Network Working Group   J. Halpern
Internet-Draft   Ericsson
Intended status: Informational   J. Strassner
Expires: April 16, 2017   Huawei Technologies
                                                  S. Van der Meer
                                                          Ericsson
                                                  October 13, 2016

Generic Policy Data Model for
Simplified Use of Policy Abstractions (SUPA)
draft-ietf-supa-generic-policy-data-model-02

Abstract

This document defines two YANG policy data modules. The first is a
generic policy model that is meant to be extended on an application-
specific basis. The second is an exemplary extension of the first
generic policy model, and defines rules as event-condition-action
policies. Both models are independent of the level of abstraction of
the content and meaning of a policy.

Status of this Memo

Copyright Notice

Table of Contents

1. Overview

   This document defines two YANG [RFC6020] [RFC6991] policy data
   models. The first is a generic policy model that is meant to be
   extended on an application-specific basis. It is derived from the
   Generic Policy Information Model (GPIM) defined in [1]. The second
   is an exemplary extension of the first (generic policy) model, and
   defines policy rules as event-condition-action tuples. Both models
   are independent of the level of abstraction of the content and
   meaning of a policy.

   The GPIM defines a common framework as a set of model elements
   (e.g., classes, attributes, and relationships) that specify a
   common set of policy management concepts that are independent of the
   type of policy (e.g., imperative, procedural, declarative, or
   otherwise). The first YANG data model is a translation of the GPIM
   to a YANG module. The ECA Policy Rule Information Model (EPRIM),
   also defined in [1], extends the GPIM to represent policy rules that
   use the Event-Condition-Action (ECA) paradigm. The second YANG data
   model maps the EPRIM to YANG. The second YANG data model MAY be
   used to augment the functionality of the first YANG data model.

2. Conventions Used in This Document

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in
   this document are to be interpreted as described in [RFC2119]. In
   this document, these words will appear with that interpretation
   only when in ALL CAPS. Lower case uses of these words are not to
   be interpreted as carrying [RFC2119] significance.

3. Terminology

   This section defines acronyms, terms, and symbology used in the
   rest of this document.

3.1. Acronyms

   CNF   Conjunctive Normal Form
   DNF   Disjunctive Normal Form
   ECA   Event-Condition-Action
   EPRIM   (SUPA) ECA Policy Rule Information Model [1]
   FQDN   Fully Qualified Domain Name
   FQPN   Fully Qualified Path Name
   GPIM   (SUPA) Generic Policy Information Model [1]
   GUID   Globally Unique IDentifier
   NETCONF   Network Configuration protocol
   OAM&P   Operations, Administration, Management, and Provisioning
   OCL   Object Constraint Language {2} [3]
   OID   Object IDentifier
   SUPA   Simplified Use of Policy Abstractions
   UML   Unified Modeling Language
   URI   Uniform Resource Identifier
   UUID   Universally Unique IDentifier

3.2. Definitions

   Action: a set of activities that have a set of associated behavior.

   Boolean Clause: a logical statement that evaluates to either TRUE
      or FALSE. Also called Boolean Expression.

   Condition: a set of attributes, features, and/or values that are to
      be compared with a set of known attributes, features, and/or
      values in order to make a decision. A Condition, when used in
      the context of a Policy Rule, is used to determine whether or not
      the set of Actions in that Policy Rule can be executed or not.

   Constraint: A constraint is a limitation or restriction.
      Constraints may be added to any type of object (e.g., events,

conditions, and actions in Policy Rules).

   Data Model: a data model is a representation of concepts of
      interest to an environment in a form that is dependent on data
      repository, data definition language, query language,
      implementation language, and protocol (typically one or more of
      these). This definition is taken from [1].

   ECA: Event - Condition - Action (a type of policy).

   Event: an Event is defined as any important occurrence in time in
      the system being managed, and/or in the environment of the system
      being managed. An Event may represent the changing or maintaining
      of the state of a managed object. An Event, when used in the
      context of a Policy Rule, is used to determine whether the
      Condition clause of an imperative (i.e., ECA) Policy Rule can be
      evaluated or not.

   FQPN (FUlly Qualified Path Name)
      The specification of a path to a file in a system that
      unambiguously resolves to only that specific file. In this
      sense, "fully qualified" is independent of context. However,
      in a distributed system, it may be dependent on the specific
      format of an operating system. Hence, implementations should
      consider such issues before allowing the use of FQPNs.

   Information Model: an information model is a representation of
      concepts of interest to an environment in a form that is
      independent of data repository, data definition language, query
      language, implementation language, and protocol. This definition
      is taken from [1].

   Metadata: metadata is data that provides descriptive and/or
      prescriptive information about the object(s) to which it is
      associated. This enables structure and content of the object(s)
      to which it applies, as well as usage and other information, to
      be represented in an extensible manner. It avoids "burying"
      common information in specific classes, and increases reuse.

   SUPAPolicy: A SUPAPolicy is, in this version of this document, an ECA
      policy rule that MUST contain an ECA policy rule, SHOULD contain
      one or more SUPAPolicyMetadata objects, and MAY contain other
      elements that define the semantics of the policy rule. An ECA
      Policy Rule MUST contain an event clause, a condition clause, and
      an action clause. Policies are generically defined as a means to
      monitor and control the changing and/or maintaining of the state
      of one or more managed objects. This definition is based on the
      definition of SUPAPolicy in [1].

3.3. Symbology

The following representation is used to describe YANG data modules
defined in this draft.

o Brackets "[" and "]" enclose list keys.

o Abbreviations before data node names: "rw" means configuration
   data (read-write), and "ro" means state data (read-only).

o Symbols after data node names: "?" means an optional node, "!"
   means a presence container, and "*" denotes a list and leaf-list.

o Parentheses enclose choice and case nodes, and case nodes are also
   marked with a colon (":").

o Ellipsis ("...") stands for contents of subtrees that are not
   shown.

4. Design of the SUPA Policy Data Models

This section describes the design philosophy of the YANG data model,
and how they will be maintained.

4.1. Objectives

These Data Models are derived from the SUPA Generic Policy
Information Model [1]. The overall objective is to faithfully
transform that information model into a YANG data model that can
be used for communicating policy. The policy scope to be covered is
that defined by [1]; please refer to it for more details and
background information.

This model is an extensible framework that is independent of the
implementation approach for storing polices, as well as being
independent of the content and meaning of specific policies. These
models can be extended (generally by using the groupings here and
defining additional containers for concrete classes) to represent
domain- and/or application-specific policies. The ECA model in this
document is an example of extending the general policy model towards
specific policies.

By using this approach, different policy models will use common
semantics, enabling them to be more easily integrated.

One of the important goals of this work is for the semantics of
these models to align with those of the generic policy information
model. Thus, most of this model was generate by a quasi-algorithmic
transformation of the information model. This was done by hand.
Certain changes were made to reflect the fact that this is a YANG
data model, and therefore, does not need to generically allow for
all data modelling languages. Details of the process are described
below in section 4.3.

4.2 Yang Data Model maintenance

All model changes should be done to both the information model and
the data model in parallel. Care is being taken during development
of this model to ensure that is the case.

In general, structural changes will be applied to both the
information model and the data model, and then any necessary YANG
repairs taken to preserve the validity of the YANG data model.

4.3 YANG Data Model Overview

This YANG data model is generated by applying suitable YANG
constructs to represent the information in the information model.

There are three key information modeling concepts that this data
model needs to represent consistently. These are classes, class
inheritance (also known as subclassing) and associations. The
SUPA generic policy information model [1] makes extensive use of
these concepts.

Each class in the model is represented by a YANG identity and by a
YANG grouping. The use of groupings enables us to define these
classes abstractly. Each grouping begins with two leaves (either
defined in the grouping or inherited via a uses clause), which
provide common functionality. One leaf is used for the system-wide
unique identifier for this instance. This is either named
supa-policy-ID (for the SUPAPolicyObject tree, which contains
everything EXCEPT metadata objects) or supa-policy-metadata-id (for
the SUPAPolicyMetadata tree, which ONLY contains metadata). All
associations use supa-policy-IDs. The second leaf is always called
the entity-class. It is an identityref which is set to the identity
of the instance. The default value for this leaf is always
correctly defined by the grouping. It is read-write in the YANG
formalism due to restrictions on the use of MUST clauses.

Class inheritance (or subclassing) is done by defining an identity
and a grouping for the new class. The identity is based on the
parent identity, and is given a new name to represent this class.
The new grouping uses the parent grouping. It refines the

entity-class of the parent, replacing the default value of the
entity-class with the correct value for this class.

Associations are represented by the use of instance-identifiers and association classes. Association classes are classes, using the above construction, which contain leaves representing the set of instance-identifiers for each end of the association, along with any other properties the information model assigns to the association. The two associated classes each have a leaf with an instance-identifier that points to the association class instance. Each instance-identifier leaf is defined with a must clause. That must clause references the entity-class of the target of the instance-identifier, and specifies that the entity class type must be the same as, or subclassed from, a specific named class. Thus, associations can point to any instance of a selected class, or any instance of any subclass of that target.

While not mandated by the YANG, it is expected that the xpath for the instance-identifier will end with an array selection specifying the supa-policy-ID or supa-policy-metadata-id of the target. This enables us to construct the abstract class tree, with inheritance and associations. It is noted and accepted that this process does lose the distinction between containment, association, and aggregation used by the information model.

The concrete class tree is constructed as follows. The YANG model defines a container for each class that is defined as concrete by the information model. That container contains a single list, keyed by either the supa-policy-id or the supa-policy-metadata-id. The content of the list is defined by a uses clause referencing the grouping that defines the class. After this was done, certain additional modifications were made. Specifically, any information model constructs intended to represent lists of possible values were recast as YANG enumerations. Where these lists are used more than once, they are factored out into reusable enumerations.

Certain attributes that are not needed in the YANG (e.g., to represent the range of choices different data models might use for policy identification) were removed for simplicity and clarity.

