A VPL Grammar and XML DTD

A.1 LALR Grammar

The following VPL grammar was written for an LALR parser generator. The terminal symbols, of the language are in capitals, the terminal `ID` refers to an identifier.

```
policy ::=
       POLICY ID LCBRACE definitions RCBRACE;

definitions ::=
       definition definitions | definition ;

definition ::=
       role_definitions | view_definition | schema_definition ;

role_definitions ::=
       ROLES role_def_list ;

role_def_list ::=
       role_definition role_def_list | role_definition  ;

role_definition ::=
       ID inheritance_spec initial_view_defs card_constraint_def
       excl_constraint_def req_constraint_def;

inheritance_spec ::=
       COLON comma_separated_list | empty ;

comma_separated_list ::=
       ID COMMA comma_separated_list | ID ;

initial_view_defs ::=
       initial_view_def_list | empty  ;

initial_view_def_list ::=
       initial_view_def | initial_view_def initial_view_def_list ;

initial_view_def ::=
```

```
       HOLDS comma_separated_list ON ID ;

card_constraint_def ::=
       MAXCARD NUMBER | MINCARD NUMBER | empty ;

excl_constraint_def ::=
       EXCLUDES comma_separated_list | empty ;

req_constraint_def ::=
       REQUIRES comma_separated_list | empty ;

view_definition ::=
       assignable_modifier static_modifier
       VIEW ID inheritance_spec controlled_type_spec
       role_restriction_list required_views_list
       LCBRACE view_body RCBRACE
     | assignable_modifier static_modifier VIRTUAL VIEW ID
       inheritance_spec controlled_type_spec
       role_restriction_list required_views_list ;

assignable_modifier ::=
       ASSIGNABLE | empty ;

static_modifier ::=
       STATIC | empty ;

controlled_type_spec ::=
       CONTROLS ID | empty  ;

role_restriction_list ::=
       RESTRICTEDTO comma_separated_list |  empty ;

required_views_list ::=
       REQUIRES comma_separated_list | empty  ;

view_body ::=
       allowed_rights_list denied_rights_list ;

allowed_rights_list ::=
       ALLOW rights_list | empty ;

denied_rights_list ::=
       DENY rights_list | empty ;

rights_list  ::=
       STRONG ID rights_list | STRONG ID | ID | ID rights_list ;

schema_definition ::=
```

```
       SCHEMA ID OBSERVES ID LCBRACE schema_body RCBRACE ;

schema_body ::=
       schema_clauses_list ;

schema_clauses_list ::=
       schema_clause | schema_clause schema_clauses_list ;

schema_clause ::=
       ID assignments_spec_list removals_spec_list ;

assignments_spec_list ::=
       assignments_list | empty ;

assignments_list ::=
       assignment_spec assignments_list | assignment_spec ;

assignment_spec ::=
       ASSIGNS comma_separated_list ON target_spec
       TO recipient_list assign_option ;

target_spec ::=
       ID:type_name | object_ref | object_ref DOT ID ;

object_ref ::=
       THIS | RESULT | LBRACE ID RBRACE;

recipient_list ::=
       CALLER | CALLER COMMA comma_separated_list | comma_separated_list ;

removals_spec_list ::=
       removals_list | empty  ;

removals_list ::=
       removal_spec removals_list | removal_spec ;

removal_spec ::=
       REMOVES comma_separated_list ON target_spec FROM recipient_list ;

assign_option ::=
       WITHASSIGNOPTION | empty ;

empty ::= /* nothing */ ;
```

A.2 XML Document Type Definition

```
<?xml version="1.0" encoding="UTF-8"?>

<!--                                                                -->
<!--            DTD with syntax for VPL                             -->
<!--                                                                -->

<!ELEMENT policy (role*,  ( view | schema )+ )>
<!ATTLIST policy
          name ID #REQUIRED
>

<!--    Roles    -->

<!ELEMENT role-ref (#PCDATA)>

<!ELEMENT role ( inherits*, holds*, cardinality-constraint?,
          ( exclusion-constraint | prerequisite-constraint )* )>

<!ATTLIST role name ID #REQUIRED>

<!ELEMENT inherits EMPTY>
<!ATTLIST inherits role IDREF #REQUIRED>

<!ELEMENT holds EMPTY>
<!ATTLIST holds view CDATA #REQUIRED
               on-type CDATA #REQUIRED>

<!ELEMENT cardinality-constraint EMPTY>
<!ATTLIST cardinality-constraint
          value CDATA #REQUIRED>

<!ELEMENT exclusion-constraint EMPTY>
<!ATTLIST exclusion-constraint
          role IDREF #REQUIRED>

<!ELEMENT prerequisite-constraint EMPTY>
<!ATTLIST prerequisite-constraint
          role IDREF #REQUIRED>

<!-- alternate root element, for roles only -->
<!ELEMENT roles (role*)>

<!--    Views    -->

<!ELEMENT view-ref  (#PCDATA)>
```

```
<!ELEMENT view ( allow?, deny?)>
<!ATTLIST view
          name CDATA #REQUIRED
          extends CDATA #IMPLIED
          controls CDATA #IMPLIED
          requires CDATA #IMPLIED
          restricted-to CDATA #IMPLIED
          assignable ( true | false ) "false"
          static ( true | false ) "false"
          virtual ( true | false ) "false"
>

<!--  allow/deny -->

<!ELEMENT allow ( right+ ) >
<!ELEMENT deny ( right+ )>

<!ELEMENT right EMPTY>

<!ATTLIST right name CDATA #REQUIRED
          priority ( strong | weak ) "weak"
>

<!--  Schemas -->

<!ELEMENT schema (rights-change)+>
<!ATTLIST schema
          name CDATA #REQUIRED
          observes CDATA #REQUIRED
>

<!ELEMENT rights-change (  change+ ) >
<!ATTLIST rights-change operation CDATA #REQUIRED>

<!ELEMENT change ( ( view-ref+ ), target, recipient+ )+ >
<!ATTLIST change mode ( assigns | removes ) #REQUIRED >

<!ELEMENT target ( type | argument | dynref )>

<!ELEMENT type EMPTY >
<!ATTLIST type name CDATA #REQUIRED>

<!ELEMENT dynref EMPTY >
<!ATTLIST dynref
          type ( result | this  ) #REQUIRED
          member CDATA #IMPLIED
>
```

```
<!ELEMENT argument EMPTY >
<!ATTLIST argument
          name CDATA #REQUIRED
          member CDATA #IMPLIED
>

<!ELEMENT recipient ( caller | subjectref+ ) >
<!ELEMENT caller EMPTY >
<!ELEMENT subjectref ( #PCDATA )>

<!-- end of DTD -->
```