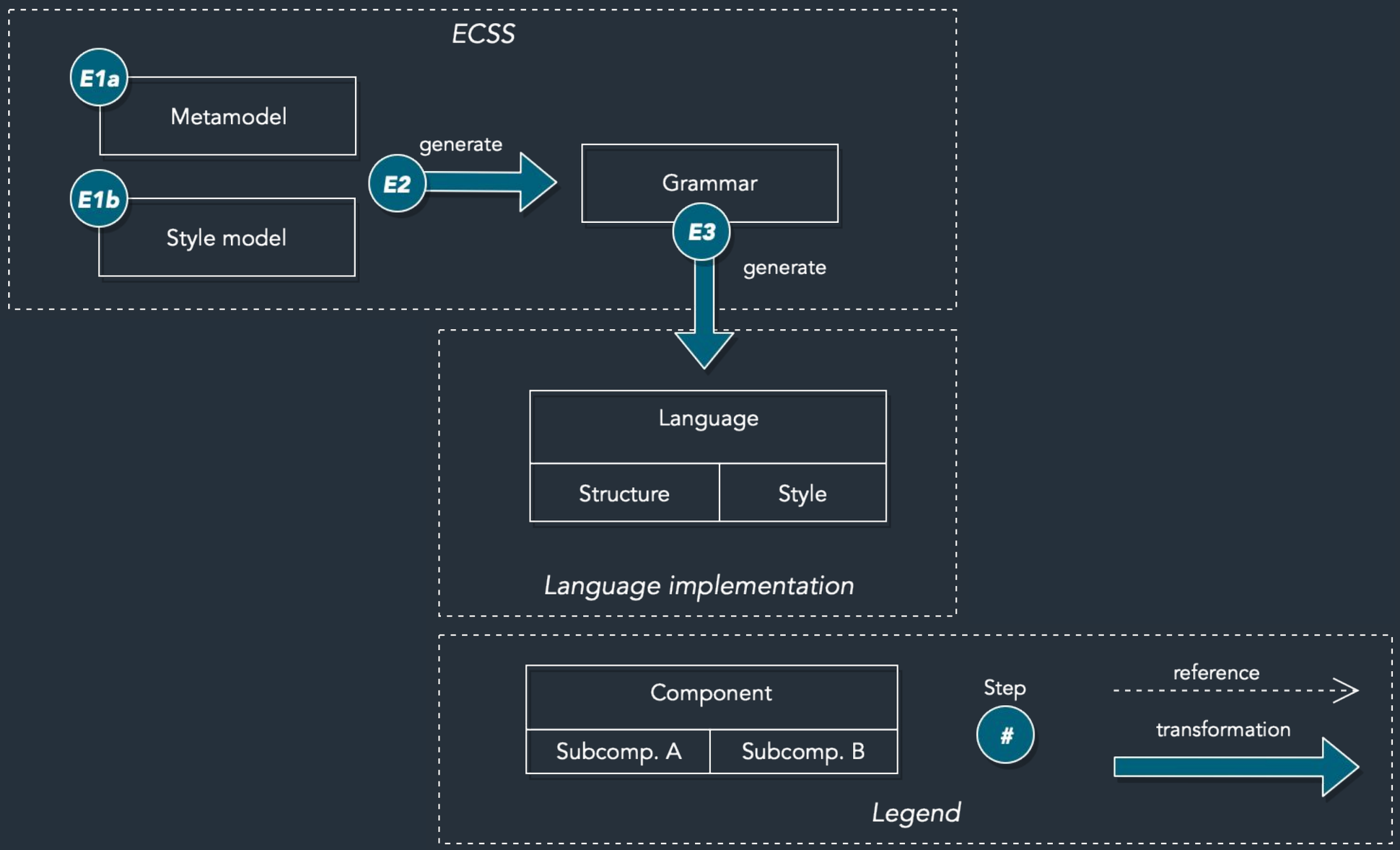
Generate *styled* grammar





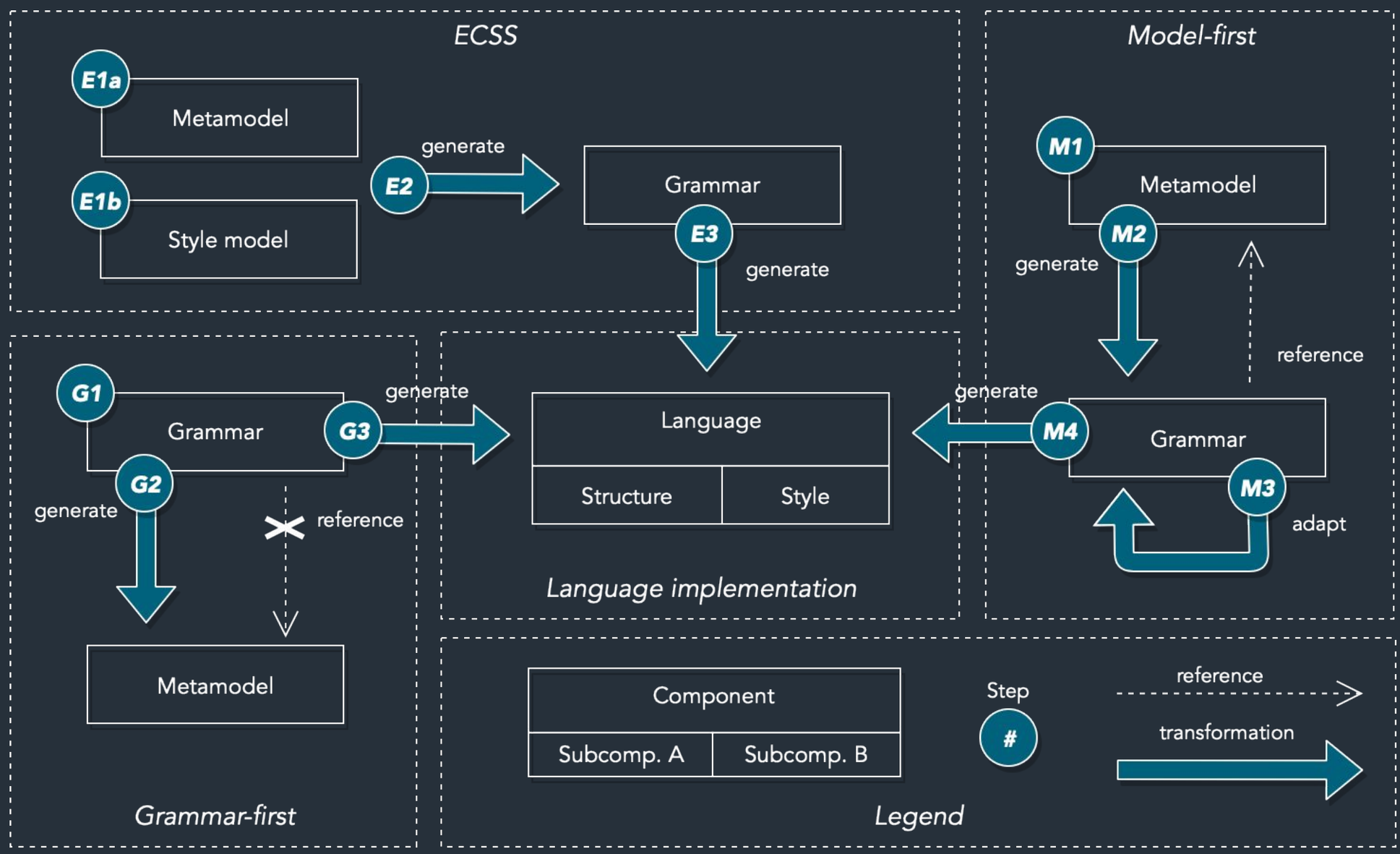
ECSS Approach

Generate language implementation from *styled* grammar



Summary of Approaches

- Grammar-first, Model-first, and ECSS



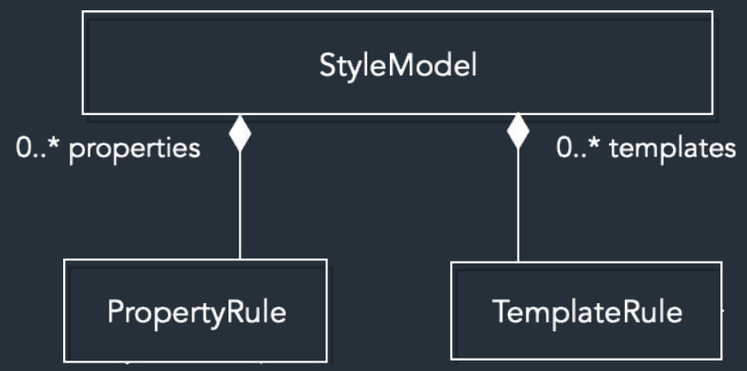
ECSS Language

StyleModel (root)