4.4. YANG Tree Diagram

The YANG Tree Diagram starts on the next page. It uses the following abbreviations for datatypes:

```
 - B:    Boolean
 - E:    enumeration
 - II:   instance-identifier
 - IR:   identityref
 - PC:   policy-constraint-language-list
 - PD:   policy-data-type-encoding-list
 - S:    string
```

```
   - YD:   yang:date-and-time
   - UI:   uint32
```

```
module:  ietf-supa-policy
   +--rw supa-encoding-clause-container
   |  +--rw supa-encoding-clause-list* [supa-policy-ID]
   |     +--rw entity-class?                                   IR
   |     +--rw supa-policy-ID                                  S
   |     +--rw supa-policy-name?                               S
   |     +--rw supa-policy-object-description?                 S
   |     +--rw supa-has-policy-metadata-agg-ptr*               II
   |     +--rw supa-has-policy-component-decorator-part-ptr    II
   |     +--rw supa-policy-clause-deploy-status                E
   |     +--rw supa-has-policy-clause-part-ptr*                II
   I     +--rw supa-encoded-clause-content                     S
   I     +--rw supa-encoded-clause-language                    E
   +--rw supa-policy-variable-container
   |  +--rw supa-policy-variable-list* [supa-policy-ID]
   |     +--rw entity-class?                                   IR
   |     +--rw supa-policy-ID                                  S
   |     +--rw supa-policy-name?                               S
   |     +--rw supa-policy-object-description?                 S
   |     +--rw supa-has-policy-metadata-agg-ptr*               II
   |     +--rw supa-has-policy-component-decorator-part-ptr    II
   |     +--rw supa-has-policy-component-decorator-agg-ptr*    II
   |     +--rw supa-decorator-constraints*                     S
   |     +--rw supa-has-decorator-constraint-encoding?         PC
   |     +--rw supa-policy-term-is-negated?                    B
   |     +-rw supa-policy-variable-name?                       S
   +--rw supa-policy-operator-container
   |  +--rw supa-policy-operator-list* [supa-policy-ID]
   |     +--rw entity-class?                                   IR
   |     +-rw supa-policy-ID                                   S
   |     +--rw supa-policy-name?                               S
   |     +--rw supa-policy-object-description?                 S
   |     +--rw supa-has-policy-metadata-agg-ptr*               II
   |     +--rw supa-has-policy-component-decorator-part-ptr    II
   |     +--rw supa-has-policy-component-decorator-agg-ptr*    II
   |     +--rw supa-decorator-constraints*                     S
   |     +--rw supa-has-decorator-constraint-encoding?         PC
   |     +--rw supa-policy-term-is-negated?                    B
   |     +--rw supa-policy-value-op-type                       E
   +--rw supa-policy-value-container
   |  +--rw supa-policy-value-list* [supa-policy-ID]
   |     +--rw entity-class?                                   IR
   |     +--rw supa-policy-ID                                  S
   |     +--rw supa-policy-name?                               S
   |     +--rw supa-policy-object-description?                 S
   |     +--rw supa-has-policy-metadata-agg-ptr*               II
   |     +--rw supa-has-policy-component-decorator-part-ptr    II
   |     +--rw supa-has-policy-component-decorator-agg-ptr*    II
   |     +--rw supa-decorator-constraints*                     S
```

```
|      +--rw supa-has-decorator-constraint-encoding?        PC
|      +--rw supa-policy-term-is-negated?                   B
```

```
   |      +--rw supa-policy-value-content*                        S
   |      +--rw supa-policy-value-encoding?                       PD
   +--rw supa-policy-generic-decorated-container
   |   +--rw supa-encoding-clause-list* [supa-policy-ID]
   |      +--rw entity-class?                                     IR
   |      +--rw supa-policy-ID                                    S
   |      +--rw supa-policy-name?                                 S
   |      +--rw supa-policy-object-description?                   S
   |      +--rw supa-has-policy-metadata-agg-ptr*                 II
   |      +--rw supa-has-policy-component-decorator-part-ptr      II
   |      +--rw supa-has-policy-component-decorator-agg-ptr*      II
   |      +--rw supa-decorator-constraints*                       S
   |      +--rw supa-has-decorator-constraint-encoding?           PC
   |      +--rw supa-policy-generic-decorated-content*            S
   |      +--rw supa-policy-generic-decorated-encoding?           PD
   +--rw supa-policy-source-container
   |   +--rw supa-source-list* [supa-policy-ID]
   |      +--rw entity-class?                                     IR
   |      +--rw supa-policy-ID                                    S
   |      +--rw supa-policy-name?                                 S
   |      +--rw supa-policy-object-description?                   S
   |      +--rw supa-has-policy-metadata-agg-ptr*                 II
   |      +--rw supa-has-policy-source-part-ptr                   II
   +--rw supa-policy-target-container
   |   +--rw supa-target-list* [supa-policy-ID]
   |      +--rw entity-class?                                     IR
   |      +--rw supa-policy-ID                                    S
   |      +--rw supa-policy-name?                                 S
   |      +--rw supa-policy-object-description?                   S
   |      +--rw supa-has-policy-metadata-agg-ptr*                 II
   |      +--rw supa-has-policy-target-part-ptr                   II
   +--rw supa-policy-concrete-metadata-container
   |   +--rw supa-policy-concrete-metadata-list*
   |                                   [supa-policy-metadata-id]
   |      +--rw entity-class?                                     IR
   |      +--rw supa-policy-metadata-id                           S
   |      +--rw supa-policy-metadata-description?                 S
   |      +--rw supa-policy-metadata-name?                        S
   |      +--rw supa-has-policy-metadata-part-ptr*                II
   |      +--rw supa-has-policy-metadata-dec-part-ptr*            II
   |      +--rw supa-policy-metadata-valid-period-end?            YD
   |      +--rw supa-policy-metadata-valid-period-start?          YD
   +--rw supa-policy-metadata-decorator-access-container
   |   +--rw supa-policy-metadata-decorator-access-list*
   |                                   [supa-policy-metadata-id]
   |      +--rw entity-class?                                     IR
   |      +--rw supa-policy-metadata-id                           S
   |      +--rw supa-policy-metadata-description?                 S
   |      +--rw supa-policy-metadata-name?                        S
```

```
|      +--rw supa-has-policy-metadata-part-ptr*             II
|      +--rw supa-has-policy-metadata-dec-part-ptr*         II
|      +--rw supa-has-policy-metadata-dec-agg-ptr?          II
```

```
   +--rw supa-policy-metadata-decorator-version-container
   |  +--rw supa-policy-metadata-decorator-version-list*
                                    [supa-policy-metadata-id]
   |     +--rw entity-class?                                  IR
   |     +--rw supa-policy-metadata-id                        S
   |     +--rw supa-policy-metadata-description?              S
   |     +--rw supa-policy-metadata-name?                     S
   |     +--rw supa-has-policy-metadata-part-ptr*             II
   |     +--rw supa-has-policy-metadata-dec-part-ptr*         II
   |     +--rw supa-has-policy-metadata-dec-agg-ptr?          II
   +--rw supa-policy-metadata-detail-container
   |  +--rw supa-policy-metadata-detail-list [supa-policy-ID]
   |     +--rw entity-class?                                  IR
   |     +--rw supa-policy-ID                                 S
   |     +--rw supa-policy-name?                              S
   |     +--rw supa-policy-object-description?                S
   |     +--rw supa-has-policy-metadata-agg-ptr*              II
   |     +--rw supa-has-policy-metadata-detail-agg-ptr?       II
   |     +--rw supa-has-policy-metadata-detail-part-ptr?      II
   |     +--rw supa-policy-metadata-detail-is-applicable?     B
   |     +--rw supa-policy-metadata-detail-constraint*        S
   |     +--rw supa-policy-metadata-detail-constraint-encoding?   PC
   +--rw supa-policy-component-decorator-detail-container
   |  +--rw supa-policy-component-decorator-detail-list*
                                           [supa-policy-ID]
   |     +--rw entity-class?                                  IR
   |     +--rw supa-policy-ID                                 S
   |     +--rw supa-policy-name?                              S
   |     +--rw supa-policy-object-description?                S
   |     +--rw supa-has-policy-metadata-agg-ptr*              II
   |     +--rw supa-has-policy-component-decorator-agg-ptr?   II
   |     +--rw supa-has-policy-component-decorator-part-ptr?  II
   |     +--rw supa-has-decorator-constraint*                 S
   |     +--rw supa-has-decorator-constraint-encoding         PC
   +--rw supa-policy-source-detail-container
   |  +--rw supa-policy-source-detail-list* [supa-policy-ID]
   |     +--rw entity-class?                                  IR
   |     +--rw supa-policy-ID                                 S
   |     +--rw supa-policy-name?                              S
   |     +--rw supa-policy-object-description?                S
   |     +--rw supa-has-policy-metadata-agg-ptr*              II
   I     +--rw supa-has-policy-source-detail-agg-ptr?         II
   I     +--rw supa-has-policy-source-detail-part-ptr?        II
   I     +--rw supa-policy-source-is-authenticated?           B
   I     +--rw supa-policy-source-is-trusted?                 B
   +--rw supa-policy-target-detail-container
   |  +--rw supa-policy-target-detail-list* [supa-policy-ID]
   |     +--rw entity-class?                                  IR
   |     +--rw supa-policy-ID                                 S
```

```
|      +--rw supa-policy-name?                          S
|      +--rw supa-policy-object-description?            S
|      +--rw supa-has-policy-metadata-agg-ptr*          II
```

```
   I      +--rw supa-has-policy-target-detail-agg-ptr?          II
   I      +--rw supa-has-policy-target-detail-part-ptr?         II
   I      +--rw supa-policy-target-is-authenticated?            B
   I      +--rw supa-policy-target-is-enabled?                  B
   +--rw supa-policy-clause-detail-container
   |  +--rw supa-policy-clause-detail-list* [supa-policy-ID]
   |     +--rw entity-class?                                    IR
   |     +--rw supa-policy-ID                                   S
   |     +--rw supa-policy-name?                                S
   |     +--rw supa-policy-object-description?                  S
   |     +--rw supa-has-policy-metadata-agg-ptr*                II
   |     +--rw supa-has-policy-clause-detail-agg-ptr?           II
   |     +--rw supa-has-policy-clause-detail-part-ptr?          II
   +--rw supa-policy-exec-fail-take-action-detail-container
   |  +--rw supa-policy-exec-fail-take-action-detail-list*
   |                                          [supa-policy-ID]
   |     +--rw entity-class?                                    IR
   |     +--rw supa-policy-ID                                   S
   |     +--rw supa-policy-name?                                S
   |     +--rw supa-policy-object-description?                  S
   |     +--rw supa-has-policy-metadata-agg-ptr*                II
   |     +--rw supa-has-exec-fail-action-detail-agg-ptr?        II
   |     +--rw supa-has-exec-fail-action-detail-part-ptr?       II
   |     +--rw supa-policy-exec-fail-take-action-name*          S
   +--rw supa-policy-metadata-decorator-detail-container
      +--rw supa-policy-metadata-decorator-detail-list*
                                       [supa-policy-metadata-id]
         +--rw entity-class?                                    IR
         +--rw supa-policy-metadata-id                          S
         +--rw supa-policy-metadata-description?                S
         +--rw supa-policy-metadata-name?                       S
         +--rw supa-has-policy-metadata-part-ptr*               II
         +--rw supa-has-policy-metadata-dec-part-ptr*           II
         +--rw supa-has-policy-metadata-detail-dec-agg-ptr?     II
         +--rw supa-has-policy-metadata-detail-dec-part-ptr?    II
```

[5](#). **SUPA Policy Data Model YANG Module**

The SUPA YANG data model module is divided into two main parts:

1) a set of containers that represent the objects that make
   updated a Policy Rule and its Policy Rule Components
2) a set of containers that represent the objects that define and
   apply metadata to Policy Rules and/or Policy Rule Components

Editor's note: This will be described in more detail in version 03

```
<CODE BEGINS> file "ietf-supa-policy@2016-10-10.yang"

module ietf-supa-policy {

    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-supa-policy";
    prefix supa-pdm;

    import ietf-yang-types {
        prefix yang;
    }

    organization "IETF";
        contact
            "Editor: Joel Halpern
             email: jmh@joelhalpern.com;
             Editor: John Strassner
             email: strazpdj@gmail.com;";

    description
        "This module defines a data model for generic high level
         definition of policies to be applied to a network.
         This module is derived from and aligns with
         draft-ietf-supa-generic-policy-info-model-01.
         Details on all classes, associations, and attributes
         can be found there.
         Copyright (c) 2015 IETF Trust and the persons identified
         as the document authors.  All rights reserved.
         Redistribution and use in source and binary forms, with or
         without modification, is permitted pursuant to, and
         subject to the license terms contained in, the Simplified
         BSD License set forth in Section 4.c of the IETF Trust's
         Legal Provisions Relating to IETF Documents
         (http://trustee.ietf.org/license-info).";

    revision "2016-10-10" {
        description
            "20161010:  Changed back to transitive identities (to
                        enforce inheritance) after determining that
                        errors were from a confdc bug.
             20161008:  Fixed errors found in latest pyang compiler
                        and from YANG Doctors.
             20161001:  Minor edits in association definitions.
             20160928:  Generated yang tree.
             20160924:  Rewrote association documentation; rebuilt
                        how all classes are named for consistency.
             20160904:  Optimization of module by eliminating leaves
                        that are not needed; rewrote section 4.
             20160824:  Edits to sync data model to info model.
```

