A1.2 Syntax of proof contexts

The following changes and additions to the above productions show the location and form of proof annotations.

* statement ::=
simple_statement | compound_statement | proof_statement

\+ proof_statement ::=
assert_statement | check_statement

* loop_statement ::= [*loop*_statement_identifier :]
[iteration_scheme [loop_invariant]] **loop**
sequence_of_statements
end loop [*loop*_identifier];

\+ loop_invariant ::= assert_statement

\+ assert_statement ::= --# **assert** predicate;

\+ check_statement ::= --# **check** predicate;

\+ procedure_annotation ::=
[global_definition]
[dependency_relation]
[precondition]
[postcondition]

\+ function_annotation ::=
[global_definition]
[precondition]
[return_annotation]

\+ precondition ::= --# **pre** predicate;

\+ postcondition ::= --# **post** predicate;

\+ return_annotation ::=
--# **return** expression;
| --# **return** identifier => predicate;

\+ predicate ::= *boolean*_expression

* expression ::=
quantified_expression
| relation -> relation | relation <-> relation
| relation {**and** relation } | relation {**and then** relation
| relation {**or** relation } | relation {**or else** relation
| relation {**xor** relation}

\+ quantified_expression ::= quantifier_kind
defining_identifier **in** *discrete*_subtype_mark [**range** range]
=> (predicate)

\+ quantifier_kind ::= **for all** | **for some**

```
* name ::= direct_name [~]
        | indexed_component | selected_component [~]
        | attribute_reference | type_conversion | function_call
        | record_update | array_update

+ record_update ::= prefix [ selector_name => expression
                                        {; selector_name => expression} ]

+ array_update ::= prefix [ index_list => expression
                                        {; index_list => expression} ]

+ index_list ::= expression {, expression}

* basic_declarative_item ::= basic_declaration | representation_clause
        | proof_function_declaration | basic_proof_declaration

+ proof_function_declaration ::= --# function_specification;

+ basic_proof_declaration ::= proof_type_declaration | type_assertion

+ proof_type_declaration ::=
        --# type defining_identifier is proof_type_definition;

+ proof_type_definition ::= abstract

+ type_assertion ::= --# assert identifier'Base is subtype_mark;

+ own_variable_clause ::= --# own own_variable_specification
                                        {own_variable_specification}

+ own_variable_specification ::= own_variable_list [: subtype_mark];
```

Notes

In the productions for record_update and array_update the square brackets stand for themselves and are not metasymbols.