StyleModel

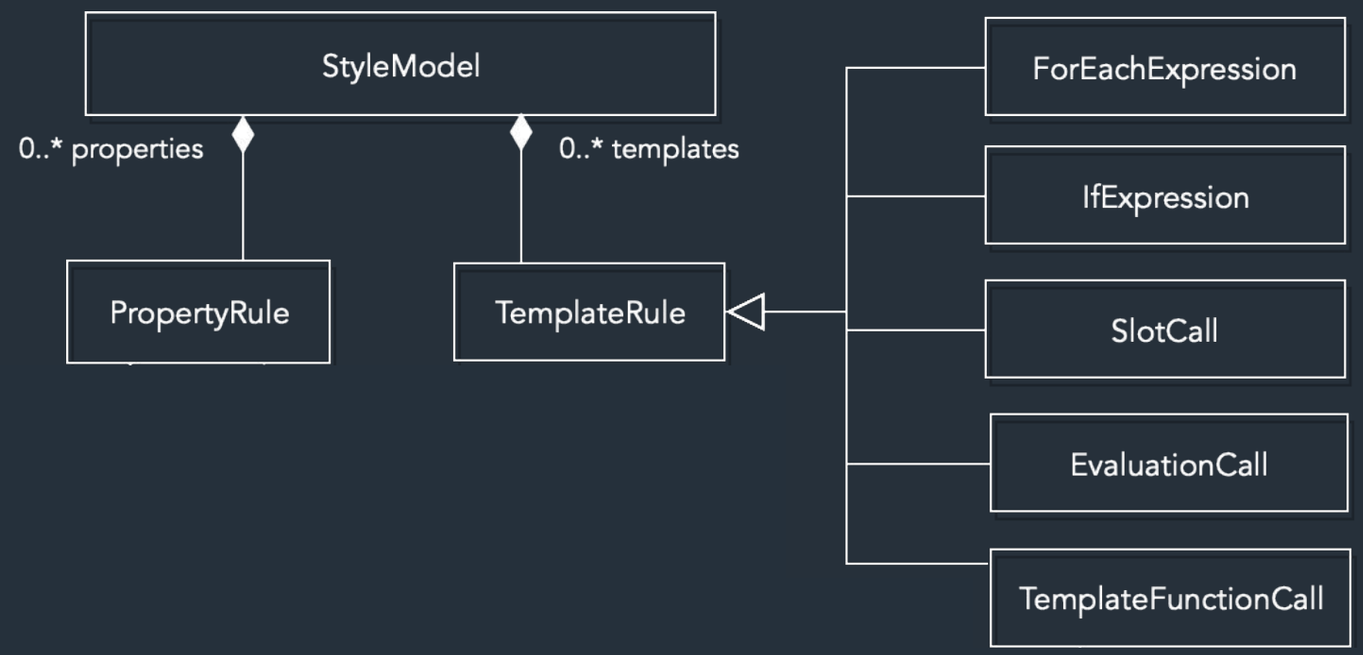
ECSS Language

StyleModel *properties* and *templates*



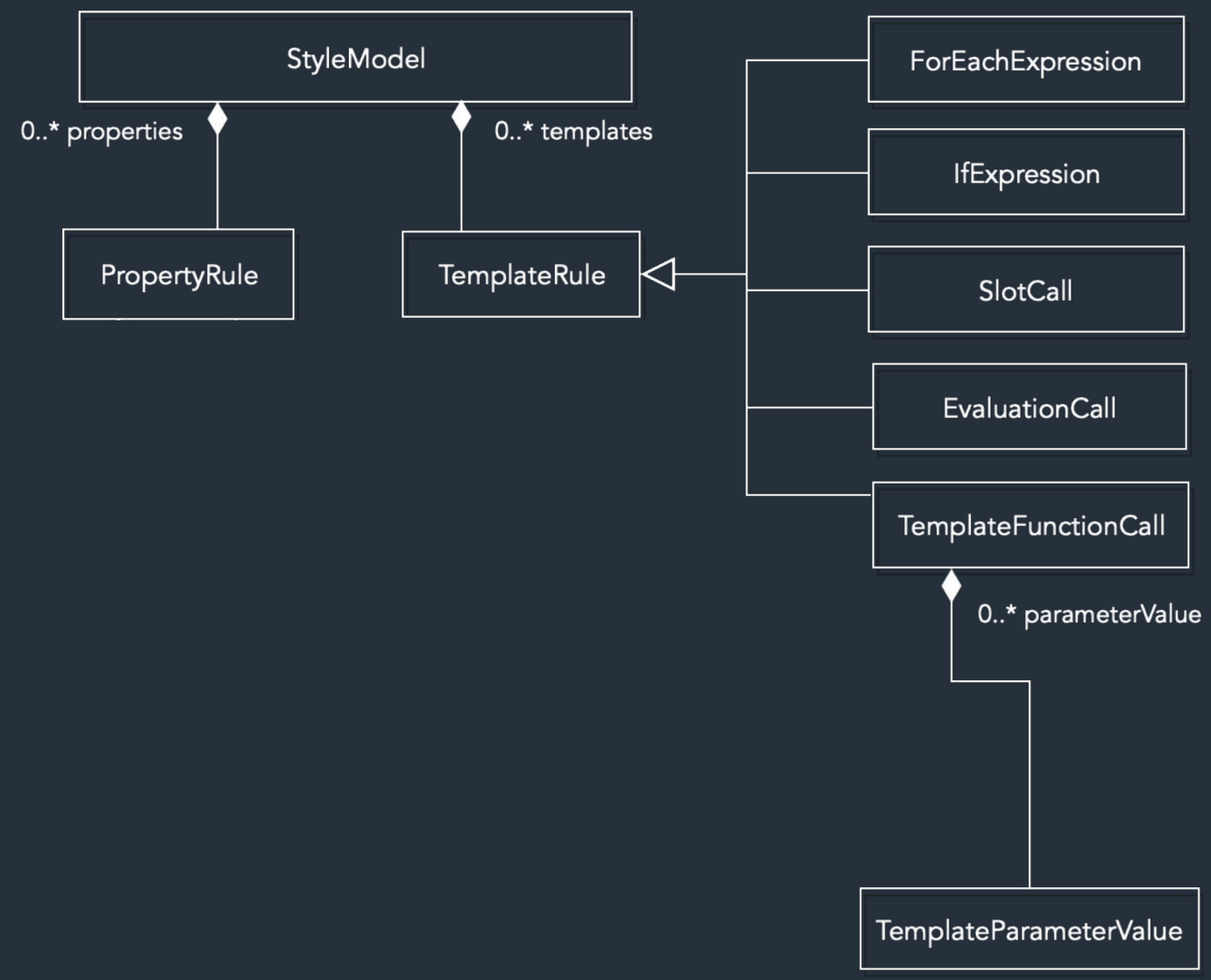
ECSS Language

TemplateRule *manifestations*



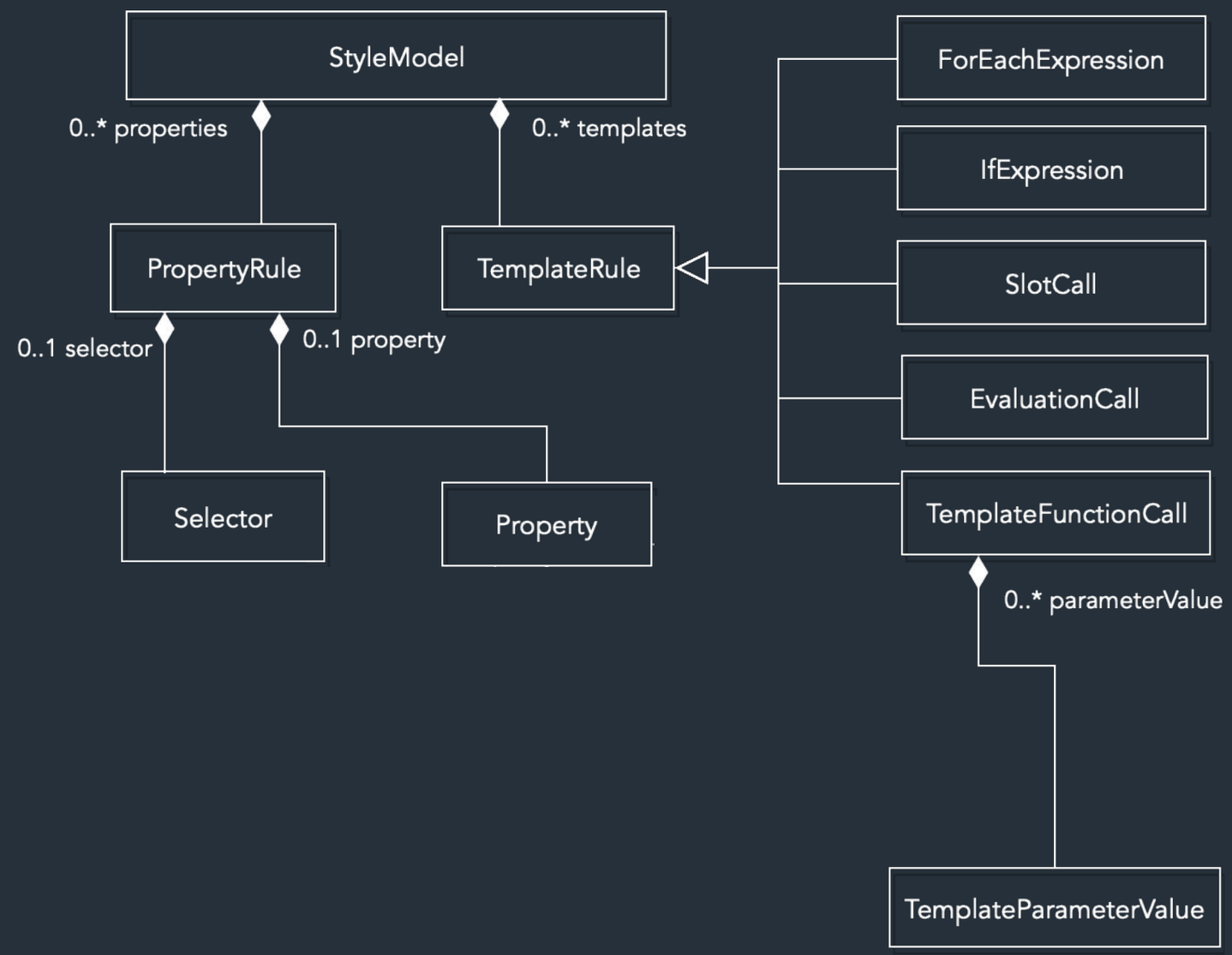