20160720: Conversion to WG draft. Fixed pyang 1.1
compilation errors. Fixed must clause derefencing
used in grouping statements. Reformatted and expanded

```
            descriptions. Fixed various typos.
             20160321:  Initial version.";
        reference
            "draft-ietf-supa-policy-data-model-02";
    }

    typedef policy-constraint-language-list {
        type enumeration {
            enum "error" {
                description
                    "This signifies an error state.";
            }
            enum "init" {
                description
                    "This signifies a generic initialization state.";
            }
            enum "OCL2.4" {
                description
                    "Object Constraint Language v2.4 [2]. This is a
                     declarative language for describing rules for
                     defining constraints and query expressions.";
            }
            enum "OCL2.x" {
                description
                    "Object Constraint Language, v2.0 through 2.3.1
                     [2].";
            }
            enum "OCL1.x" {
                description
                    "Object Constraint Language, any version prior
                     to v2.0 [3].";
            }
            enum "QVT1.2R" {
                description
                    "QVT Relational Language [5].";
            }
            enum "QVT1.2O" {
                description
                    "QVT Operational language [5].";
            }
            enum "Alloy" {
                description
                    "A language for defining structures and
                     and relations using constraints [4].";
            }
        }
        description
            "The language used to encode the constraints
             relevant to the relationship between the metadata
```

```
              and the underlying policy object.";
      }
```

```
typedef policy-data-type-id-encoding-list {
    type enumeration {
        enum "error" {
            description
                "This signifies an error state.";
        }
        enum "init" {
            description
                "This signifies a generic initialization state.";
        }
        enum "primary_key" {
            description
                "This represents the primary key of a table, which
                 uniquely identifies each record in that table.
                 It MUST NOT be NULL. It MAY consist of a single
                 or multiple fields. Note that a YANG data model
                 implementation does NOT have to implement this
                 enumeration.";
        }
        enum "foreign_key" {
            description
                "This represents the foreign key, which is a set
                 or more fields in one table that uniquely
                 identify a row in another table. It MAY be
                 NULL. Note that a YANG data model implementation
                 does NOT have to implement this enumeration.";
        }
        enum "GUID" {
            description
                "The object is referenced by this GUID.";
        }
        enum "UUID" {
            description
                "The object is referenced by this UUID.";
        }
        enum "URI" {
            description
                "The object is referenced by this URI.";
        }
        enum "FQDN" {
            description
                "The object is referenced by this FQDN.";
        }
        enum "FQPN" {
            description
                "The object is referenced by this FQPN. Note that
                 FQPNs assume that all components can access a
                 single logical file repostory.";
        }
```

```
            enum "string_instance_id" {
                description
                    "A string that is the canonical representation,
                     in ASCII, of an instance ID of this object.";
            }
        }
        description
            "The list of possible data types used to represent object
             IDs for all SUPA object instances.";
    }

    typedef policy-data-type-encoding-list {
        type enumeration {
            enum "error" {
                description
                    "This signifies an error state.";
            }
            enum "init" {
                description
                    "This signifies an initialization state.";
            }
            enum "string" {
                description
                    "This represents a string data type.";
            }
            enum "integer" {
                description
                    "This represents an integer data type.";
            }
            enum "boolean" {
                description
                    "This represents a Boolean data type.";
            }
            enum "floating point" {
                description
                    "This represents a floating point data type.";
            }
            enum "date-and-time" {
                description
                    "This represents a data type that can specify
                     date and/or time.";
            }
            enum "GUID" {
                description
                    "This represents a GUID data type.";
            }
            enum "UUID" {
                description
                    "This represents a UUID data type.";
```

```
        }
```

```
            enum "URI" {
                description
                    "This represents a URI data type.";
            }
            enum "DN" {
                description
                    "This represents a DN data type.";
            }
            enum "FQDN" {
                description
                    "The object is referenced by this FQDN.";
            }
            enum "FQPN" {
                description
                    "The object is referenced by this FQPN. Note that
                     FQPNs assume that all components can access a
                     single logical file repostory.";
            }
            enum "NULL" {
                description
                    "This represents a NULL data type. NULL means the
                     absence of an actual value. NULL is frequently
                     used to represent a missing or invalid value.";
            }
            enum "string_instance_id" {
                description
                    "A string that is the canonical representation,
                     in ASCII, of an instance ID of this object.";
            }
        }
        description
            "The set of allowable data types used to encode
             multi-valued SUPA Policy attributes.";
    }

// Identities are used in this model as a means to provide simple
// introspection to allow an instance-identifier to be tested as to
// what class it represents. This allows must clauses to specify
// that the target of a particular instance-identifier leaf must be a
// specific class, or within a certain branch of the inheritance tree.
// This depends upon the ability to refine the entity class default
// value. The entity class should be read-only. However, as this is
// the target of a MUST condition, it cannot be config-false. Also,
// it appears that we cannot put a MUST condition on its definition,
// as the default (actual) value changes for each inherited object.
// Finally, note that since identities are irreflexive, we define a
// parent identitym called SUPA-ROOT-TYPE, to serve as the single root
// from which all identity statements are derived.
```

```
identity SUPA-ROOT-TYPE {
    description
        "The identity corresponding to a single root for all
         identities in the SUPA Data Model. Note that section
         7.18.2 in RFC7950 says that identity derivation is
         irreflexive (i.e., an identity cannot be derived
         from itself.";
}

identity POLICY-OBJECT-TYPE {
    base SUPA-ROOT-TYPE;
    description
        "The identity corresponding to a SUPAPolicyObject
         object instance.";
}

grouping supa-policy-object-type {
    leaf entity-class {
        type identityref {
            base SUPA-ROOT-TYPE;
        }
        default POLICY-OBJECT-TYPE;
        description
            "The identifier of the class of this grouping.";
    }
    leaf supa-policy-ID {
        type string;
        mandatory true;
        description
            "The string identifier of this policy object, which
             functions as the unique object identifier of this
             object instance. This attribute MUST be unique within
             the policy system. This attribute is named
             supaObjectIDContent in [1], and is used with another
             attribute (supaObjectIDEncoding); since the YANG data
             model does not need this genericity, the
             supaObjectIDContent attribute was renamed, and the
             supaObjectIDEncoding attribute was not mapped.";
    }
    leaf supa-policy-name {
        type string;
        description
            "A human-readable name for this policy object. Note
             that this is NOT the object ID.";
    }
```

```
        leaf supa-policy-object-description {
            type string;
            description
                "A human-readable description of the characteristics
                 and behavior of this policy object.";
        }
        leaf-list supa-has-policy-metadata-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-METADATA-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
                 reference a SUPAHasPolicyMetadata association [1].
                 This association is represented by the grouping
                 supa-has-policy-metadata-detail. This association
                 describes how each SUPAPolicyMetadata instance is
                 related to a given SUPAPolicyObject instance. Since
                 this association class contains attributes, the
                 instance-identifier MUST point to an instance using
                 the grouping supa-has-policy-metadata-detail (which
                 includes subclasses of this association class).";
        }
        description
            "This represents the SUPAPolicyObject [1] class. It is the
             superclass for all SUPA Policy objects (i.e., all objects
             that are either Policies or components of Policies). Note
             that SUPA Policy Metadata objects are NOT subclassed from
             this class; they are instead subclassed from the
             SUPAPolicyMetadata (i.e., supa-policy-metadata-type)
             object. This class (supa-policy-object-type) is used to
             define common attributes and relationships that all SUPA
             Policy subclasses inherit. It MAY be augmented with a set
             of zero or more SUPAPolicyMetadata objects using the
             SUPAHasPolicyMetadata association, which is represented
             by the supa-has-policy-metadata-agg leaf-list.";
    }

    identity POLICY-COMPONENT-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyComponentStructure object instance.";
    }

    grouping supa-policy-component-structure-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-COMPONENT-TYPE;
            }
```

```
        }
```

```
        leaf supa-has-policy-component-decorator-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC')";
            mandatory true;
            description
                "This leaf holds instance-identifiers that
                 reference a SUPAHasDecoratedPolicyComponent
                 association [1], and is represented by the grouping
                 supa-has-decorator-policy-component-detail. This
                 association describes how each
                 SUPAPolicyComponentStructure instance is related to a
                 given SUPAPolicyComponentDecorator instance. Multiple
                 SUPAPolicyComponentDecorator instances may be attached
                 to a SUPAPolicyComponentStructure instance that is
                 referenced in this association by using the Decorator
                 pattern [1]. Since this association class contains
                 attributes, the instance-identifier MUST point to an
                 instance using the grouping
                 supa-has-decorator-policy-component-detail (which
                 includes subclasses of this association class).";
        }
        description
          "This represents the SUPAPolicyComponent class [1], which is
           the superclass for all objects that represent different
           components of a Policy. Important subclasses include the
           SUPAPolicyClause and the SUPAPolicyComponentDecorator.
           This object is the root of the Decorator pattern [1]; as
           such, it enables all of its concrete subclasses to be
           wrapped with other concrete subclasses of the
           SUPAPolicyComponentDecorator class.";
    }

    identity POLICY-COMPONENT-DECORATOR-TYPE {
        base POLICY-COMPONENT-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyComponentDecorator object instance.";
    }

    grouping supa-policy-component-decorator-type {
        uses supa-policy-component-structure-type {
            refine entity-class {
                default POLICY-COMPONENT-DECORATOR-TYPE;
            }
        }
```

```
        leaf-list supa-has-policy-component-decorator-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC')";
            min-elements 1;
            description
                "This leaf-list holds instance-identifiers that
                 reference a SUPAHasDecoratedPolicyComponent
                 association [1]. This association is represented by the
                 grouping supa-has-decorator-policy-component-detail.
                 This leaf-list helps implement the Decorator pattern
                 [1], which enables all or part of one or more object
                 instances to wrap another object instance. For
                 example, any concrete subclass of SUPAPolicyClause,
                 such as SUPAEncodedClause, may be wrapped by any
                 concrete subclass of SUPAPolicyComponentDecorator
                 (e.g., SUPAPolicyEvent). Since this association class
                 contains attributes, the instance-identifier MUST
                 point to an instance using the grouping
                 supa-has-decorator-policy-component-detail (which
                 includes subclasses of this association class).";
        }
        leaf-list supa-decorator-constraints {
            type string;
            description
                "This is a set of constraint expressions that are
                 applied to this decorator, allowing the specification
                 of details not captured in its subclasses, using an
                 appropriate constraint language that is specified in
                 the supa-has-decorator-constraint-encoding leaf.";
        }
        leaf supa-has-decorator-constraint-encoding {
            type policy-constraint-language-list;
            description
                "The language in which the constraints on the
                 policy component decorator is expressed. Examples
                 include OCL 2.4 [2], Alloy [3], and English text.";
        }
        description
            "This object implements the Decorator pattern [1], which
             enables all or part of one or more concrete objects to
             wrap another concrete object.";
    }

    identity POLICY-COMPONENT-CLAUSE-TYPE {
        base POLICY-COMPONENT-TYPE;
        description
            "The identity corresponding to a SUPAPolicyClause
```

```
         object instance.";
    }
```

```
     grouping supa-policy-clause-type {
         uses supa-policy-component-structure-type {
             refine entity-class {
                 default POLICY-COMPONENT-CLAUSE-TYPE;
             }
         }
         leaf supa-policy-clause-deploy-status {
             type enumeration {
                 enum "error" {
                     description
                         "This signifies an error state. OAM&P Policies
                          SHOULD NOT use this SUPAPolicyClause if the
                          value of this attribute is error.";
                 }
                 enum "init" {
                     description
                         "This signifies an initialization state.";
                 }
                 enum "deployed and enabled" {
                     description
                         "This SUPAPolicyClause has been deployed in
                          the system and is currently enabled.";
                 }
                 enum "deployed and in test" {
                     description
                         "This SUPAPolicyClause has been deployed in the
                          system, but is currently in a test state and
                          SHOULD NOT be used in OAM&P policies.";
                 }
                 enum "deployed but not enabled" {
                     description
                         "This SUPAPolicyClause has been deployed in the
                          system, but has been administratively
                          disabled. Therefore, it MUST NOT be used in
                          OAM&P Policies.";
                 }
                 enum "ready to be deployed" {
                     description
                         "This SUPAPolicyClause has been properly
                          initialized, and is now ready to be deployed.";
                 }
                 enum "cannot be deployed" {
                     description
                         "This SUPAPolicyClause has been administratively
                          disabled, and MUST NOT be used as part of
                          an OAM&P policy.";
                 }
             }
```

```
            mandatory true;
            description
                "This defines whether this SUPAPolicy has been
                 deployed and, if so, whether it is enabled and
                 ready to be used or not.";
        }
        leaf-list supa-has-policy-clause-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-CLAUSE-ASSOC')";
            min-elements 1;
            description
                "This leaf-list holds instance-identifiers that
                 reference a SUPAHasPolicyClause association [1],
                 and is represented by the grouping
                 supa-has-policy-clause-detail. This association
                 describes how each SUPAPolicyClause instance is
                 related to this particular SUPAPolicyStructure
                 instance. For example, this association may restrict
                 which concrete subclasses of the SUPAPolicyStructure
                 class can be associated with which contrete subclasses
                 of the SUPAPolicyClause class. The set of
                 SUPAPolicyClauses, identified by this leaf-list,
                 define the content of this SUPAPolicyStructure.
                 Since this association class contains attributes, the
                 instance-identifier MUST point to an instance using
                 the grouping supa-has-policy-clause-detail (which
                 includes subclasses of this association class).";
        }
        description
           "The parent class for all SUPA Policy Clauses. A
            SUPAPolicyClause is a fundamental building block for
            creating SUPA Policies. A SUPAPolicy is a set of
            statements, and a SUPAPolicyClause can be thought of as all
            or part of a statement. The Decorator pattern [1] is used,
            which enables the contents of a SUPAPolicyClause to be
            adjusted dynamically at runtime without affecting other
            objects of either type.";
    }

    identity POLICY-ENCODED-CLAUSE-TYPE {
        base POLICY-COMPONENT-CLAUSE-TYPE;
        description
            "The identity corresponding to a SUPAEncodedClause
             object instance.";
    }
```

```
    grouping supa-encoded-clause-type {
        uses supa-policy-clause-type {
            refine entity-class {
                default POLICY-ENCODED-CLAUSE-TYPE;
            }
        }
        leaf supa-encoded-clause-content {
            type string;
            mandatory true;
            description
                "This defines the content of this SUPAEncodedClause; the
                 language used to express this content is defined by the
                 supa-encoded-clause-language attribute.";
        }
        leaf supa-encoded-clause-language {
            type enumeration {
                enum "error" {
                    description
                        "This signifies an error state. OAM&P Policies
                         SHOULD NOT use this SUPAEncodedClause if the
                         value of this attribute is error.";
                }
                enum "init" {
                    description
                        "This signifies an initialization state.";
                }
                enum "YANG" {
                    description
                        "This defines the language used in this
                         SUPAEncodedClause as a type of YANG.
                         Additional details may be provided by
                         attaching a SUPAPolicyMetadata object to
                         this SUPAEncodedClause object instance.";
                }
                enum "XML" {
                    description
                        "This defines the language as a type of XML.
                         Additional details may be provided by
                         attaching a SUPAPolicyMetadata object to
                         this SUPAEncodedClause object instance.";
                }
                enum "TL1" {
                    description
                        "This defines the language as a type of
                         Transaction Language 1. Additional details may
                         be provided by attaching a SUPAPolicyMetadata
                         object to this SUPAEncodedClause object
                         instance.";
```

```
        }
```

```
                enum "Text" {
                    description
                        "This is a textual string that can be used to
                         define a language choice that is not listed
                         by a specific enumerated value. This string
                         MUST be parsed by the policy system to
                         identify the language being used. A
                         SUPAPolicyMetadata object (represented as a
                         supa-policy-metadata-type leaf) can be used to
                         provide further details about the language";
                }
            }
            mandatory true;
            description
                "Indicates the language used for this SUPAEncodedClause
                 object instance. Prescriptive and/or descriptive
                 information about the usage of this SUPAEncodedClause
                 may be provided by one or more SUPAPolicyMetadata
                 objects, which are each attached to the object
                 instance of this SUPAEncodedClause.";
        }
        description
            "This class refines the behavior of the supa-policy-clause
             by encoding the contents of the clause into the attributes
             of this object. This enables clauses that are not based on
             other SUPA objects to be modeled. For example, a POLICY
             Application could define a CLI or YANG configuration
             snippet and encode that snipped into a SUPAEncodedClause.
             Note that a SUPAEncodedClause simply defines the content
             of the clause. In particular, it does NOT provide a
             response. The policy engine that is parsing and evaluating
             the SUPAPolicy needs to assign a response to any
             SUPAEncodedClause that it encounters.";
    }

    container supa-encoding-clause-container {
        description
            "This is a container to collect all object instances of
             type SUPAEncodedClause.";
      list supa-encoding-clause-list {
            key supa-policy-ID;
            uses supa-encoded-clause-type;
            description
                "A list of all instances of supa-encoding-clause-type.
                 If a module defines subclasses of the encoding clause,
                 those will be stored in a separate container.";
        }
    }
```

```
    identity POLICY-COMPONENT-TERM-TYPE {
        base POLICY-COMPONENT-DECORATOR-TYPE;
        description
            "The identity corresponding to a SUPAPolicyTerm object
             instance.";
    }

    grouping supa-policy-term-type {
        uses supa-policy-component-decorator-type {
            refine entity-class {
                default POLICY-COMPONENT-TERM-TYPE;
            }
        }
        leaf supa-policy-term-is-negated {
            type boolean;
            description
                "If the value of this attribute is true, then
                 this particular term is negated.";
        }
        description
           "This is the superclass of all SUPA policy objects that are
            used to test or set the value of a variable. It does this
            by defining a {variable-operator-value} three-tuple, where
            each element of the three-tuple is defined by a concrete
            subclass of the appropriate type (e.g., SUPAPolicyVariable,
            SUPAPolicyOperator, or SUPAPolicyVariable).";
    }

    identity POLICY-COMPONENT-VARIABLE-TYPE {
        base POLICY-COMPONENT-TERM-TYPE;
        description
            "The identity corresponding to a SUPAPolicyVariable
             object instance.";
    }

    grouping supa-policy-variable-type {
        uses supa-policy-term-type {
            refine entity-class {
                default POLICY-COMPONENT-VARIABLE-TYPE;
            }
        }
        leaf supa-policy-variable-name {
            type string;
            description
                "A human-readable name for this policy variable.";
        }
```

```
        description
            "This is one formulation of a SUPA Policy Clause. It uses
             the canonical form of an expression, which is a three-tuple
             in the form {variable, operator, value}. In this approach,
             each of the three terms can either be a subclass of the
             appropriate SUPAPolicyTerm class, or another object that
             plays the role (i.e., a variable) of that term. The
             attribute defined by the supa-policy-variable-name
             specifies the name of an attribute whose content should be
             compared to the value portion of a SUPAPolicyTerm, which is
             typically specified by a SUPAPolicyValue object.";
    }

    container supa-policy-variable-container {
        description
            "This is a container to collect all object instances of
              type SUPAPolicyVariable.";
        list supa-policy-variable-list {
            key supa-policy-ID;
            uses supa-policy-variable-type;
            description
                "List of all instances of supa-policy-variable-type.
                 If a module defines subclasses of this class,
                 those will be stored in a separate container.";
        }
    }

    identity POLICY-COMPONENT-OPERATOR-TYPE {
        base POLICY-COMPONENT-TERM-TYPE;
        description
            "The identity corresponding to a SUPAPolicyOperator
             object instance.";
    }

    grouping supa-policy-operator-type {
        uses supa-policy-term-type {
            refine entity-class {
                default POLICY-COMPONENT-OPERATOR-TYPE;
            }
        }
        leaf supa-policy-value-op-type {
            type enumeration {
                enum "error" {
                    description
                        "This signifies an error state.";
                }
                enum "init" {
                    description
                        "This signifies an initialization state.";
```

```
            }
```

```
                    enum "greater than" {
                        description
                            "A greater-than operator.";
                    }
                    enum "greater than or equal to" {
                        description
                            "A greater-than-or-equal-to operator.";
                    }
                    enum "less than" {
                        description
                            "A less-than operator.";
                    }
                    enum "less than or equal to" {
                        description
                            "A less-than-or-equal-to operator.";
                    }
                    enum "equal to" {
                        description
                            "An equal-to operator.";
                    }
                    enum "not equal to"{
                        description
                            "A not-equal-to operator.";
                    }
                    enum "IN" {
                        description
                            "An operator that determines whether a given
                             value of a variable in a SUPAPolicyTerm
                             matches a value in a SUPAPolicyTerm.";
                    }
                    enum "NOT IN" {
                        description
                            "An operator that determines whether a given
                             variable in a SUPAPolicyTerm does not match
                             any of the specified values in a
                             SUPAPolicyTerm.";
                    }
                    enum "SET" {
                        description
                            "An operator that makes the value of the
                             result equal to the input value.";
                    }
                    enum "CLEAR"{
                        description
                            "An operator that sets the value of the
                             specified object to a value that is 0 for
                             integer datatypes, an empty string for
                             textual datatypes, and FALSE for Boolean
                             datatypes. This value MUST NOT be NULL.";
```

```
        }
```

```
                enum "BETWEEN" {
                    description
                        "An operator that determines whether a given
                         value is within a specified range of values.
                         Note that this is an inclusive operator.";
                }
            }
            mandatory true;
            description
                "The type of operator used to compare the variable
                 and value portions of this SUPAPolicyTerm.";
        }
        description
           "This is one formulation of a SUPA Policy Clause. It uses
            the canonical form of an expression, which is a three-tuple
            in the form {variable, operator, value}. In this approach,
            each of the three terms can either be a subclass of the
            appropriate SUPAPolicyTerm class, or another object that
            plays the role (i.e., an operator) of that term.
            The value of the supa-policy-value-op-type attribute
            specifies an operator that SHOULD be used to compare the
            variable and value portions of a SUPAPolicyTerm. This is
            typically specified by a SUPAPolicyOperator object.";
    }

    container supa-policy-operator-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyOperator.";
        list supa-policy-operator-list {
            key supa-policy-ID;
            uses supa-policy-operator-type;
            description
                "List of all instances of supa-policy-operator-type.
                 If a module defines subclasses of this class,
                 those will be stored in a separate container.";
        }
    }

    identity POLICY-COMPONENT-VALUE-TYPE {
        base POLICY-COMPONENT-TERM-TYPE;
        description
            "The identity corresponding to a SUPAPolicyValue
             object instance.";
    }
```

```
grouping supa-policy-value-type {
    uses supa-policy-term-type {
        refine entity-class {
            default POLICY-COMPONENT-VALUE-TYPE;
        }
    }
    leaf-list supa-policy-value-content {
        type string;
        description
            "The content of the value portion of this SUPA Policy
             Clause. The data type of the content is specified in
             the supa-policy-value-encoding attribute.";
    }
    leaf supa-policy-value-encoding {
        type policy-data-type-encoding-list;
        description
            "The data type of the supa-policy-value-content
             attribute.";
    }
    description
       "This is one formulation of a SUPA Policy Clause. It uses
        the canonical form of an expression, which is a three-tuple
        in the form {variable, operator, value}. In this approach,
        each of the three terms can either be a subclass of the
        appropriate SUPAPolicyTerm class, or another object that
        plays the role (i.e., a value) of that term. The
        attribute defined by supa-policy-value-content specifies a
        a value (which is typically specified by a subclass of
        SUPAPolicyVariable) that should be compared to a value in
        the variable portion of the SUPAPolicyTerm.";
}

container supa-policy-value-container {
    description
        "This is a container to collect all object instances of
         type SUPAPolicyValue.";
    list supa-policy-value-list {
        key supa-policy-ID;
        uses supa-policy-value-type;
        description
            "List of all instances of supa-policy-value-type.
             If a module defines subclasses of this class,
             those will be stored in a separate container.";
    }
}

identity POLICY-GENERIC-DECORATED-TYPE {
    base POLICY-COMPONENT-DECORATOR-TYPE;
    description
```

```
        "The identity corresponding to a
         SUPAGenericDecoratedComponent object instance.";
    }
```

```
    grouping supa-policy-generic-decorated-type {
        uses supa-policy-component-decorator-type {
            refine entity-class {
                default POLICY-GENERIC-DECORATED-TYPE;
            }
        }
        leaf-list supa-policy-generic-decorated-content {
            type string;
            description
                "The content of this SUPAGenericDecoratedComponent
                 object instance. The data type of this attribute is
                 specified in the leaf
                 supa-policy-generic-decorated-encoding.";
        }
        leaf supa-policy-generic-decorated-encoding {
            type policy-data-type-encoding-list;
            description
                "The datatype of the
                 supa-policy-generic-decorated-content attribute.";
        }
        description
            "This class enables a generic object to be defined and
             used as a decorator in a SUPA Policy Clause. This class
             should not be confused with the SUPAEncodedClause class.
             A SUPAGenericDecoratedComponent object represents a single,
             atomic object that defines a portion of the contents of a
             SUPAPolicyClause, whereas a SUPAPolicyEncodedClause
             represents the entire contents of a SUPAPolicyClause.";
    }

    container supa-policy-generic-decorated-container {
        description
            "This is a container to collect all object instances of
             type SUPAGenericDecoratedComponent.";
        list supa-encoding-clause-list {
            key supa-policy-ID;
            uses supa-policy-generic-decorated-type;
            description
                "List of all instances of
                 supa-policy-generic-decorated-type. If a module
                 defines subclasses of this class, those will be
                 stored in a separate container.";
        }
    }

    identity POLICY-STRUCTURE-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAPolicyStructure
```

```
            object instance.";
     }
```

```
    grouping supa-policy-structure-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-STRUCTURE-TYPE;
            }
        }
        leaf supa-policy-admin-status {
            type enumeration {
                enum "error" {
                    description
                        "This signifies an error state. OAM&P Policies
                         SHOULD NOT use this SUPAPolicy if the value
                         of this attribute is error.";
                }
                enum "init" {
                    description
                        "This signifies an initialization state.";
                }
                enum "enabled" {
                    description
                        "This signifies that this SUPAPolicy has been
                         administratively enabled.";
                }
                enum "disabled" {
                    description
                        "This signifies that this SUPAPolicy has been
                         administratively disabled.";
                }
                enum "in test" {
                    description
                        "This signifies that this SUPAPolicy has been
                         administratively placed into test mode, and
                         SHOULD NOT be used as part of an operational
                         policy rule.";
                }
            }
            mandatory true;
            description
                "The current admnistrative status of this SUPAPolicy.";
        }
        leaf supa-policy-continuum-level {
            type uint32;
            description
                "This is the current level of abstraction of this
                 particular SUPAPolicyRule. By convention, the
                 values 0 and 1 should be used for error and
                 initialization states; a value of 2 is the most
                 abstract level, and higher values denote more
```

```
                concrete levels.";
        }
```

```
    leaf supa-policy-deploy-status {
        type enumeration {
            enum "error" {
                description
                    "This signifies an error state.";
            }
            enum "init" {
                description
                    "This signifies an initialization state.";
            }
            enum "deployed and enabled" {
                description
                    "This SUPAPolicy has been deployed in the
                     system and is currently enabled.";
            }
            enum "deployed and in test" {
                description
                    "This SUPAPolicy has been deployed in the
                     system, but is currently in test and SHOULD
                     NOT be used in OAM&P policies.";
            }
            enum "deployed but not enabled" {
                description
                    "This SUPAPolicy has been deployed in the
                     system, but has been administratively
                     disabled.";
            }
            enum "ready to be deployed" {
                description
                    "This SUPAPolicy has been properly initialized,
                     and is now ready to be deployed.";
            }
            enum "cannot be deployed" {
                description
                    "This SUPAPolicy has been administratively
                     disabled, and SHOULD NOT be used as part of
                     an OAM&P policy.";
            }
        }
        mandatory true;
        description
            "This attribute defines whether this SUPAPolicy has
             been deployed and, if so, whether it is enabled and
             ready to be used or not.";
    }
    leaf supa-policy-exec-fail-strategy {
        type enumeration {
            enum "error" {
                description
```

```
                    "This signifies an error state.";
          }
```

```
              enum "init" {
                  description
                      "This signifies an initialization state.";
              }
              enum "rollback all" {
                  description
                      "This means that execution of this SUPAPolicy
                       SHOULD be stopped, and rollback of all
                       SUPAPolicyActions (whether they were
                       successfully executed or not) performed by
                       this particular SUPAPolicy is attempted. Also,
                       all SUPAPolicies that otherwise would have
                       been executed as a result of this SUPAPolicy
                       SHOULD NOT be executed.";
              }
              enum "rollback single" {
                  description
                      "This means that execution of this SUPAPolicy
                       SHOULD be stopped, and rollback is attempted
                       for ONLY the SUPAPolicyAction (belonging to
                       this particular SUPAPolicy) that failed to
                       execute correctly. All remaining actions
                       including SUPAPolicyActions and SUPAPolicies
                       that otherwise would have been executed as a
                       result of this SUPAPolicy, SHOULD NOT
                       be executed.";
              }
              enum "stop execution" {
                  description
                      "This means that execution of this SUPAPolicy
                       SHOULD be stopped without any other action
                       being performed; this includes corrective
                       actions, such as rollback, as well as any
                       SUPAPolicyActions or SUPAPolicies that
                       otherwise would have been executed.";
              }
              enum "ignore" {
                  description
                      "This means that any failures produced by this
                       SUPAPolicy SHOULD be ignored, and hence, no
                       corrective actions, such as rollback, will
                       be performed at this time. Hence, any other
                       SUPAPolicyActions or SUPAPolicies SHOULD
                       continue to be executed.";
              }
          }
          mandatory true;
          description
```

```
                 "This defines what actions, if any, should be taken by
                  this particular SUPA Policy Rule if it fails to
                  execute correctly. Some implementations may not be
                  able to accommodate the rollback failure options;
                  hence, these options may be skipped.";
        }
        leaf-list supa-has-policy-source-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-SOURCE-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
                 reference SUPAHasPolicySource associations [1].
                 This association is represented by the grouping
                 supa-has-policy-source-detail, and describes how
                 this SUPAPolicyStructure instance is related to a
                 set of SUPAPolicySource instances. Each
                 SUPAPolicySource instance defines a set of
                 unambiguous sources of this SUPAPolicy. Since
                 this association class contains attributes, the
                 instance-identifier MUST point to an instance using
                 the grouping supa-has-policy-source-detail (which
                 includes subclasses of this association class).";
        }
        leaf-list supa-has-policy-target-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-TARGET-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
                 reference SUPAHasPolicyTarget associations [1].
                 This association is represented by the grouping
                 supa-has-policy-target-detail, and describes how
                 this SUPAPolicyStructure instance is related to a
                 set of SUPAPolicyTarget instances. Each
                 SUPAPolicyTarget instance defines a set of
                 unambiguous managed entities to which this
                 SUPAPolicy will be applied to. Since this association
                 class contains attributes, the instance-identifier
                 MUST point to an instance using the grouping
                 supa-has-policy-target-detail (which includes
                 subclasses of this association class).";
        }
        leaf-list supa-has-policy-clause-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-CLAUSE-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
```