ECSS Language

TemplateFunctionCall *parameterValue*



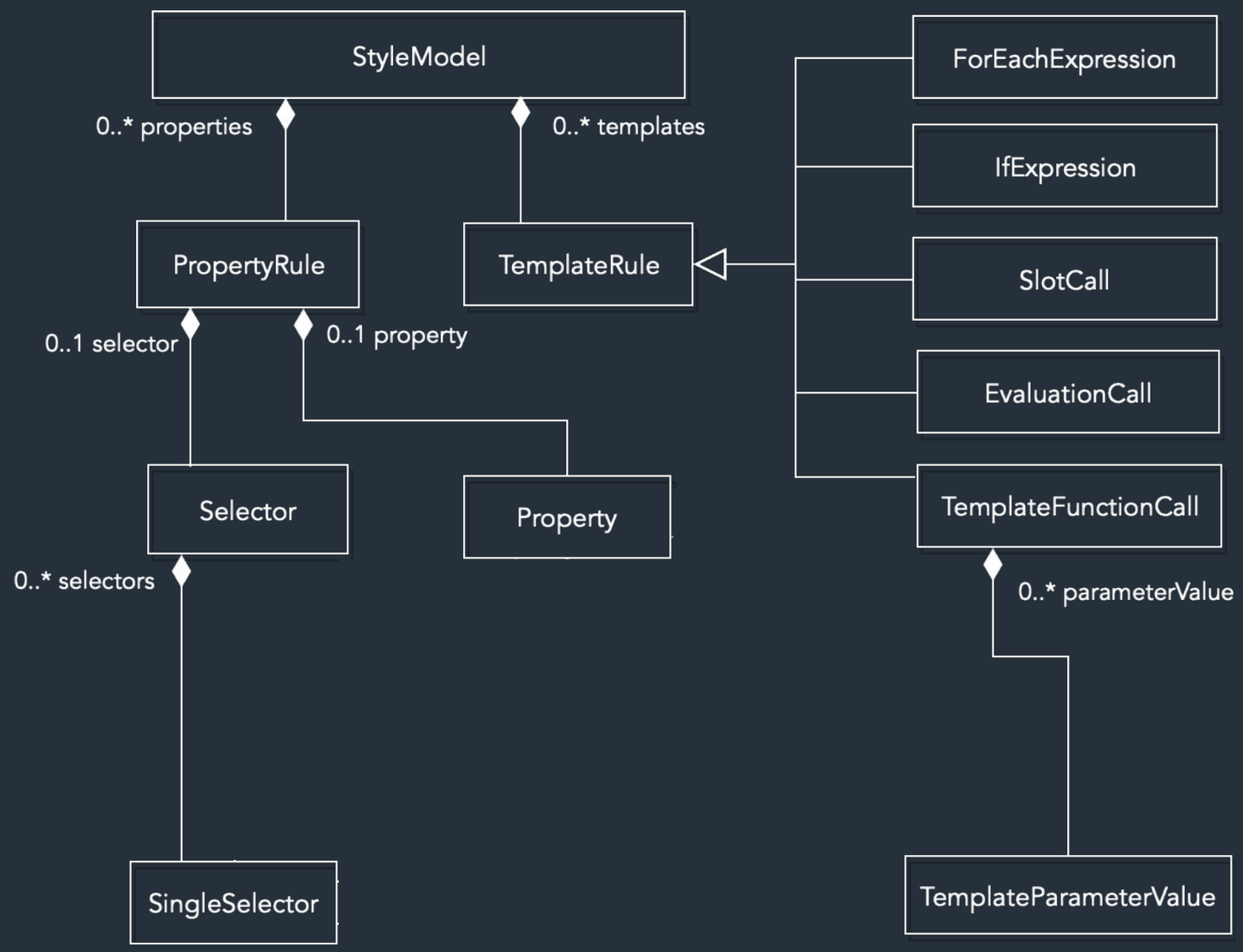
ECSS Language

PropertyRule *selector* and *property*



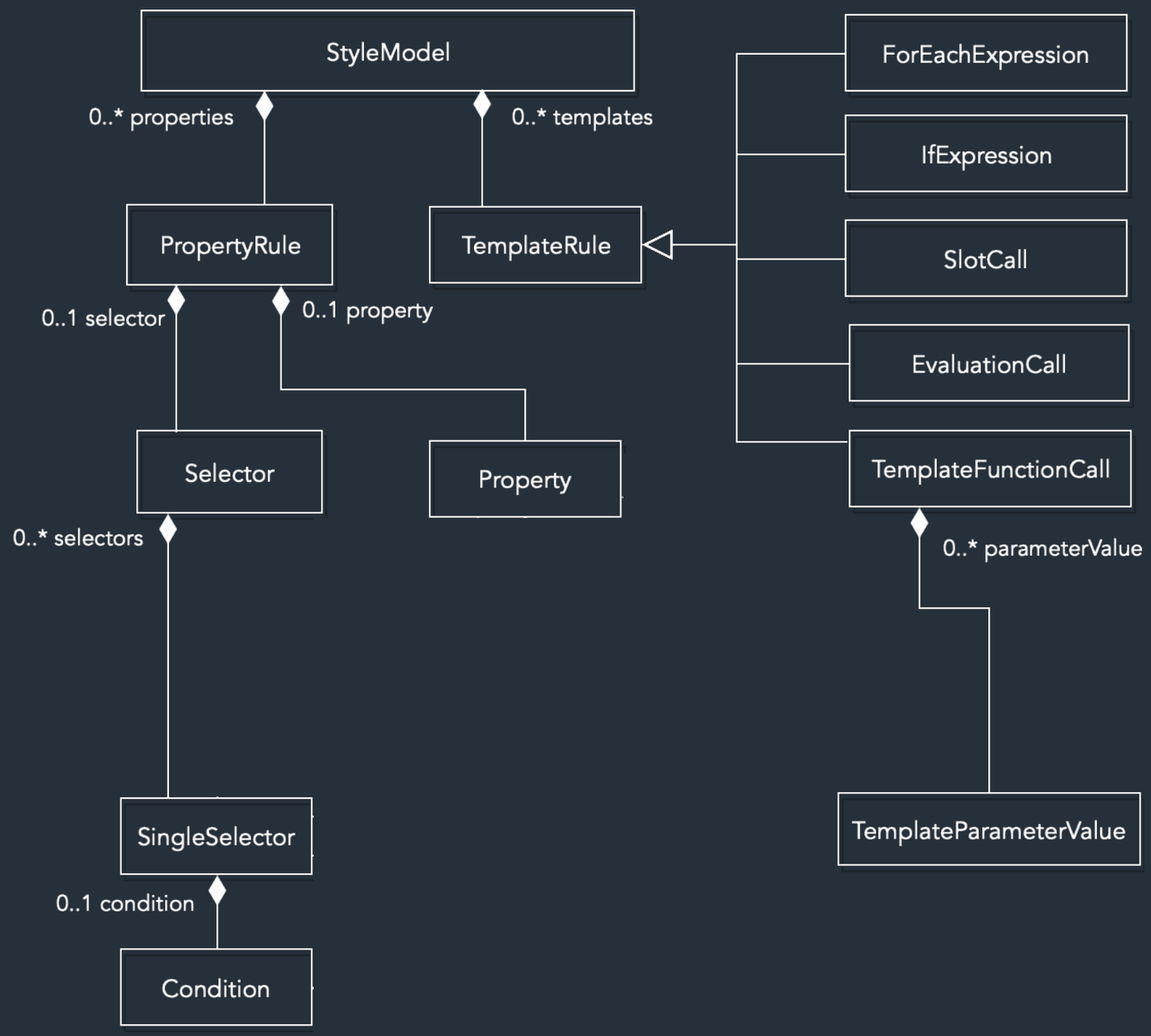
ECSS Language

Selector *selectors* (SingleSelector)



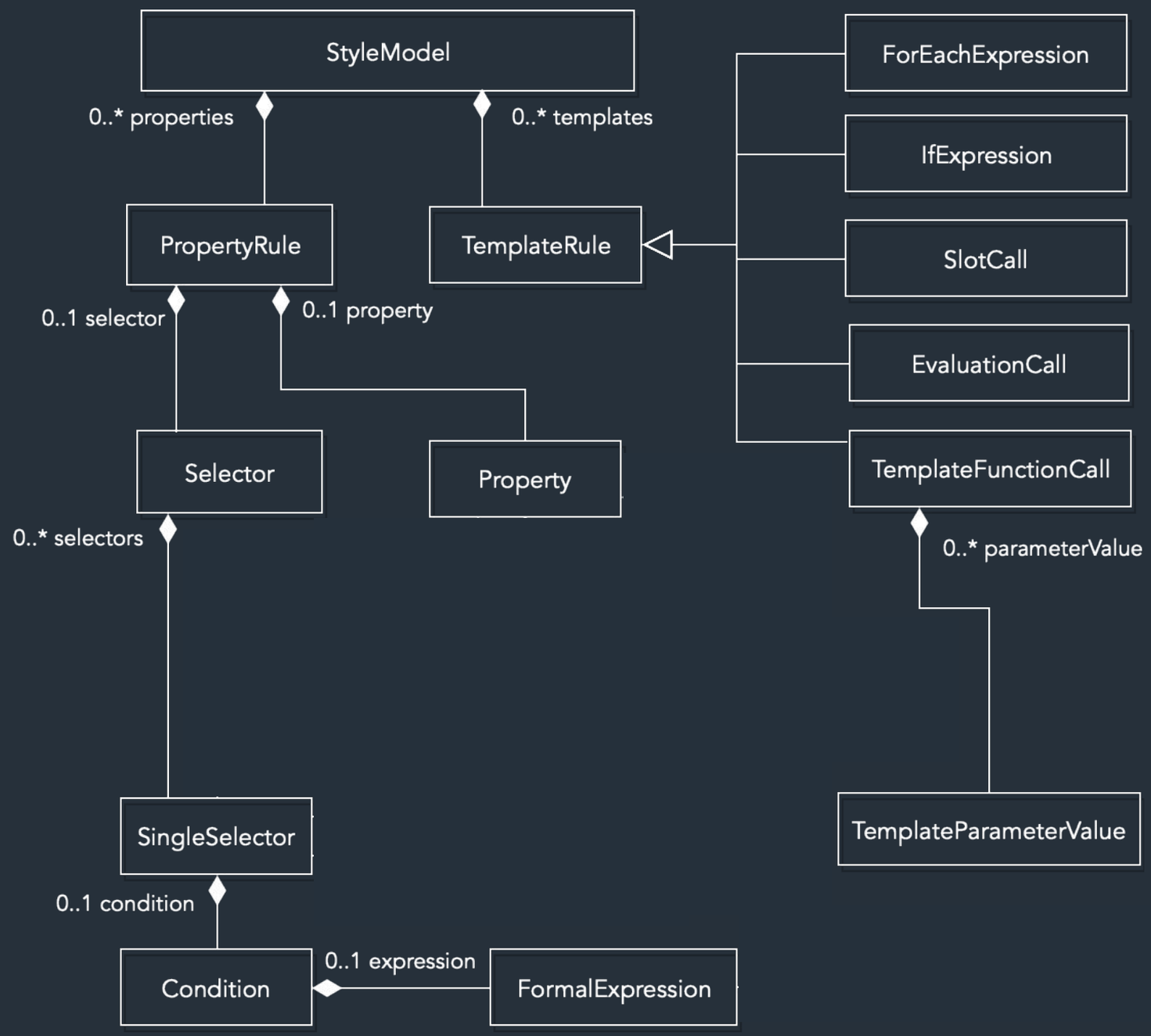
ECSS Language

SingleSelector *condition*



ECSS Language

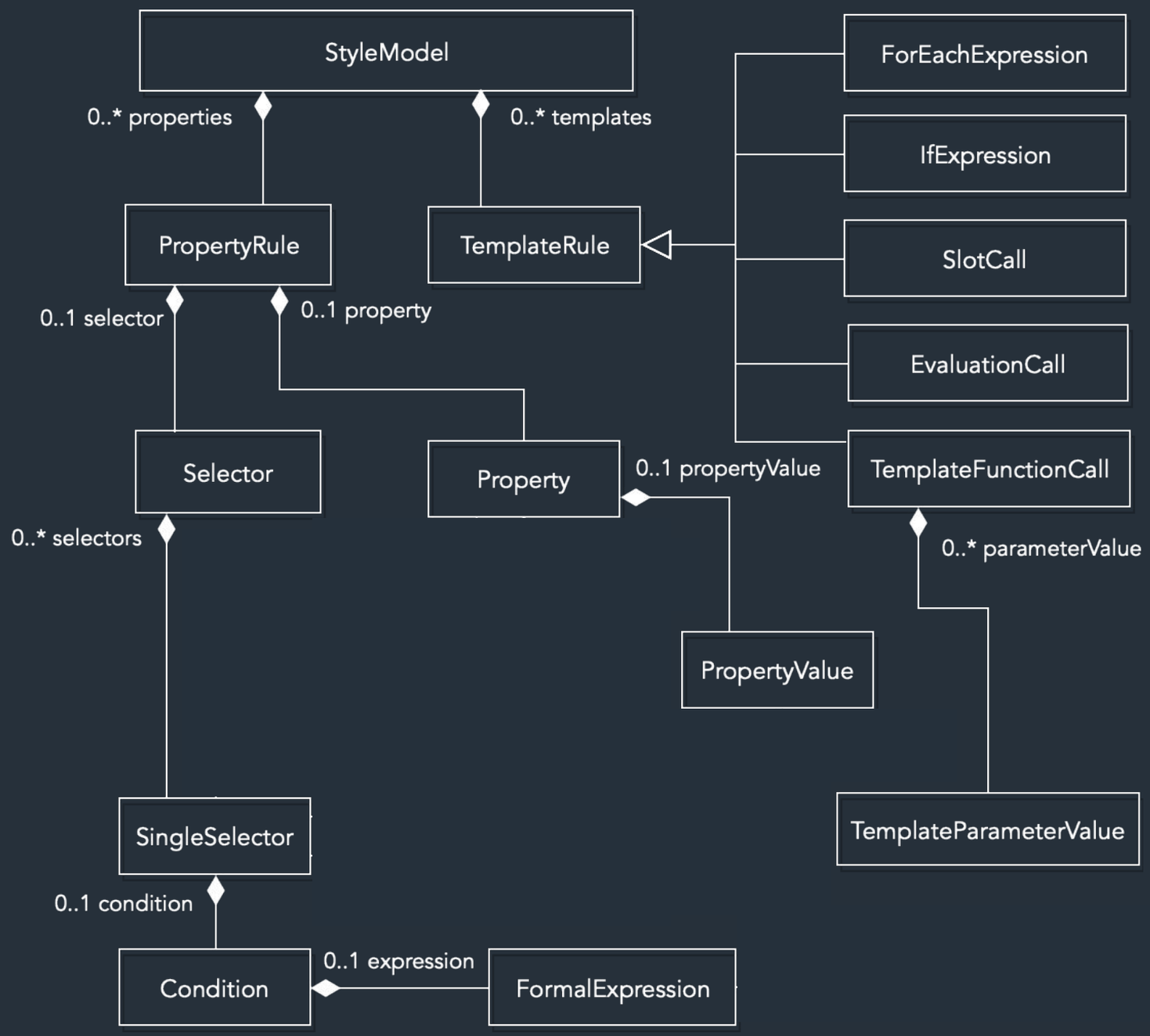
Condition *expression* (FormalExpression)



ECSS Language

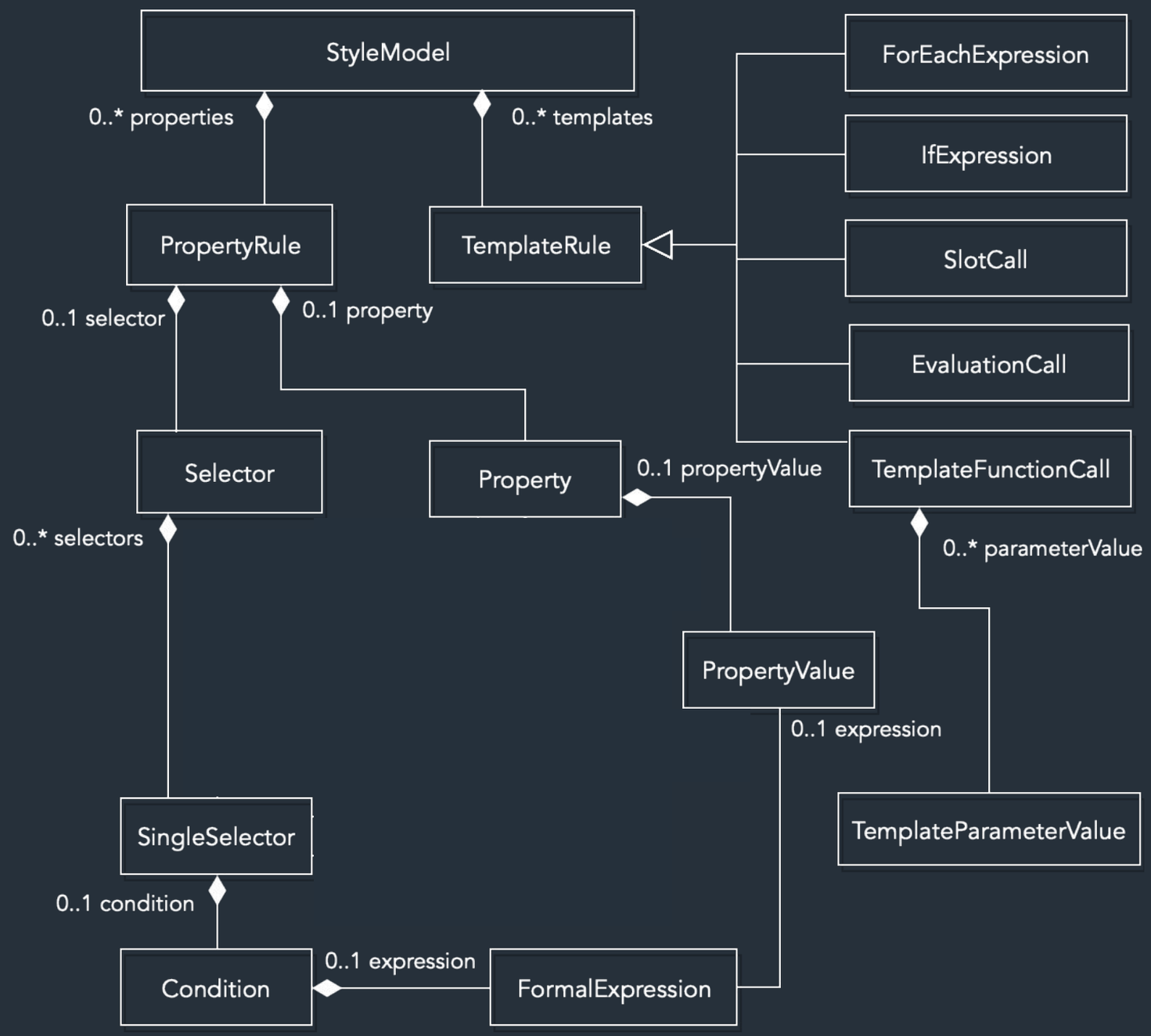
UNIVERSITY of York

Property *propertyValue*



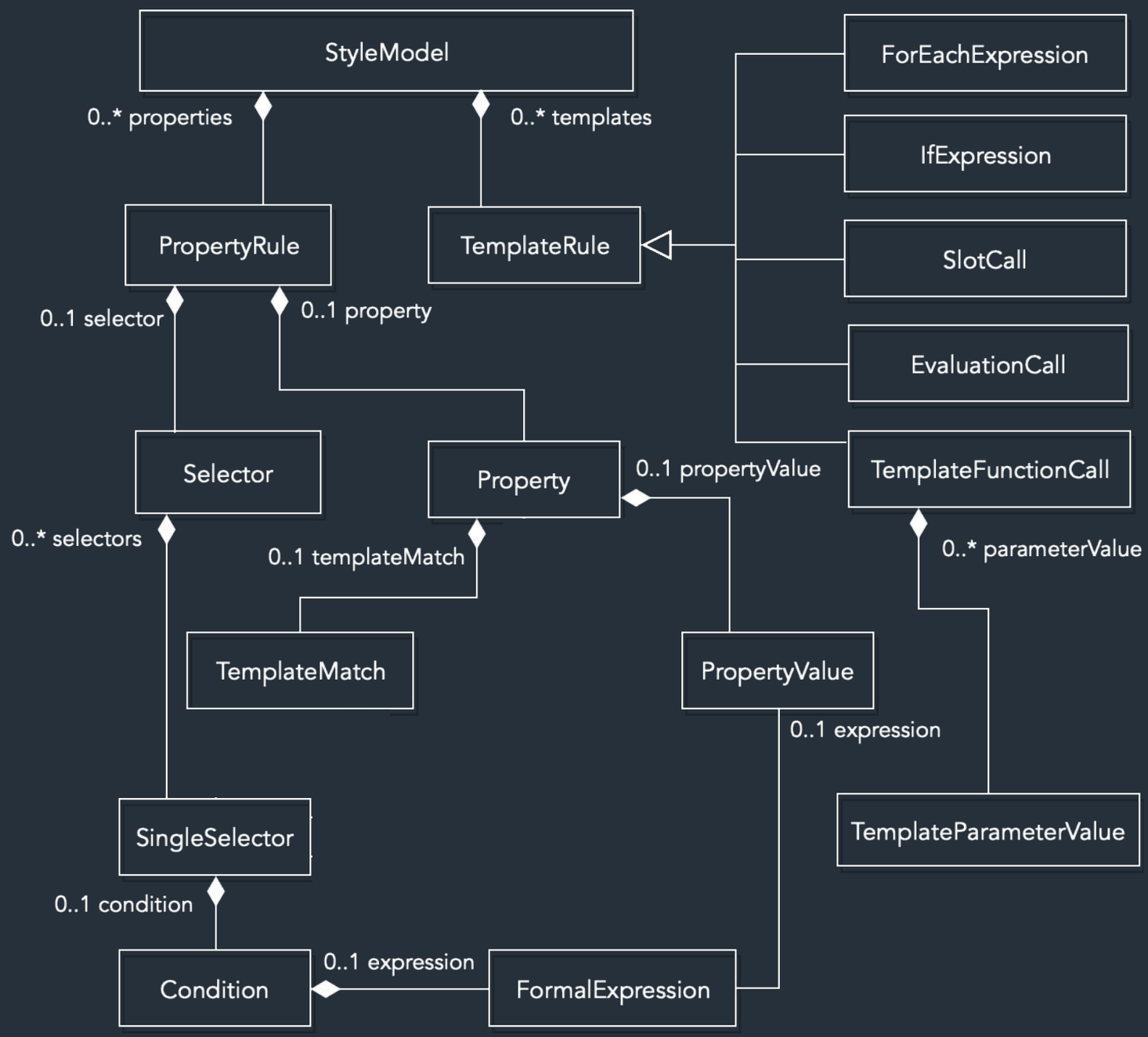
ECSS Language

PropertyValue *expression* (FormalExpression)