reference SUPAHasPolicyClause associations [1]. This
association is represented by the grouping
supa-has-policy-clause-detail. This association

```
                describes how this particular SUPAPolicyStructure
                instance is related to this set of SUPAPolicyClause
                instances. Since this association class contains
                attributes, the instance-identifier MUST point to an
                instance using the supa-has-policy-clause-detail
                (which includes subclasses of this association
                class).";
        }
        leaf-list supa-has-policy-exec-fail-action-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-EXEC-ACTION-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
                reference a SUPAHasPolExecFailtActionToTake
                association [1]. This association is represented by
                the supa-has-policy-exec-action-detail grouping. This
                association relates this SUPAPolicyStructure instance
                (the parent) to one or more SUPAPolicyStructure
                instances (the children), where each child
                SUPAPolicyStructure contains one or more
                SUPAPolicyActions to be executed if the parent
                SUPAPolicyStructure instance generates an error while
                it is executing. Since this association class contains
                attributes, the instance-identifier MUST point to an
                instance using the grouping
                supa-has-policy-exec-action-detail (which includes
                subclasses of this association class).";
        }
        leaf-list supa-has-policy-exec-fail-action-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                 'SUPA-HAS-POLICY-EXEC-ACTION-ASSOC')";
            min-elements 1;
            description
                "This leaf-list holds instance-identifiers that
                reference a SUPAHasPolExecFailtActionToTake
                association [1]. This association is represented by
                the supa-has-policy-exec-action-detail grouping. This
                association relates this SUPAPolicyStructure instance
                (the child) to another SUPAPolicyStructure instance
                (the parent). The child SUPAPolicyStructure contains
                one or more SUPAPolicyActions to be executed if the
                parent SUPAPolicyStructure instance generates an error
                while it is executing; the parent SUPAPolicyStructure
                contains one or more child SUPAPolicyStructure
                instances to enable it to choose how to handle each
                type of failure. Since this association class contains
                attributes, the instance-identifier MUST point to an
```

```
                    instance using the grouping
                    supa-has-policy-exec-action-detail (which includes
                    subclasses of this association class).";
        }
        description
           "A superclass for all objects that represent different types
            of SUPAPolicies. Currently, this is limited to a single
            type, which is the event-condition-action (ECA) Policy
            Rule. A SUPA Policy may be an individual policy, or a set
            of policies. Subclasses MAY support this feature by
            implementing the composite pattern.";
    }

    identity POLICY-SOURCE-TYPE {
        base POLICY-OBJECT-TYPE;
        description
           "The identity corresponding to a SUPAPolicySource
            object instance.";
    }

    grouping supa-policy-source-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-SOURCE-TYPE;
            }
        }
        leaf-list supa-has-policy-source-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'SUPA-HAS-POLICY-SOURCE-ASSOC')";
            description
                "This leaf-list holds the instance-identifiers that
                 reference a SUPAHasPolicySource association [1], which
                 is represented by the supa-has-policy-source-detail
                 grouping. This association describes how each
                 SUPAPolicySource instance is related to this
                 particular SUPAPolicyStructure instance. Since
                 this association class contains attributes, the
                 instance-identifier MUST point to an instance using
                 the grouping supa-has-policy-source-detail (which
                 includes subclasses of this association class).";
        }
        description
           "This object defines a set of managed entities that
            authored, or are otherwise responsible for, this
            SUPAPolicy. Note that a SUPAPolicySource does NOT evaluate
            or execute SUPAPolicies. Its primary use is for
            auditability and the implementation of deontic logic (i.e.,
            how concepts such as obligation and permission work) and/or
```