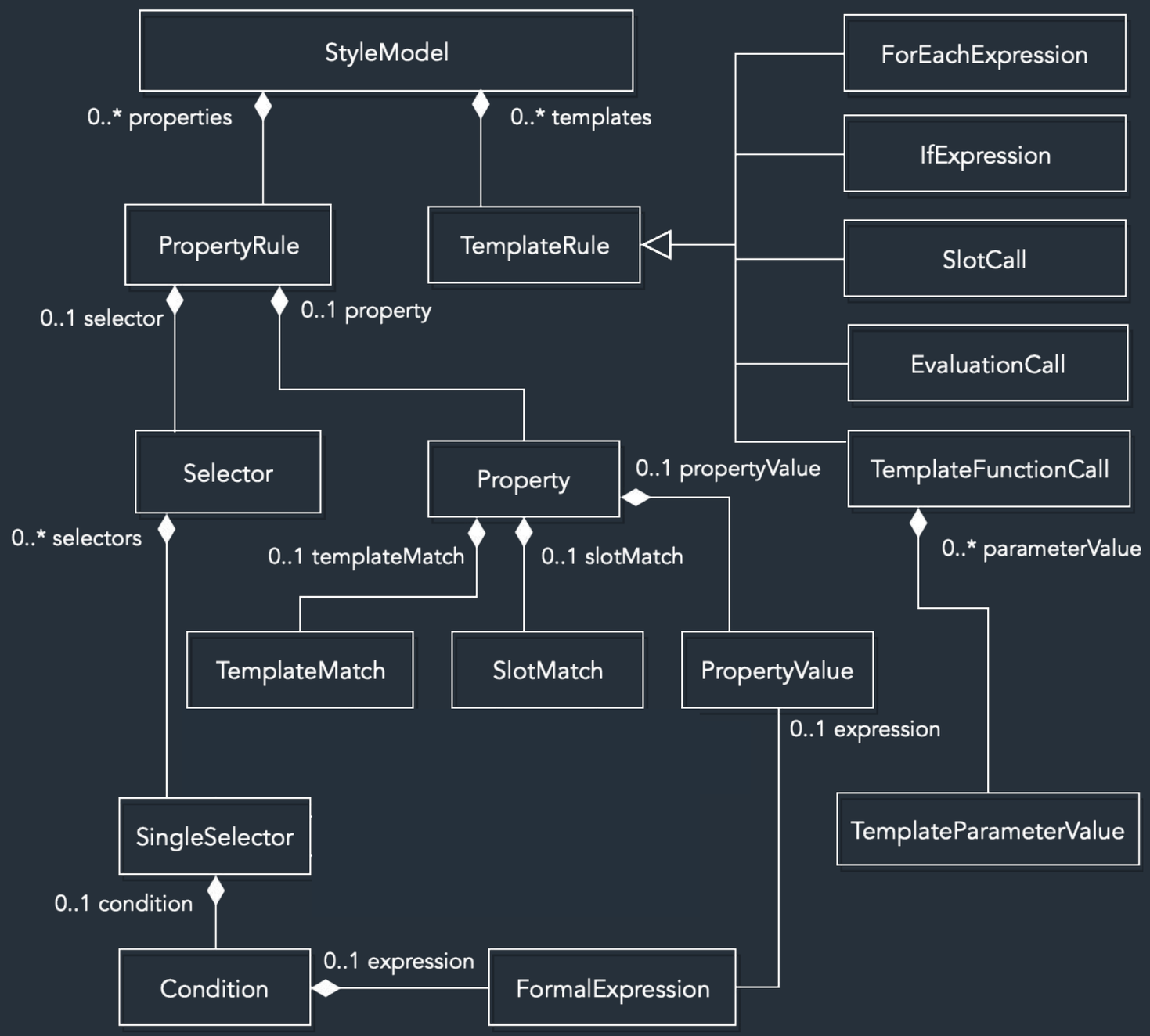
ECSS Language

Property *templateMatch*



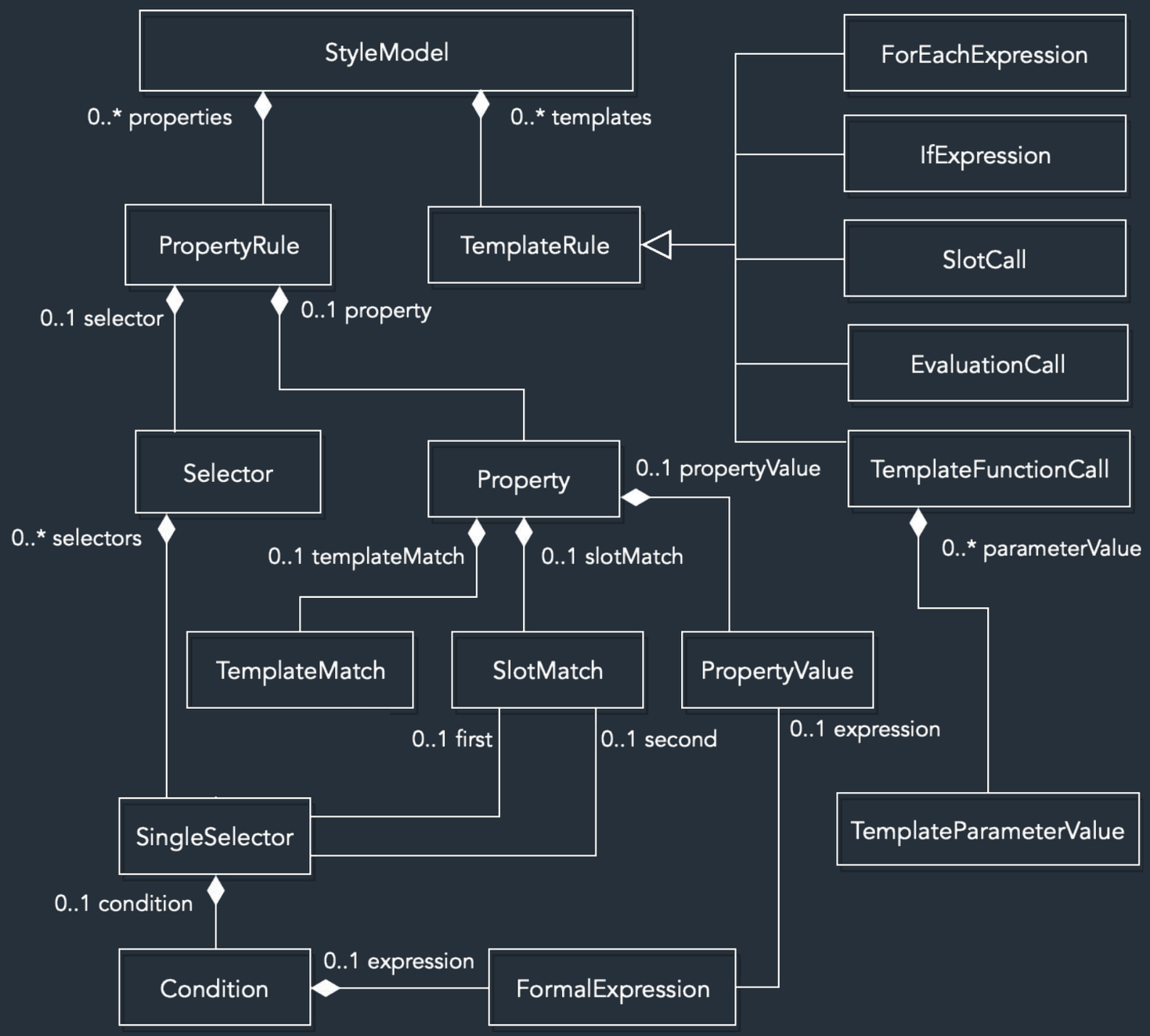
ECSS Language

Property *slotMatch*



ECSS Language

SlotMatch *first* and *second*



Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' "  " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " return " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Rule name

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name " "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' "  " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()"  priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Rule template generator

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' "  " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Rule group(s)

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' "  " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Defines rule *header*

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99]) ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

If slot *name* has no associated attribute, require the class name as first content

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99]) ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

If slot named *name* has associated attributes set, start object with the definition of these attributes

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[ 0 .. 99]) ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Rule call to select a subrule in the rule group *nameDistRules*

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[0 .. 99]) ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Create additional indentation for object features

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[0 .. 99]) ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Metamodel-dependent rule

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[0 .. 99]) ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

For subclasses of *NamedElement*, the *classname* is the name of the Ecore class in uppercase characters

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' "  " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Global rule (star-selector)

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' " " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[0 .. 99]) ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Lower-priority global rule will only be applied in case the higher-priority rule is not applicable

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' "  " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Set value of feature *classname* to be the (actual) name of the class

Style Modeling

Requirement #1: indentation-based layout

- *ws-aware.ecss*

```
import "default.ecss";

templateGen classGenTemplate extends classTemplate;

rule whitespaceClassRule:classGenTemplate:: classRules:
    class.name " returns " class.name ":" "{"
        class.name
    "}"
    [% if (slot_name.getValues().isEmpty()) { %]
        " '" classname "' "  " ':' "
    [% } else { %]
        nameDistRules(~name[0 .. 1])
    [% } %]
    ::BEGIN()
        " (" attributeDistRules(~other[0 .. 99])  ")"
    ::END()
    ';'
;

NamedElement+ {
    classname: ocl "rule.class.name.toUpperCase()" priority(2.0);
}

* {
    classname: ocl "rule.class.name" priority(1.0);
    slot(name,name): 2.0;
}
```

Associate features named *name* to the slot *name* with priority 2.0

Style Modeling

Requirement #2: arbitrary order of declaration

- *arbitrary-order.ecss*

```
import "ws-aware.ecss";
```

Import style model created to fulfil requirement #1

```
template attributeTemplate: uk.ac.york.cs.ecss.newproc.AttributeXtendRule;
templateGen attributeGenTemplate  extends attributeTemplate;

rule arbitraryAttributeDistr: attributeGenTemplate:: attributeDistRules:
     "("
     for esf: features  join ") | (" {
          attributeRule(esf)
          [% List<EStructuralFeature> smallerFeatures = new ArrayList(features); %]
          [% smallerFeatures.remove(esf);  boolean first = true; %]
          "(("
          for EStructuralFeature sub: smallerFeatures  join ") & (" {
               "','" attributeRule(sub)
          } "))"
     } ")"
;
```

Style Modeling

Requirement #2: arbitrary order of declaration

- *arbitrary-order.ecss*

Define mapping of template *attributeTemplate* to Java class *AttributeXtendRule*

```
import "ws-aware.ecss";

template attributeTemplate: uk.ac.york.cs.ecss.newproc.AttributeXtendRule;
templateGen attributeGenTemplate  extends attributeTemplate;

rule arbitraryAttributeDistr: attributeGenTemplate:: attributeDistRules:
    "("
    for esf: features  join ") | (" {
        attributeRule(esf)
        [% List<EStructuralFeature> smallerFeatures = new ArrayList(features); %]
        [% smallerFeatures.remove(esf);  boolean first = true; %]
        "(("
        for EStructuralFeature sub: smallerFeatures  join ") & (" {
            "',''" attributeRule(sub)
        } "))"
    } ")"
;
```

Style Modeling

Requirement #2: arbitrary order of declaration

- *arbitrary-order.ecss*

Define that rules of template generator *attributeGenTemplate* extend Java class *AttributeXtendRule*

```
import "ws-aware.ecss";

template attributeTemplate: uk.ac.york.cs.[illegible].newproc.AttributeXtendRule;
templateGen attributeGenTemplate  extends attributeTemplate;

rule arbitraryAttributeDistr: attributeGenTemplate:: attributeDistRules:
     "("
     for esf: features  join ") | (" {
          attributeRule(esf)
          [% List<EStructuralFeature> smallerFeatures = new ArrayList(features); %]
          [% smallerFeatures.remove(esf);  boolean first = true; %]
          "(("
          for EStructuralFeature sub: smallerFeatures  join ") & (" {
               "','" attributeRule(sub)
          } "))"
     } ")"
;
```

Style Modeling

Requirement #2: arbitrary order of declaration

- *arbitrary-order.ecss*

```
import "ws-aware.ecss";

template attributeTemplate: uk.ac.york.cs.ecss.newproc.AttributeXtendRule;
templateGen attributeGenTemplate  extends attributeTemplate;

rule arbitraryAttributeDistr: attributeGenTemplate:: attributeDistRules:
    "("
    for esf: features join ") | (" {
        attributeRule(esf)
        [% List<EStructuralFeature> smallerFeatures = new ArrayList(features); %]
        [% smallerFeatures.remove(esf); boolean first = true; %]
        "(("
        for EStructuralFeature sub: smallerFeatures join ") & (" {
            "',' attributeRule(sub)
        } "))"
    } ")"
;
```

Arbitrary first feature

Unordered group of remaining features, prefixed by comma

Style Modeling

Requirement #2: arbitrary order of declaration

- *arbitrary-order.ecss*

```
import "ws-aware.ecss";

template attributeTemplate: uk.ac.york.cs.ecss.newproc.AttributeXtendRule;
templateGen attributeGenTemplate  extends attributeTemplate;

rule arbitraryAttributeDistr: attributeGenTemplate:: attributeDistRules:
    "("
    for esf: features  join ") | (" {
        attributeRule(esf)
        [% List<EStructuralFeature> smallerFeatures = new ArrayList(features); %]
        [% smallerFeatures.remove(esf);  boolean first = true; %]
        "(("
        for EStructuralFeature sub: smallerFeatures  join ") & (" {
            "','" attributeRule(sub)
        } "))"
    } ")"
;
```

Allow an arbitrary feature to occur first

Style Modeling

Requirement #2: arbitrary order of declaration

- *arbitrary-order.ecss*

```
import "ws-aware.ecss";

template attributeTemplate: uk.ac.york.cs.ecss.newproc.AttributeXtendRule;
templateGen attributeGenTemplate  extends attributeTemplate;

rule arbitraryAttributeDistr: attributeGenTemplate:: attributeDistRules:
    "("
    for esf: features join ") | (" {
        attributeRule(esf)
        [% List<EStructuralFeature> smallerFeatures = new ArrayList(features); %]
        [% smallerFeatures.remove(esf); boolean first = true; %]
        "(("
        for EStructuralFeature sub: smallerFeatures join ") & (" {
            "','" attributeRule(sub)
        } "))"
    } ")"
;
```

Calculate remaining features using *embedded* Java code

Style Modeling

Requirement #2: arbitrary order of declaration

- *arbitrary-order.ecss*

```
import "ws-aware.ecss";

template attributeTemplate: uk.ac.york.cs.ecss.newproc.AttributeXtendRule;
templateGen attributeGenTemplate  extends attributeTemplate;

rule arbitraryAttributeDistr: attributeGenTemplate:: attributeDistRules:
     "("
     for esf: features  join ") | (" {
          attributeRule(esf)
          [% List<EStructuralFeature> smallerFeatures = new ArrayList(features); %]
          [% smallerFeatures.remove(esf); boolean first = true; %]
          "(("
          for EStructuralFeature sub: smallerFeatures  join ") & (" {
               "',' attributeRule(sub)
          } "))"
     } ")"
;
```

Remaining features are prefixed with ',' and of arbitrary order

Example Scenario

Styled Grammar (excerpt w/o comma-prefixed feature separation)

```
grammar spacetransportationservice.MyEcssGenSts

import "http://cs.york.ac.uk/ecss/examples/spacetransportationservice"
import "http://www.eclipse.org/emf/2002/Ecore"  as ecore

...

LaunchSite: 'LAUNCHSITE'   ':'
    {LaunchSite}
    'name' ':' name=IDDT
    BEGIN (
        'locationLatitude' ':' locationLatitude=EDOUBLEDT  ';' &
        'locationLongitude' ':' locationLongitude=EDOUBLEDT  ';' &
        ( 'operator' ':' operator=ESTRING )?  ';' &
        'numberOfLaunchpads' ':' numberOfLaunchpads=EINT  ';' &
        ( 'physicalProperties' ':' BEGIN
            physicalProperties+=PhysicalProperty
            physicalProperties+=PhysicalProperty * END)?';' &
        'operational' ':' operational?=EBOOLEAN  ';'
    ) END;

IDDT:
    EBOOLEAN | ID;

...

terminal BEGIN: 'synthetic:BEGIN';
terminal END: 'synthetic:END';
terminal ID: '^'? ('a'..'z' | 'A'..'Z' | '_') ('a'..'z' | 'A'..'Z' | '0'..'9' | '_' )*;

...
```

Arbitrary order of declaration

Indentation-based layout

*Does **not** show comma-prefixed grammar (quadratically larger)*

Indentation-based layout

Comma-prefixed grammar (quadratically larger)

Arbitrary order of declaration

```
LaunchSite : 'LAUNCHSITE' ':'
   {LaunchSite}
   'name' ':' name = IDDT
   BEGIN (
      'locationLatitude' ':' locationLatitude= EDOUBLEDT
      (',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
         ',' ( 'operator' ':' operator= ESTRING )? &
         ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
         ',' ( 'physicalProperties' ':'
            BEGIN
               physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
            END
      )? &
      ',' 'operational' ':' operational?= EBOOLEAN ) |
      'locationLongitude' ':' locationLongitude= EDOUBLEDT (
         ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
         ',' ( 'operator' ':' operator= ESTRING )? &
         ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
         ',' ( 'physicalProperties' ':'
         BEGIN
            physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
         END
      )? &
      ',' 'operational' ':' operational?= EBOOLEAN ) |
      ( 'operator' ':' operator= ESTRING )? (
         ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
         ',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
         ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
         ',' ( 'physicalProperties' ':'
         BEGIN
            physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
         END
      )? &
      ',' 'operational' ':' operational?= EBOOLEAN ) |
      'numberOfLaunchpads' ':' numberOfLaunchpads= EINT (
         ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
         ',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
         ',' ( 'operator' ':' operator= ESTRING )? &
         ',' ( 'physicalProperties' ':'
         BEGIN
            physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
         END
         )? & ',' 'operational' ':' operational?= EBOOLEAN ) | (
            'physicalProperties' ':'
            BEGIN
               physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
            END )? (
               ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
               ',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
               ',' ( 'operator' ':' operator= ESTRING )? &
               ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
               ',' 'operational' ':' operational?= EBOOLEAN
            ) |
            'operational' ':' operational?= EBOOLEAN (
               ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
               ',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
               ',' ( 'operator' ':' operator= ESTRING )? &
               ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
               ',' ( 'physicalProperties' ':'
                  BEGIN
                     physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
                  END
               )?
            )
      )
   END;
```

Indentation-based layout

Comma-prefixed grammar (quadratically larger)

Arbitrary order of declaration

```
LaunchSite : 'LAUNCHSITE' ':'
   {LaunchSite}
   'name' ':' name = IDDT
   BEGIN (
      'locationLatitude' ':' locationLatitude= EDOUBLEDT
      (',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
         ',' ( 'operator' ':' operator= ESTRING )? &
         ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
         ',' ( 'physicalProperties' ':'
            BEGIN
               physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
            END
      )? &
      ',' 'operational' ':' operational?= EBOOLEAN ) |
      'locationLongitude' ':' locationLongitude= EDOUBLEDT (
         ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
         ',' ( 'operator' ':' operator= ESTRING )? &
         ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
         ',' ( 'physicalProperties' ':'
         BEGIN
            physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
         END
      )? &
      ',' 'operational' ':' operational?= EBOOLEAN ) |
      ( 'operator' ':' operator= ESTRING )? (
         ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
         ',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
         ',' 'numberOfLaunchpads' ':' numberOfLaunchpads= EINT &
         ',' ( 'physicalProperties' ':'
         BEGIN
            physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
         END
   )? &
   ',' 'operational' ':' operational?= EBOOLEAN ) |
   'numberOfLaunchpads' ':' numberOfLaunchpads= EINT (
      ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
      ',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
      ',' ( 'operator' ':' operator= ESTRING )? &
      ',' ( 'physicalProperties' ':'
      BEGIN
         physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
      END
      )? & ',' 'operational' ':' operational?= EBOOLEAN ) | (
         'physicalProperties' ':'
         BEGIN
            physicalProperties+= PhysicalProperty physicalProperties+= PhysicalProperty *
         END )? (
            ',' 'locationLatitude' ':' locationLatitude= EDOUBLEDT &
            ',' 'locationLongitude' ':' locationLongitude= EDOUBLEDT &
            ',' ( 'operator' ':' operator= ESTRING )? &
```