alethic logic (i.e., how concepts such as necessity,
possibility, and contigency work). It is expected that this

```
            grouping will be extended (i.e., subclassed) when used, so
            that the system an add specific information appropriate to
            sources of policy of that particular system.";
    }


    container supa-policy-source-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicySource.";
        list supa-policy-source-list {
            key supa-policy-ID;
            uses supa-policy-source-type;
            description
                "A list of all supa-policy-source instances in the
                 system.";
        }
    }


    identity POLICY-TARGET-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAPolicyTarget
             object instance.";
    }


    grouping supa-policy-target-type {
        uses supa-policy-object-type {
            refine entity-class {
                default POLICY-TARGET-TYPE;
            }
        }
        leaf-list supa-has-policy-target-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'SUPA-HAS-POLICY-TARGET-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
                 reference a SUPAHasPolicyTarget association. This is
                 represented by the supa-has-policy-target-detail
                 grouping. This association describes how each
                 SUPAPolicyTarget instance is related to a particular
                 SUPAPolicyStructure instance. For example, this
                 association may restrict which SUPAPolicyTarget
                 instances can be used by which SUPAPolicyStructure
                 instances. The SUPAPolicyTarget defines a
                 set of managed entities that this SUPAPolicyStructure
                 will be applied to. Since this association class
                 contains attributes, the instance-identifier MUST
                 point to an instance using the grouping
```

```
              supa-has-policy-target-detail (which
              includes subclasses of this association class).";
       }
```

```
        description
           "This object defines a set of managed entities that a
            SUPAPolicy is applied to.  It is expected that this
            grouping will be extended (i.e., subclassed) when used,
            so that the system can add specific information
            appropriate to policy targets of that particular system.";
    }

    container supa-policy-target-container {
        description
           "This is a container to collect all object instances of
            type SUPAPolicyTarget.";
        list supa-policy-target-list {
            key supa-policy-ID;
            uses supa-policy-target-type;
            description
                "A list of all supa-policy-target instances in the
                 system.";
        }
    }

    identity POLICY-METADATA-TYPE {
        base SUPA-ROOT-TYPE;
        description
           "The identity corresponding to a SUPAPolicyMetadata
            object instance.";
    }

    grouping supa-policy-metadata-type {
        leaf entity-class {
            type identityref {
                base SUPA-ROOT-TYPE;
            }
            description
               "The identifier of the class of this grouping.";
        }
        leaf supa-policy-metadata-id {
            type string;
            mandatory true;
            description
                "This represents the object identifier of an instance
                 of this class. This attribute is named
                 supaPolMetadataIDContent in [1], and is used with
                 another attribute (supaPolMetadataIDEncoding); since
                 the YANG data model does not need this genericity, the
                 supaPolMetadataIDContent attribute was renamed, and
                 the supaPolMetadataIDEncoding attribute was
                 not mapped.";
        }
```

```
        leaf supa-policy-metadata-description {
            type string;
            description
                "This contains a free-form textual description of this
                 metadata object (e.g., what it may be used for).";
        }
        leaf supa-policy-metadata-name {
            type string;
            description
                "This contains a human-readable name for this
                 metadata object.";
        }
        leaf-list supa-has-policy-metadata-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'SUPA-HAS-POLICY-METADATA-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
                 reference a SUPAHasPolicyMetadata association [1],
                 which is represented by the grouping
                 supa-has-policy-metadata-detail. Each instance-
                 identifier defines a unique set of information that
                 describe and/or prescribe additional information,
                 provided by this SUPAPolicyMetadata instance, that can
                 be associated with this SUPAPolicyObject instance.
                 Multiple SUPAPolicyMetadata objects may be attached to
                 a concrete subclass of the SUPAPolicyObject class that
                 is referenced in this association by using the
                 Decorator pattern [1]. For example, a
                 SUPAPolicyVersionMetadataDef instance could wrap a
                 SUPAECAPolicyRuleAtomic instance; this would define
                 the version of this particular SUPAECAPolicyRuleAtomic
                 instance. Since this association class contains
                 attributes, the instance-identifier MUST point to an
                 instance using the grouping
                 supa-has-policy-metadata-detail (which includes
                 subclasses of this association class).";
        }
        leaf-list supa-has-policy-metadata-dec-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'SUPA-HAS-POLICY-METADATA-DECORATOR-DETAIL-ASSOC')";
            min-elements 1;
            description
                "This leaf-list holds instance-identifiers that
                 reference a SUPAHasMetadaDecorator association [1].
                 This association is represented by the grouping
                 supa-has-policy-metadata-dec-detail. This association
                 describes how a SUPAPolicyMetadataDecorator instance
```

wraps a given SUPAPolicyMetadata instance using the
Decorator pattern [1]. Multiple concrete subclasses

```
                    of SUPAPolicyMetadataDecorator may be used to wrap
                    the same SUPAPolicyMetadata instance. Since this
                    association class contains attributes, the
                    instance-identifier MUST point to an instance using
                    the grouping supa-has-policy-metadata-dec-detail (which
                    includes subclasses of this association class).";
        }
        description
           "This is the superclass of all metadata classes. Metadata
            is information that describes and/or prescribes the
            characteristics and behavior of another object that is
            not an inherent, distinguishing characteristics or
            behavior of that object.";
    }

    identity POLICY-METADATA-CONCRETE-TYPE {
        base POLICY-METADATA-TYPE;
        description
            "The identity corresponding to a SUPAPolicyConcreteMetadata
             object instance.";
    }

    grouping supa-policy-concrete-metadata-type {
        uses supa-policy-metadata-type {
            refine entity-class {
                default POLICY-METADATA-CONCRETE-TYPE;
            }
        }
        leaf supa-policy-metadata-valid-period-end {
            type yang:date-and-time;
            description
                "This defines the ending date and time that this
                 metadata object is valid for.";
        }
        leaf supa-policy-metadata-valid-period-start {
            type yang:date-and-time;
            description
                "This defines the starting date and time that this
                 metadata object is valid for.";
        }
        description
           "This is a concrete class that will be wrapped by concrete
            instances of the SUPA Policy Metadata Decorator class. It
            can be viewed as a container for metadata that will be
            attached to a subclass of SUPA Policy Object. It may
            contain all or part of one or more metadata subclasses.";
    }
```

```
    container supa-policy-concrete-metadata-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyConcreteMetadata.";
        list supa-policy-concrete-metadata-list {
            key supa-policy-metadata-id;
            uses supa-policy-concrete-metadata-type;
            description
                "A list of all supa-policy-metadata instances in the
                 system.";
        }
    }

    identity POLICY-METADATA-DECORATOR-TYPE {
        base POLICY-METADATA-TYPE;
        description
            "The identity corresponding to a
             SUPAPolicyMetadataDecorator object instance.";
    }

    grouping supa-policy-metadata-decorator-type {
        uses supa-policy-metadata-type {
            refine entity-class {
                default POLICY-METADATA-DECORATOR-TYPE;
            }
        }
        leaf supa-has-policy-metadata-dec-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'SUPA-HAS-POLICY-METADATA-DECORATOR-DETAIL-ASSOC')";
            description
                "This leaf-list holds instance-identifiers that
                 reference a SUPAHasMetadaDecorator association [1].
                 This association is represented by the grouping
                 supa-has-policy-metadata-dec-detail. This association
                 describes how a SUPAPolicyMetadataDecorator instance
                 wraps a given SUPAPolicyMetadata instance
                 using the Decorator pattern [1]. Multiple concrete
                 subclasses of SUPAPolicyMetadataDecorator may be used
                 to wrap the same SUPAPolicyMetadata instance. Since
                 this association class contains attributes, the
                 instance-identifier MUST point to an instance using
                 the grouping supa-has-policy-metadata-dec-detail (which
                 includes subclasses of this association class).";
        }
        description
            "This object implements the Decorator pattern [1] for all
             SUPA metadata objects. This enables all or part of one or
             more metadata objects to wrap another concrete metadata
```

```
        object. The only concrete subclass of SUPAPolicyMetadata
        in this document is SUPAPolicyConcreteMetadata.";
    }
```

```
identity POLICY-METADATA-DECORATOR-ACCESS-TYPE {
    base POLICY-METADATA-DECORATOR-TYPE;
    description
        "The identity corresponding to a
         SUPAPolicyAccessMetadataDef object instance.";
}

grouping supa-policy-metadata-decorator-access-type {
    uses supa-policy-metadata-decorator-type {
        refine entity-class {
            default POLICY-METADATA-DECORATOR-ACCESS-TYPE;
        }
    }
    leaf supa-policy-metadata-access-priv-def {
        type enumeration {
            enum "error" {
                description
                    "This signifies an error state. OAM&P Policies
                     SHOULD NOT use this SUPAPolicyAccessMetadataDef
                     if the value of this attribute is error.";
            }
            enum "init" {
                description
                    "This signifies an initialization state.";
            }
            enum "read only" {
                description
                    "This defines access as read only for ALL
                     SUPAPolicyObject objects that are adorned
                     with this SUPAPolicyAccessMetadataDef object.
                     As such, an explicit access control model,
                     such as RBAC [7], is NOT present.";
            }
            enum "read write" {
                description
                    "This defines access as read and/or write for
                     ALL SUPAPolicyObject objects that are adorned
                     with this SUPAPolicyAccessMetadataDef object.
                     As such, an explicit access control model,
                     such as RBAC [7], is NOT present.";
            }
            enum "specified by MAC" {
                description
                    "This uses an external Mandatory Access Control
                     (MAC) [7] model to define access control for
                     ALL SUPAPolicyObject objects that are adorned
                     with this SUPAPolicyAccessMetadataDef object.
                     The name and location of this access control
```

model are specified, respectively, in the

```
                            supa-policy-metadata-access-priv-model-name
                            and supa-policy-metadata-access-priv-model-ref
                            attributes of this SUPAPolicyAccessMetadataDef
                            object.";
                    }
                    enum "specified by DAC" {
                        description
                            "This uses an external Discretionary Access
                            Control (DAC) [7] model to define access
                            control for ALL SUPAPolicyObject objects that
                            are adorned with this
                            SUPAPolicyAccessMetadataDef object. The name
                            and location of this access control model are
                            specified, respectively, in the
                            supa-policy-metadata-access-priv-model-name
                            and supa-policy-metadata-access-priv-model-ref
                            attributes of this SUPAPolicyAccessMetadataDef
                            object.";
                    }
                    enum "specified by RBAC" {
                        description
                            "This uses an external Role-Based Access Control
                            (RBAC) [7] model to define access control for
                            ALL SUPAPolicyObject objects that are adorned
                            with this SUPAPolicyAccessMetadataDef object.
                            The name and location of this access control
                            model are specified, respectively, in the
                            supa-policy-metadata-access-priv-model-name
                            and supa-policy-metadata-access-priv-model-ref
                            attributes of this SUPAPolicyAccessMetadataDef
                            object.";
                    }
                    enum "specified by ABAC" {
                        description
                            "This uses an external Attribute-Based Access
                            Control (ABAC) [8] model to define access
                            control for ALL SUPAPolicyObject objects that
                            are adorned with this
                            SUPAPolicyAccessMetadataDef object. The name
                            and location of this access control model are
                            specified, respectively, in the
                            supa-policy-metadata-access-priv-model-name
                            and supa-policy-metadata-access-priv-model-ref
                            attributes of this SUPAPolicyAccessMetadataDef
                            object.";
                    }
                    enum "specified by custom" {
                        description
                            "This uses an external Custom Access Control
```

model to define access control for ALL
SUPAPolicyObject objects that are adorned
with this SUPAPolicyAccessMetadataDef object.

```
                        The name and location of this access control
                        model are specified, respectively, in the
                        supa-policy-metadata-access-priv-model-name
                        and supa-policy-metadata-access-priv-model-ref
                        attributes of this SUPAPolicyAccessMetadataDef
                        object.";
            }
        }
        description
            "This defines the type of access control model that is
             used by this SUPAPolicyObject object instance.";
    }
    leaf supa-policy-metadata-access-priv-model-name {
        type string;
        description
            "This contains the name of the access control model
             being used. If the value of the
             supa-policy-metadata-access-priv-model-ref is
             error, then this SUPAPolicyAccessMetadataDef object
             MUST NOT be used. If the value of the
             supa-policy-metadata-access-priv-model-ref is init,
             then this SUPAPolicyAccessMetadataDef object has been
             properly initialized, and is ready to be used. If the
             value of the supa-policy-metadata-access-priv-model-ref
             is read only or read write, then the value of this
             attribute is not applicable (because a type of model
             is NOT being defined; instead, the access control for
             all SUPAPolicyObjects is being defined). Otherwise,
             the text in this class attribute SHOULD be interpreted
             according to the value of the
             supa-policy-metadata-access-priv-model-ref class
             attribute.";
    }
    leaf supa-policy-metadata-access-priv-model-ref {
        type enumeration {
            enum "error" {
                description
                    "This signifies an error state. OAM&P Policies
                     SHOULD NOT use this SUPAPolicyAccessMetadataDef
                     object if the value of this attribute is
                     error.";
            }
            enum "init" {
                description
                    "This signifies an initialization state.";
            }
            enum "URI"  {
                description
                    "The access control model is referenced by
```

```
                    this URI.";
        }
```

```
                enum "GUID"  {
                    description
                        "The access control model is referenced by
                         this GUID.";
                }
                enum "UUID"  {
                    description
                        "The access control model is referenced by
                         this UUID.";
                }
                enum "FQDN"  {
                    description
                        "The access control model is referenced by
                         this FQDN.";
                }
                enum "FQPN"  {
                    description
                        "The access control model is referenced by
                         this FQPN.";
                }
                enum "string_instance_id" {
                    description
                        "A string that is the canonical representation,
                         in ASCII, of an instance ID of this object.";
                }
            }
            description
                "This defines the data type of the
                 supa-policy-metadata-access-priv-model-name
                 attribute.";
        }
        description
            "This is a concrete class that defines metadata for access
             control information that can be added to any
             SUPAPolicyObject object instance.
             This is done using the SUPAHasPolicyMetadata association
             in conjunction with the Decorator pattern [1].";
    }

    container supa-policy-metadata-decorator-access-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyAccessMetadataDef.";
        list supa-policy-metadata-decorator-access-list {
            key supa-policy-metadata-id;
            uses supa-policy-metadata-decorator-type;
            description
                "A list of all supa-policy-metadata-decorator-access
                 instances in the system.  Instances of subclasses
```

```
            will be in a separate list.";
        }
    }
```

```
identity POLICY-METADATA-DECORATOR-VERSION-TYPE {
    base POLICY-METADATA-DECORATOR-TYPE;
    description
        "The identity corresponding to a
         SUPAPolicyVersionMetadataDef object instance.";
}

grouping supa-policy-metadata-decorator-version-type {
    uses supa-policy-metadata-decorator-type {
        refine entity-class {
            default POLICY-METADATA-DECORATOR-VERSION-TYPE;
        }
    }
    leaf supa-policy-metadata-version-major {
        type string;
        description
            "This contains a string representation of an integer
             that is greater than or equal to zero. It indicates
             that a significant increase in functionality is present
             in this version. It MAY also indicate that this version
             has changes that are NOT backwards-compatible (the
             supa-policy-metadata-version-build class attribute is
             used to denote such changes). The string 0.1.0
             defines an initial version that MUST NOT be considered
             stable. Improvements to this initial version are
             denoted by incrementing the minor and patch class
             attributes (supa-policy-metadata-version-major and
             supa-policy-metadata-version-patch, respectively). The
             major version X (i.e., X.y.z, where X > 0) MUST be
             incremented if any backwards-incompatible changes are
             introduced. It MAY include minor and patch level
             changes. The minor and patch version numbers MUST be
             reset to 0 when the major version number is
             incremented.";
    }
    leaf supa-policy-metadata-version-minor {
        type string;
        description
            "This contains a string representation of an integer
             that is greater than or equal to zero. It indicates
             that this release contains a set of features and/or
             bug fixes that MUST be backwards-compatible. The
             minor version Y (i.e., x.Y.z, where x > 0) MUST be
             incremented if new, backwards-compatible changes are
             introduced. It MUST be incremented if any features are
             marked as deprecated. It MAY be incremented if new
             functionality or improvements are introduced, and MAY
             include patch level changes. The patch version number
             MUST be reset to 0 when the minor version number is
```

```
            incremented.";
        }
```

```
        leaf supa-policy-metadata-version-patch {
            type string;
            description
                "This contains a string representation of an integer
                 that is greater than or equal to zero. It indicates
                 that this version contains ONLY bug fixes. The patch
                 version Z (i.e., x.y.Z, where x > 0) MUST be
                 incremented if new, backwards-compatible changes are
                 introduced. A bug fix is defined as an internal change
                 that fixes incorrect behavior.";
        }
        leaf supa-policy-metadata-version-prerelease {
            type string;
            description
                "This contains a string that defines the pre-release
                 version. A pre-release version MAY be denoted by
                 appending a hyphen and a series of dot-separated
                 identifiers immediately following the patch version.
                 Identifiers MUST comprise only ASCII alphanumerics and
                 a hyphen. Identifiers MUST NOT be empty. Numeric
                 identifiers MUST NOT include leading zeroes.
                 Pre-release versions have a lower precedence than the
                 associated normal version. A pre-release version
                 indicates that the version is unstable and might not
                 satisfy the intended compatibility requirements as
                 denoted by its associated normal version. Examples
                 include: 1.0.0-alpha and 1.0.0-0.3.7.";
        }
        leaf supa-policy-metadata-version-build {
            type string;
            description
                "This contains a string that defines the metadata of
                 this build. Build metadata is optional. If present,
                 build metadata MAY be denoted by appending a plus
                 (+) sign to the version, followed by a series of
                 dot-separated identifiers. This may follow either
                 the patch or pre-release portions of the version.
                 If build metadata is present, then any identifiers
                 that it uses MUST be made up of only ASCII
                 alphanumerics and a hyphen. The identifier portion of
                 the build metadata MUST NOT be empty. Build metadata
                 SHOULD be ignored when determining version precedence.
                 Examples include: 1.0.0.-alpha+1, 1.0.0.-alpha+1.1,
                 1.0.0+20130313144700, and 1.0.0-beta+exp.sha.5114f85.";
        }
        description
            "This is a concrete class that defines metadata for version
             control information that can be added to any
```