Conclusion

Reusable Textual Styles for Domain-Specific Modeling Languages

- Extends model-first approach with style models
- Style language for modeling of textual representation of structural (meta-)language components
  - Supports metamodel-agnostic and metamodel-dependent styles

Future Work

- Modeling of styles for open-source DSMLs
  - Comparison of expressiveness and conciseness of grammar and style model
- Extension and adaptation of ECSS language in regards to findings (previous point)
- Empirical user study on usability of proposed approach

Reusable Textual Styles for Domain-Specific Modeling Languages

Patrick Neubauer[1], Robert Bill[2], Dimitris Kolovos[1], Richard Paige[3], Manuel Wimmer[4]

[1] University of York, UK

[2] Technische Universität Wien, Austria

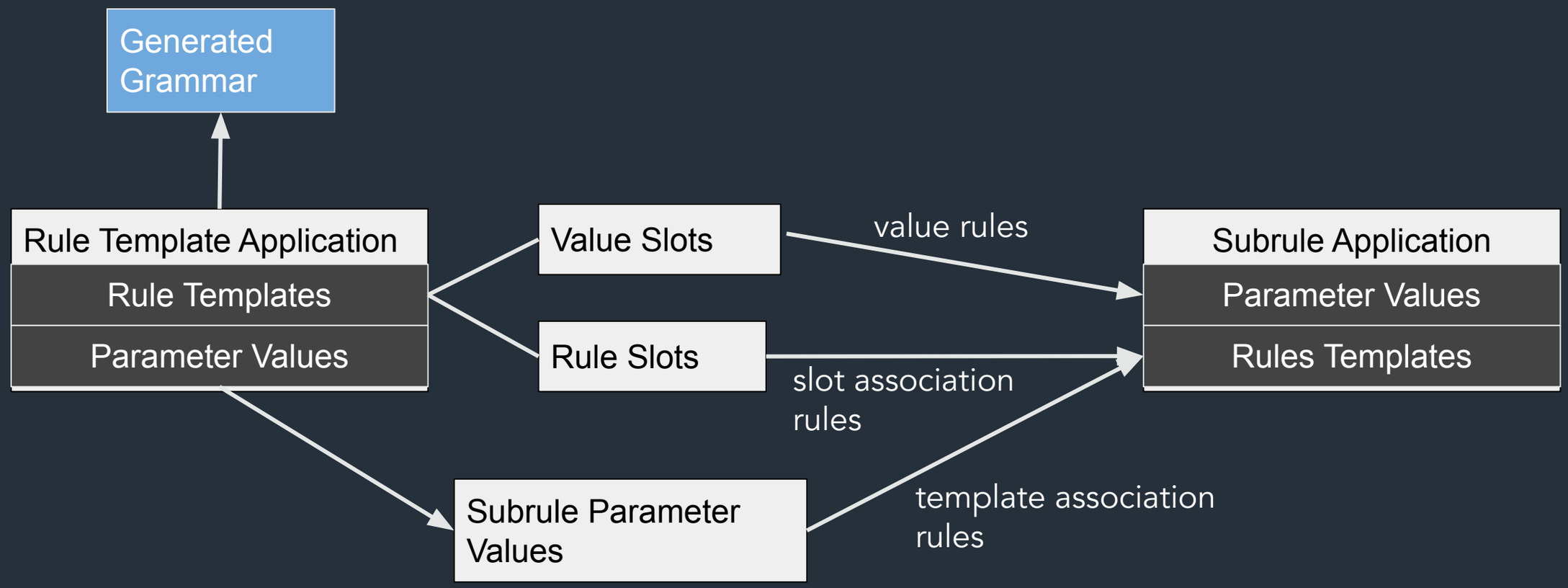
[3] McMaster University, Canada

[4] Johannes Kepler Universität, Austria