SUPAPolicyObject. This is done using the
SUPAHasPolicyMetadata association. This class uses the
Semantic Versioning Specification [6] as follows:

```
                <major>.<minor>.<patch>[<pre-release>][<build-metadata>]
            where the first three components (major, minor, and patch)
            MUST be present, and the latter two components (pre-release
            and build-metadata) MAY be present. A version number MUST
            take the form <major>.<minor>.<patch>, where <major>,
            <minor>, and <patch> are each non-negative integers that
            MUST NOT contain leading zeros. In addition, the value of
            each of these three elements MUST increase numerically.
            In this approach, supaVersionMajor denotes a new release;
            supaVersionMinor denotes a minor release; supaVersionPatch
            denotes a version that consists ONLY of bug fixes. Version
            precedence MUST be calculated by separating the version
            into major, minor, patch, and pre-release identifiers, in
            that order. See [1] for more information.";
    }

    container supa-policy-metadata-decorator-version-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyVersionMetadataDef.";
        list supa-policy-metadata-decorator-version-list {
            key supa-policy-metadata-id;
            uses supa-policy-metadata-decorator-type;
            description
                "A list of all supa-policy-metadata-decorator-version
                 instances in the system. Instances of subclasses
                 will be in a separate list.";
        }
    }

    identity SUPA-HAS-POLICY-METADATA-DECORATOR-TYPE {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a
             SUPAHasPolicyMetadataDetail association class
             object instance.";
    }

    grouping supa-has-policy-metadata-detail {
        uses supa-policy-object-type {
            refine entity-class {
                default SUPA-HAS-POLICY-METADATA-DECORATOR-TYPE;
            }
        }
        leaf supa-has-policy-metadata-detail-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'POLICY-OBJECT-TYPE')";
            description
```

"This leaf is an instance-identifier that references
                the SUPAPolicyObject instance end point of the
                association represented by this instance of the

```
            SUPAHasPolicyMetadata association [1]. The groupings
            supa-policy-object-type and supa-policy-metadata-type
            represent the SUPAPolicyObject and SUPAPolicyMetadata
            classes, respectively. Thus, the instance identified
            by this leaf is the SUPAPolicyObject instance that is
            associated by this association to the set of
            SUPAPolicyMetadata instances referenced by the
            supa-has-policy-metadata-detail-part-ptr leaf of
            this grouping.";
    }
    leaf supa-has-policy-metadata-detail-part-ptr {
        type instance-identifier;
        must "derived-from-or-self (deref(.)/entity-class,
             'POLICY-METADATA-TYPE')";
        description
            "This leaf is an instance-identifier that references
            the SUPAPolicyMetadata instance end point of the
            association represented by this instance of the
            SUPAHasPolicyMetadata association [1]. The groupings
            supa-policy-object-type and supa-policy-metadata-type
            represents the SUPAPolicyObject and SUPAPolicyMetadata
            classes, respectively. Thus, the instance
            identified by this leaf is the SUPAPolicyMetadata
            instance that is associated by this association to
            the set of SUPAPolicyObject instances referenced by
            the supa-has-policy-metadata-detail-agg-ptr leaf of
            this grouping.";
    }

    leaf supa-policy-metadata-detail-is-applicable {
        type boolean;
        description
            "This attribute controls whether the associated
            metadata is currently considered applicable to this
            SUPAPolicyObject; this enables metadata to be turned
            on and off when needed without disturbing the
            structure of the object that the metadata applies to,
            or affecting other objects in the system.";
    }
    leaf-list supa-policy-metadata-detail-constraint {
        type string;
        description
            "A list of constraints, expressed as strings, in
            the language defined by the
            supa-policy-metadata-detail-encoding attribute.
            If there are no constraints on using this
            SUPAPolicyMetadata object with this particular
            SUPAPolicyObject object, then this leaf-list will
            consist of a list of a single null string.";
```

```
        }
```

```
        leaf supa-policy-metadata-detail-constraint-encoding {
            type policy-constraint-language-list;
            description
                "The language used to encode the constraints relevant
                 to the relationship between the SUPAPolicyMetadata
                 object and the underlying SUPAPolicyObject.";
        }
        description
            "This is a concrete association class that defines the
             semantics of the SUPAHasPolicyMetadata association. This
             enables the attributes and relationships of the
             SUPAHasPolicyMetadataDetail class to be used to constrain
             which SUPAPolicyMetadata objects can be associated by
             this particular SUPAPolicyObject instance.";
    }

    container supa-policy-metadata-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyMetadataDetail.";
        list supa-policy-metadata-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-metadata-detail;
            description
                "This is a list of all supa-policy-metadata-detail
                 instances in the system.  Instances of subclasses
                 will be in a separate list. Note that this association
                 class is made concrete for exemplary purposes.  To be
                 useful, it almost certainly needs refinement.";
        }
    }

    identity SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC {
        base POLICY-COMPONENT-TYPE;
        description
            "The identity corresponding to a
             SUPAHasDecoratedPolicyComponentDetail association class
             object instance.";
    }

    grouping supa-has-decorator-policy-component-detail {
        uses supa-policy-object-type {
            refine entity-class {
                default SUPA-HAS-POLICY-COMPONENT-DECORATOR-ASSOC;
            }
        }
        leaf supa-has-policy-component-decorator-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
```

```
         'POLICY-COMPONENT-DECORATOR-TYPE')";
    description
```

```
                    "This leaf is an instance-identifier that references
                     the SUPAPolicyComponentDecorator instance end point of
                     the association represented by this instance of the
                     SUPAHasDecoratedPolicyComponent association [1]. The
                     groupings supa-policy-component-decorator-type and
                     supa-policy-component-structure-type represent the
                     SUPAPolicyComponentDecorator and
                     SUPAPolicyComponentStructure classes, respectively.
                     Thus, the instance identified by this leaf is the
                     SUPAPolicyComponentDecorator instance that is
                     associated by this association to the set of
                     SUPAPolicyComponentStructure instances referenced by
                     the supa-has-policy-component-decorator-part-ptr leaf
                     of this grouping.";
        }
        leaf supa-has-policy-component-decorator-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'POLICY-COMPONENT-TYPE')";
            description
                "This leaf is an instance-identifier that references
                 the SUPAPolicyComponentStructure instance end point of
                 the association represented by this instance of the
                 SUPAHasDecoratedPolicyComponent association [1]. The
                 groupings supa-policy-component-decorator-type and
                 supa-policy-component-structure-type represent the
                 SUPAPolicyComponentDecorator and
                 SUPAPolicyComponentStructure classes, respectively.
                 Thus, the instance identified by this leaf is the
                 SUPAPolicyComponentStructure instance that is
                 associated by this association to the set of
                 SUPAPolicyComponentStructure instances referenced by
                 the supa-has-policy-component-decorator-agg-ptr leaf
                 of this grouping.";
        }
        leaf-list supa-has-decorator-constraint {
            type string;
            description
                "A constraint expression applying to this association
                 between a SUPAPolicyComponentDecorator and the
                 decorated component (which is a concrete subclass of
                 the SUPAPolicyComponentStructure class, such as
                 SUPAEncodedClause or SUPABooleanClauseAtomic). The
                 supa-has-decorator-constraint-encoding attribute
                 specifies the language used to write the set of
                 constraint expressions.";
        }
```

```
        leaf supa-has-decorator-constraint-encoding {
            type policy-constraint-language-list;
            description
                "The language used to encode the constraints relevant
                 to the relationship between the
                 SUPAPolicyComponentDecorator and the
                 SUPAPolicyComponentStructure object instances.";
        }
        description
            "This is a concrete association class that defines the
             semantics of the SUPAHasDecoratedPolicyComponent
             association. The purpose of this class is to use the
             Decorator pattern [1] to detemine which
             SUPAPolicyComponentDecorator object instances, if any,
             are required to augment the functionality of a concrete
             subclass of SUPAPolicyClause that is being used.";
    }

    container supa-policy-component-decorator-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicyComponentDecoratorDetail.";
        list supa-policy-component-decorator-detail-list {
            key supa-policy-ID;
            uses supa-has-decorator-policy-component-detail;
            description
                "This is a list of all
                 supa-policy-component-decorator-details.";
        }
    }

    identity SUPA-HAS-POLICY-SOURCE-ASSOC {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAHasPolicySource
             association class object instance.";
    }

    grouping supa-has-policy-source-detail {
        uses supa-policy-object-type {
            refine entity-class {
                default SUPA-HAS-POLICY-SOURCE-ASSOC;
            }
        }
        leaf supa-has-policy-source-detail-agg-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                  'POLICY-STRUCTURE-TYPE')";
            description
```

"This leaf is an instance-identifier that references
          a SUPAPolicyStructure instance end point of the
          association represented by this instance of the

```
                SUPAHasPolicySource association [1]. The grouping
                supa-has-policy-source-detail represents the
                SUPAHasPolicySourceDetail class. Thus, the instance
                identified by this leaf is the SUPAPolicyStructure
                instance that is associated by this association to the
                SUPAPolicySource instance referenced by the
                supa-has-policy-source-detail-part-ptr leaf of
                this grouping.";
        }
        leaf supa-has-policy-source-detail-part-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                   'POLICY-SOURCE-TYPE')";
            description
                "This leaf is an instance-identifier that references
                 a SUPAPolicySource instance end point of the
                 association represented by this instance of the
                 SUPAHasPolicySource association [1]. The grouping
                 supa-has-policy-source-detail represents the
                 SUPAHasPolicySourceDetail class. Thus, the instance
                 identified by this leaf is the SUPAPolicySource
                 instance that is associated by this association to the
                 SUPAPolicyStructure instance referenced by the
                 supa-has-policy-source-detail-agg-ptr leaf of
                 this grouping.";
        }
        leaf supa-policy-source-is-authenticated {
            type boolean;
            description
                "If the value of this attribute is true, then this
                 SUPAPolicySource object has been authenticated by
                 a policy engine or application that is executing this
                 particular SUPAPolicyStructure object.";
        }
        leaf supa-policy-source-is-trusted {
            type boolean;
            description
                "If the value of this attribute is true, then this
                 SUPAPolicySource object has been verified to be
                 trusted by a policy engine or application that is
                 executing this particular SUPAPolicyStructure object.";
        }
        description
            "This is an association class, and defines the semantics of
             the SUPAHasPolicySource association. The attributes and
             relationships of this class can be used to define which
             SUPAPolicySource objects can be attached to which
             particular set of SUPAPolicyStructure objects. Note that a
             SUPAPolicySource object is NOT responsible for evaluating
```