19th International Workshop in OCL and Textual Modeling, 2019, Munich, Germany

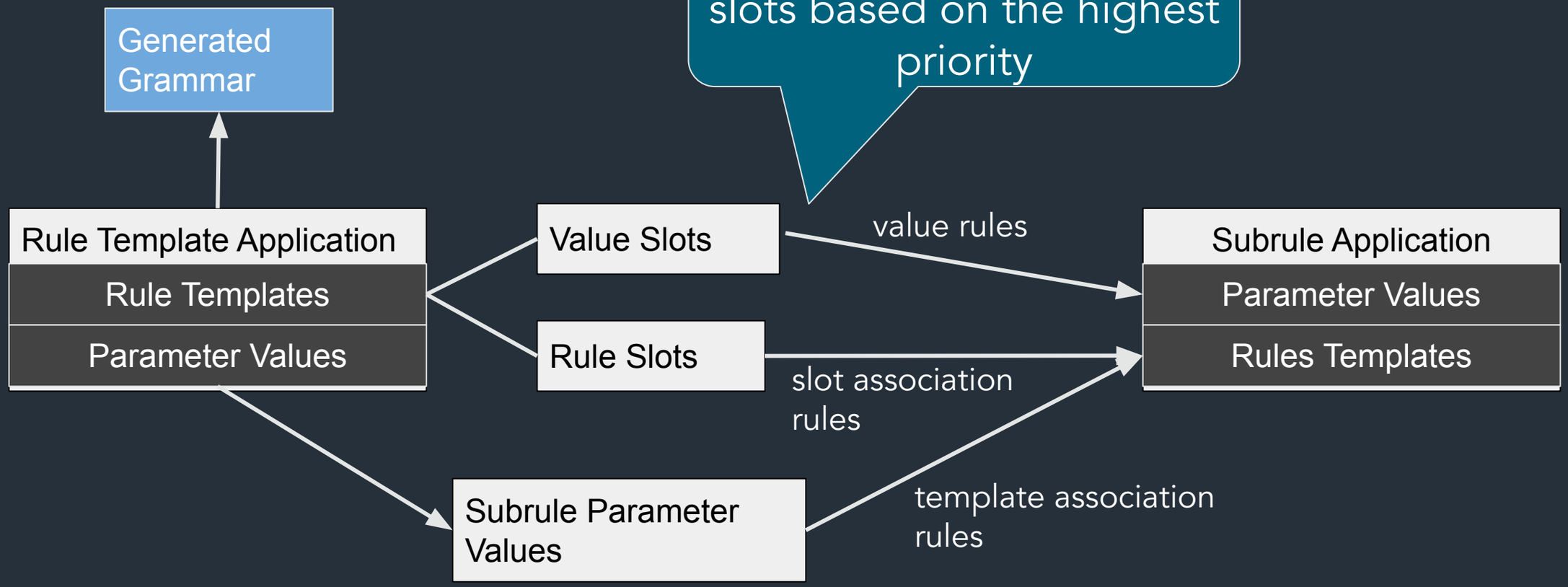
Grammar and Language Generation

Rule value assignment process and grammar rule generation



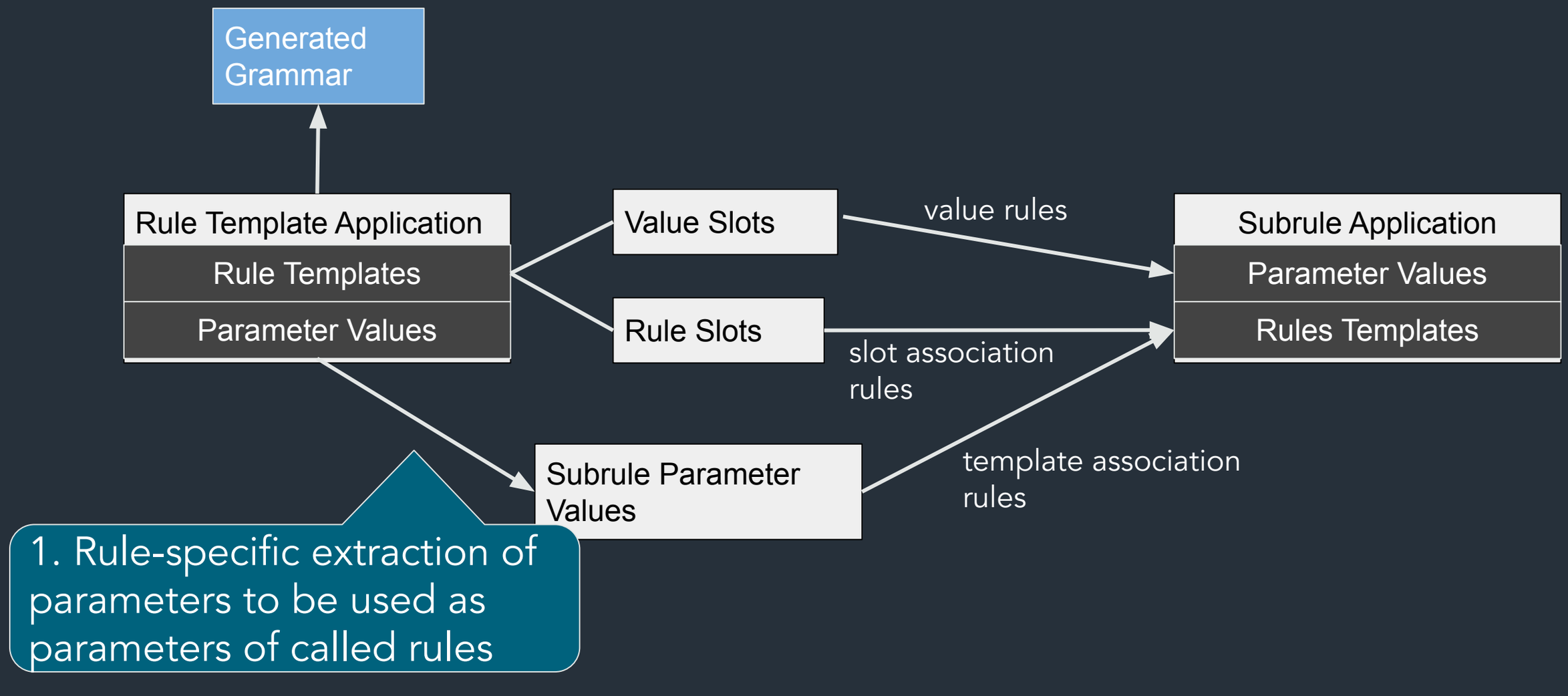
Grammar and Language Generation

Rule value assignment process and grammar rule generation



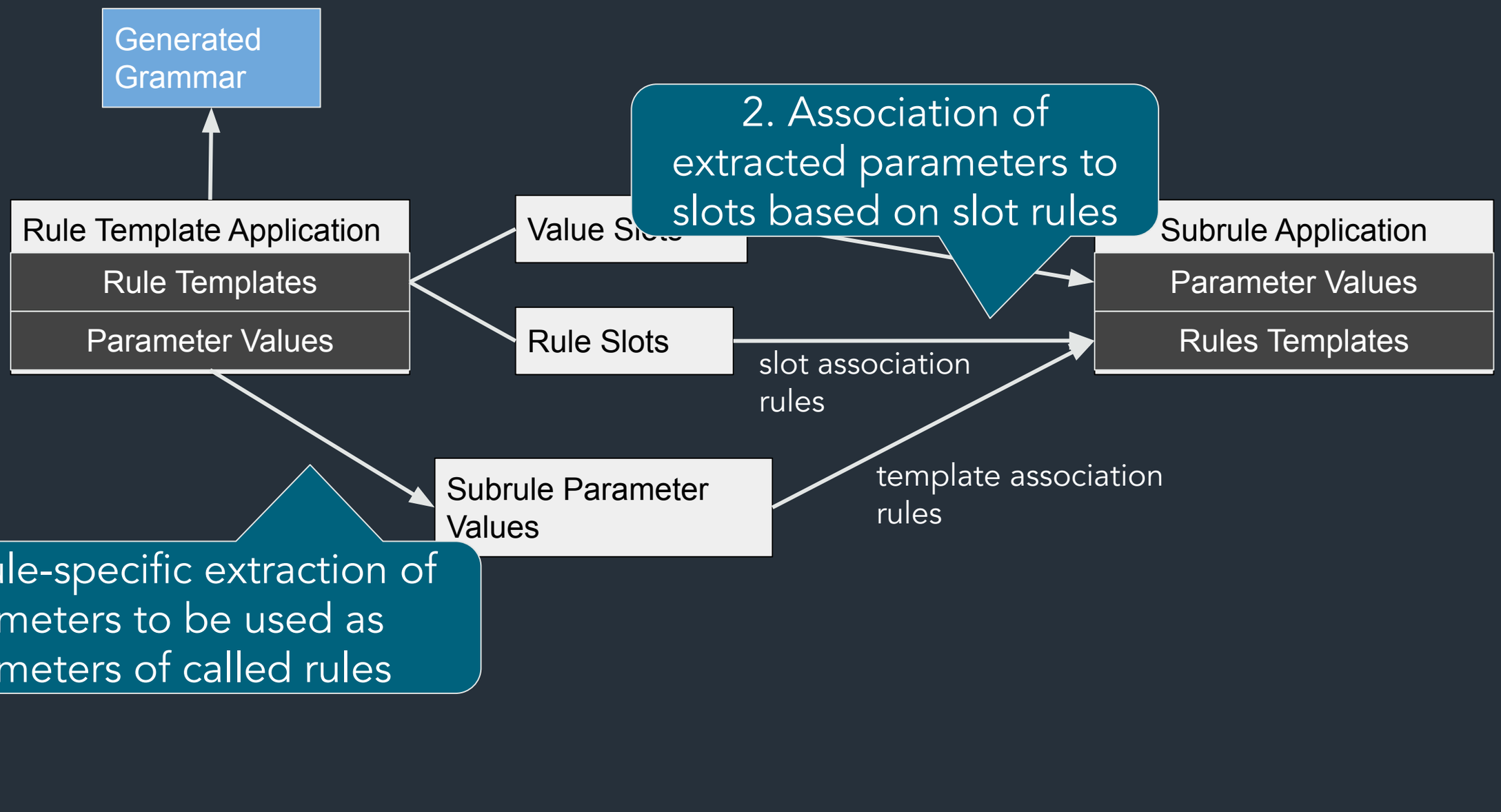
Grammar and Language Generation

Rule value assignment process and grammar rule generation



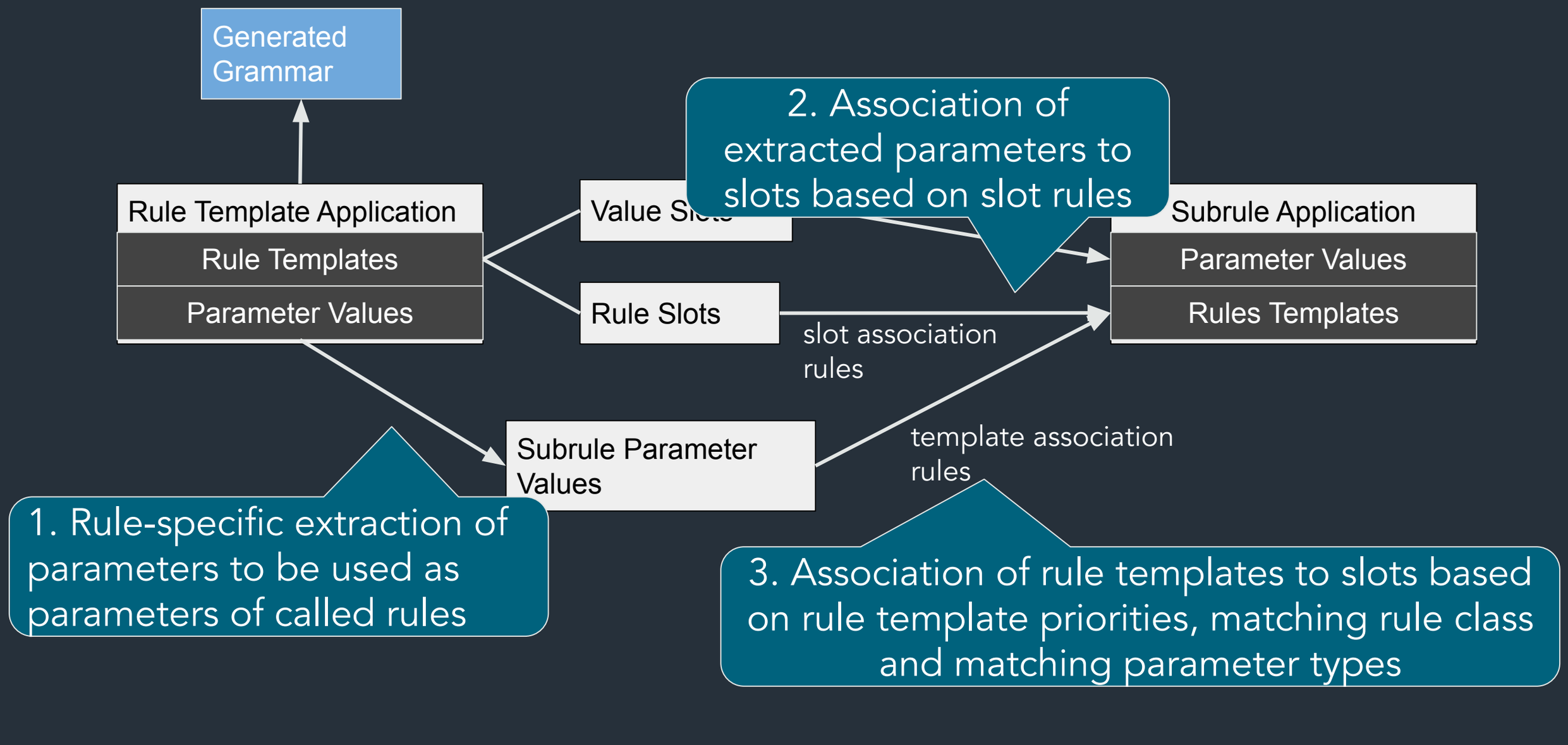
Grammar and Language Generation

Rule value assignment process and grammar rule generation



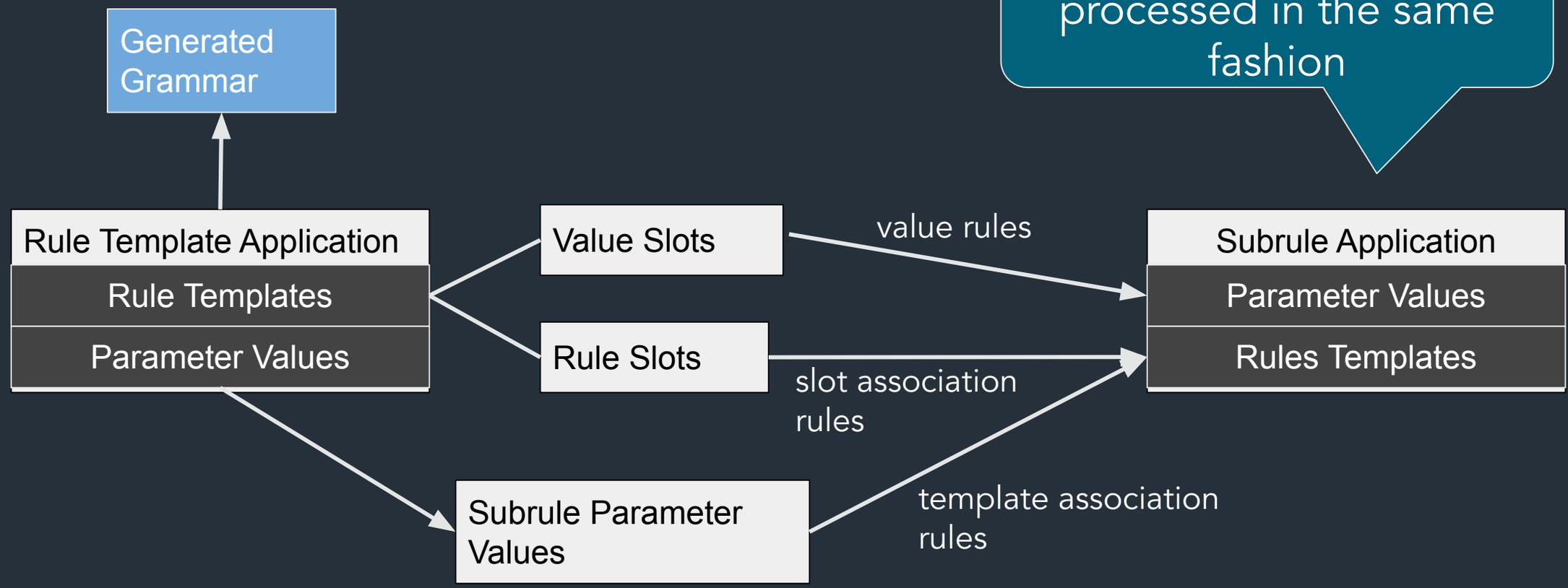
Grammar and Language Generation

Rule value assignment process and grammar rule generation



Grammar and Language Generation

Rule value assignment process and grammar rule generation



Grammar and Language Generation

Rule value assignment process and grammar rule generation