or executing SUPAPolicies; rather, it identifies the set
of entities that are responsible for managing this

```
            SUPAPolicySource object. Its primary uses are for
            auditability, as well as processing deontic logic. This
            object represents the semantics of associating a
            SUPAPolicySource to a SUPAPolicyTarget.";
    }

    container supa-policy-source-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolicySourceDetail.";
        list supa-policy-source-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-source-detail;
            description
                "This is a list of all supa-policy-source-detail
                 objects.";
        }
    }

    identity SUPA-HAS-POLICY-TARGET-ASSOC {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAHasPolicyTarget
             association class object instance.";
    }

    grouping supa-has-policy-target-detail {
        uses supa-policy-object-type {
            refine entity-class {
                default SUPA-HAS-POLICY-TARGET-ASSOC;
            }
        }
        leaf supa-has-policy-target-detail-agg-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                    'POLICY-STRUCTURE-TYPE')";
            description
                "This leaf is an instance-identifier that references
                 a SUPAPolicyStructure instance end point of the
                 association represented by this instance of the
                 SUPAHasPolicyTarget association [1]. The grouping
                 supa-has-policy-target-detail represents the
                 SUPAHasPolicyTargetDetail class. Thus, the instance
                 identified by this leaf is the SUPAPolicyStructure
                 instance that is associated by this association to the
                 SUPAPolicyTarget instance referenced by the
                 supa-has-policy-target-detail-part-ptr leaf of
                 this grouping.";
        }
```

```
        leaf supa-has-policy-target-detail-part-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                    'POLICY-TARGET-TYPE')";
            description
                "This leaf is an instance-identifier that references
                 a SUPAPolicyTarget instance end point of the
                 association represented by this instance of the
                 SUPAHasPolicyTarget association [1]. The grouping
                 supa-has-policy-target-detail represents the
                 SUPAHasPolicyTargetDetail class. Thus, the instance
                 identified by this leaf is the SUPAPolicyTarget
                 instance that is associated by this association to the
                 SUPAPolicyStructure instance referenced by the
                 supa-has-policy-source-detail-agg-ptr leaf of
                 this grouping.";
        }
        leaf supa-policy-target-is-authenticated {
            type boolean;
            description
                "If the value of this attribute is true, then this
                 SUPAPolicyTarget object has been authenticated by
                 a policy engine or application that is executing this
                 particular SUPAPolicyStructure object.";
        }
        leaf supa-policy-target-is-enabled {
            type boolean;
            description
                "If the value of this attribute is true, then each
                 SUPAPolicyTarget object that is referenced by this
                 SUPAHasPolicyTarget aggregation is able to be used as
                 a SUPAPolicyTarget by the SUPAPolicyStructure object
                 that is referenced by this SUPAHasPolicyTarget
                 aggregation. This means that this SUPAPolicyTarget has
                 agreed to: 1) have SUPAPolicies applied to it, and 2)
                 process (directly or with the aid of a proxy) one or
                 more SUPAPolicies, or receive the results of a
                 processed SUPAPolicy and apply those results to
                 itself.";
        }
        description
            "This is an association class, and defines the semantics of
             the SUPAHasPolicyTarget association. The attributes and
             relationships of this class can be used to define which
             SUPAPolicyTarget objects can be attached to which
             particular set of SUPAPolicyStructure objects. Note that a
             SUPAPolicyTarget is used to identify a set of managed
             entities to which a SUPAPolicy should be applied; this
             object represents the semantics of applying a SUPAPolicy
```

```
                to a SUPAPolicyTarget.";
    }
```

```
    container supa-policy-target-detail-container {
    description
            "This is a container to collect all object instances of
             type SUPAPolicyTargetDetail.";
        list supa-policy-target-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-target-detail;
            description
                "This is a list of all supa-policy-target-detail
                 objects.";
        }
    }

    identity SUPA-HAS-POLICY-CLAUSE-ASSOC {
        base POLICY-OBJECT-TYPE;
        description
            "The identity corresponding to a SUPAHasPolicyClause
             association class object instance.";
    }

    grouping supa-has-policy-clause-detail {
        uses supa-policy-object-type {
            refine entity-class {
                default SUPA-HAS-POLICY-CLAUSE-ASSOC;
            }
        }
        leaf supa-has-policy-clause-detail-agg-ptr {
            type instance-identifier;
             must "derived-from-or-self (deref(.)/entity-class,
                    'POLICY-STRUCTURE-TYPE')";
            description
                "This leaf is an instance-identifier that references
                 a concrete subclass of the SUPAPolicyStructure class
                 end point of the association represented by this
                 instance of the SUPAHasPolicyClause association [1].
                 The grouping supa-has-policy-clause-detail represents
                 the SUPAHasPolicyClauseDetail association class. Thus,
                 the instance identified by this leaf is the
                 SUPAPolicyStructure instance that is associated by
                 this association to the set of SUPAPolicyClause
                 instances referenced by the
                 supa-has-policy-clause-detail-part-ptr leaf of this
                 grouping.";
        }
        leaf supa-has-policy-clause-detail-part-ptr {
            type instance-identifier;
            must "derived-from-or-self (deref(.)/entity-class,
                    'POLICY-CLAUSE-TYPE')";
```

description

```
                    "This leaf is an instance-identifier that references
                     a concrete subclass of the SUPAPolicyClause class
                     end point of the association represented by this
                     instance of the SUPAHasPolicyClause association [1].
                     The grouping supa-has-policy-clause-detail represents
                     the SUPAHasPolicyClauseDetail association class. Thus,
                     the instance identified by this leaf is the
                     SUPAPolicyClause instance that is associated by this
                     association to the set of SUPAPolicyStructure
                     instances referenced by the
                     supa-has-policy-clause-detail-agg-ptr leaf of this
                     grouping.";
            }
            description
                "This is an association class, and defines the semantics of
                 the SUPAHasPolicyClause association. The attributes and
                 relationships of this class can be used to define which
                 SUPAPolicyTarget objects can be used by which particular
                 set of SUPAPolicyStructure objects. Every
                 SUPAPolicyStructure instance MUST aggregate at
                 least one SUPAPolicyClause instance. However, the
                 converse is NOT true. For example, a SUPAPolicyStructure
                 instance MUST aggregate at least one SUPAPolicyClause
                 instance. However, a SUPAPolicyClause object could be
                 instantiated and then stored for later use in a policy
                 repository.";
        }

        container supa-policy-clause-detail-container {
            description
                "This is a container to collect all object instances of
                 type SUPAPolicyClauseDetail.";
            list supa-policy-clause-detail-list {
                key supa-policy-ID;
                uses supa-has-policy-clause-detail;
                description
                    "This is a list of all supa-policy-clause-detail
                     objects.";
            }
        }

        identity SUPA-HAS-POLICY-EXEC-ACTION-ASSOC {
            base POLICY-OBJECT-TYPE;
            description
                "The identity corresponding to a
                 SUPAHasPolExecFailActionToTake association class
                 object instance.";
        }
```

```
grouping supa-has-policy-exec-action-detail {
    uses supa-policy-object-type {
        refine entity-class {
            default SUPA-HAS-POLICY-EXEC-ACTION-ASSOC;
        }
    }
    leaf supa-has-exec-fail-action-detail-agg-ptr {
        type instance-identifier;
        must "derived-from-or-self (deref(.)/entity-class,
             'POLICY-STRUCTURE-TYPE')";
        description
            "This leaf is an instance-identifier that references
             a SUPAPolicyStructure instance end point of the
             association represented by this instance of the
             SUPAHasPolExecFailActionToTake association [1] that
             was executing a SUPAPolicy. This SUPAPolicyStructure
             is referred to as the 'parent' SUPAPolicyStructure
             instance, while the other instance end point of this
             association is called the 'child' SUPAPolicyStructure.
             The grouping supa-policy-structure-type represents the
             SUPAPolicyStructure class. Thus, the instance
             identified by this leaf is the parent
             SUPAPolicyStructure instance that is associated by this
             association to the child SUPAPolicyStructure instance
             referenced by the
             supa-has-exec-fail-action-detail-part-ptr leaf of this
             grouping.";
    }
    leaf supa-has-exec-fail-action-detail-part-ptr {
        type instance-identifier;
        must "derived-from-or-self (deref(.)/entity-class,
             'POLICY-STRUCTURE-TYPE')";
        description
            "This leaf is an instance-identifier that references
             a SUPAPolicyStructure instance end point of the
             association represented by this instance of the
             SUPAHasPolExecFailActionToTake association [1] that
             was NOT currently executing a SUPAPolicy. This
             SUPAPolicyStructure is referred to as the 'child'
             SUPAPolicyStructure instance, while the other instance
             end point of this association is called the 'parent'
             SUPAPolicyStructure. The grouping
             supa-policy-structure-type represents the
             SUPAPolicyStructure class. Thus, the instance
             identified by this leaf is the child
             SUPAPolicyStructure instance that is associated by
             this association to the child SUPAPolicyStructure
             instance referenced by the
             supa-has-exec-fail-action-detail-part-ptr leaf of
```

```
                this grouping.";
        }
```

```
        leaf-list supa-policy-exec-fail-take-action-name {
            type string;
            description
                "This is a list that contains the set of names for
                 SUPAPolicyActions to use if the SUPAPolicyStructure
                 object that owns this association failed to execute
                 properly. This association defines a set of child
                 SUPAPolicyStructure objects to use if this (the parent)
                 SUPAPolicyStructure object fails to execute correctly.
                 Each child SUPAPolicyStructure object has one or more
                 SUPAPolicyActions; this attribute defines the name(s)
                 of each SUPAPolicyAction in each child
                 SUPAPolicyStructure that should be used to try and
                 remediate the failure.";
        }
        description
            "This is an association class, and defines the semantics of
             the SUPAHasPolExecFailTakeAction association. The
             attributes and relationships of this class can be used to
             determine which SUPAPolicyAction objects are executed in
             response to a failure of the SUPAPolicyStructure object
             instance that owns this association.";
    }

    container supa-policy-exec-fail-take-action-detail-container {
        description
            "This is a container to collect all object instances of
             type SUPAPolExecFailActionToTakeDetail.";
        list supa-policy-exec-fail-take-action-detail-list {
            key supa-policy-ID;
            uses supa-has-policy-exec-action-detail;
            description
                "This is a list of all
                 supa-has-policy-exec-action-detail objects.";
        }
    }

    identity SUPA-HAS-POLICY-METADATA-DECORATOR-DETAIL-ASSOC {
        base POLICY-METADATA-TYPE;
        description
            "The identity corresponding to a
             SUPAHasMetadataDecoratorDetail association class
             object instance.";
    }

    grouping supa-has-policy-metadata-dec-detail {
        uses supa-policy-metadata-type {
            refine entity-class {
                default SUPA-HAS-POLICY-METADATA-DECORATOR-DETAIL-ASSOC;
```

```
            }
        }
```

```
      leaf supa-has-policy-metadata-detail-dec-agg-ptr {
          type instance-identifier;
           must "derived-from-or-self (deref(.)/entity-class,
                 'POLICY-METADATA-TYPE')";
          description
              "This leaf is an instance-identifier that references
               a SUPAPolicyMetadataDecorator instance end point of
               the association represented by this instance of the
               SUPAHasMetadataDecorator association [1]. The
               grouping supa-has-policy-metadata-detail represents
               the SUPAHasMetadataDecoratorDetail association class.
               Thus, the instance identified by this leaf is the
               SUPAPolicyMetadataDecorator instance that is
               associated by this association to the set of
               SUPAPolicyMetadata instances referenced by the
               supa-has-policy-metadata-detail-dec-part-ptr leaf of
               this grouping.";
      }
      leaf supa-has-policy-metadata-detail-dec-part-ptr {
          type instance-identifier;
          must "derived-from-or-self (deref(.)/entity-class,
                 'POLICY-METADATA-TYPE')";
          description
              "This leaf is an instance-identifier that references
               a SUPAPolicyMetadata instance end point of the
               association represented by this instance of the
               SUPAHasMetadataDecorator association [1]. The
               grouping supa-has-policy-metadata-detail represents
               the SUPAHasMetadataDecoratorDetail association class.
               Thus, the instance identified by this leaf is the
               SUPAPolicyMetadata instance that is associated by
               this association to the set of
               SUPAPolicyMetadataDecorator instances referenced by
               the supa-has-policy-metadata-detail-dec-agg-ptr leaf
               of this grouping.";
      }
      description
          "This is an association class, and defines the semantics of
           the SUPAHasMetadataDecorator association. The attributes
           and relationships of this class can be used to define which
           concrete subclasses of the SUPAPolicyMetadataDecorator
           class can be used to wrap which concrete subclasses of the
           SUPAPolicyMetadata class.";
  }

  container supa-policy-metadata-decorator-detail-container {
      description
          "This is a container to collect all object instances of
           type SUPAHasMetadaDecoratorDetail.";
```

```
        list supa-policy-metadata-decorator-detail-list {
            key supa-policy-metadata-id;
            uses supa-has-policy-metadata-dec-detail;
            description
                "This is a list of all supa-policy-metadata-detail
                 objects.";
        }
    }
}
```

<CODE ENDS>

6. IANA Considerations

No IANA considerations exist for this document.

7. Security Considerations

TBD

8. Acknowledgments

This document has benefited from reviews, suggestions, comments
and proposed text provided by the following members, listed in
alphabetical order:

    Andy Bierman
    Benoit Claise
    Martin Bjorklund
    Qin Wu

9. References

This section defines normative and informative references for this
document.

9.1. Normative References

    [RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
                Requirement Levels", BCP 14, RFC 2119, March 1997.
    [RFC6020]   Bjorklund, M., "YANG - A Data Modeling Language for
                the Network Configuration Protocol (NETCONF)",

[RFC 6020](), October 2010.

    [RFC6991]   Schoenwaelder, J., "Common YANG Data Types", RFC 6991,
                July 2013.
    [RFC7950]   Bjorklund, M., Ed., "The YANG 1.1 Data Modeling
                Language", August 2016.

9.2. Informative References

    [1]   Strassner, J., Halpern, J., Coleman, J., "Generic
                Policy Information Model for Simplified Use of Policy
                Abstractions (SUPA)", March 21, 2016,
                draft-ietf-supa-generic-policy-info-model-01
    [2]   http://www.omg.org/spec/OCL/
    [3]   http://doc.omg.org/formal/2002-04-03.pdf
    [4]   http://alloy.mit.edu/alloy/
    [5]   http://www.omg.org/spec/QVT/
    [6]   http://semver.org/
    [7]   Definitions of DAC, MAC, and RBAC may be found here:
                http://csrc.nist.gov/groups/SNS/rbac/faq.html#03
    [8]   ABAC is described here:
                http://csrc.nist.gov/groups/SNS/rbac/index.html

Authors' Addresses

    Joel Halpern
    Ericsson
    P. O. Box 6049
    Leesburg, VA 20178
    Email: joel.halpern@ericsson.com

    John Strassner
    Huawei Technologies
    2330 Central Expressway
    Santa Clara, CA 95138 USA
    Email: john.sc.strassner@huawei.com

    Sven van der Meer
    LM Ericsson Ltd.
    Ericsson Software Campus
    Garrycastle
    Athlone
    N37 PV44
    Ireland
    Email: sven.van.der.meer@ericsson.com