Information technology — Open Systems Interconnection — The Directory: Models

TECHNICAL CORRIGENDUM 4

Technologies de l'information — Interconnexion de systèmes ouverts (OSI) — L'annuaire: Les modèles

RECTIFICATIF TECHNIQUE 4

Withdrawn
1) Correction of the defects reported in defect report 357

In clause 13.7.6 and Annex B replace the STRUCTURE-RULE information object with:

```
STRUCTURE-RULE ::= CLASS {
   &nameForm                NAME-FORM,
   &SuperiorStructureRules  STRUCTURE-RULE.&id OPTIONAL,
   &id                      RuleIdentifier }
WITH SYNTAX {
   NAME FORM                &nameForm
   [ SUPERIOR RULES         &SuperiorStructureRules ]
   ID                       &id }
```

2) Correction of the defects reported in defect report 359

Update the ASN.1 in clause 28.3 and Annex G as shown:

```
ModifyOperationalBindingResult ::= CHOICE {
   null       [0]  NULL,
   protected  [1]  OPTIONALLY-PROTECTED-SEQ{SEQUENCE {
      newBindingID    OperationalBindingID,
      bindingType     OPERATIONAL-BINDING.&id({OpBindingSet}),
      newAgreement    OPERATIONAL-BINDING.&Agreement
                           ({OpBindingSet}@.bindingType),
      valid           Validity OPTIONAL,
      COMPONENTS OF   CommonResultsSeq }}
```

3) Correction of the defects reported in defect report 360

Update the ASN.1 in clause 13.9.2 and Annex B as shown:

```
CONTEXT ::= CLASS {
   &Type,
   &defaultValue  &Type OPTIONAL,
   &Assertion      OPTIONAL,
   &absentMatch    BOOLEAN DEFAULT TRUE,
   &id             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
   WITH SYNTAX     &Type
   [DEFAULT-VALUE  &defaultValue]
   [ASSERTED AS    &Assertion]
   [ABSENT-MATCH   &absentMatch]
   ID               &id }
```
4) Correction of the defects reported in defect report 361

Update the clause 18.4.2.4, item b), fourth bullet as shown:

– userGroup is the set of users who are members of the groupOfUniqueNames or groupOfUniqueNames entry, identified by the specified distinguished name (with an optional unique identifier). Members of a group of unique names are treated as individual object names, and not as the names of other groups of unique names. How group membership is determined is described in 18.4.2.5.

5) Correction of the defects reported in defect report 363

Update item a) of clause 13.10.2 of X.501 as shown:

a) the attributeType component identifies the attribute type to which the DIT Context Use applies; if it applies to any attribute type the object identifier or any attribute type (id-oa-allAttributeTypes) may be used (defined in Annex B);

In Annex B of X.501 add to the end of the allocation of object identifiers for operational attributes:

id-oa-allAttributeTypes OBJECT IDENTIFIER ::= {id-oa 48}

6) Correction of the defects reported in defect report 370

In clause 22.5 just before the note, add a new paragraph:

The subordinate references making up the root naming context are conceptually placed in DSA specific entries (DSEs) immediately subordinate to the root DSE (see 24.2). The DSE type shall be subr.

7) Correction of the defects reported in defect report 371

In clause 27.3.3, change the OP-BIND-ROLE information object class as shown

OP-BIND-ROLE ::= CLASS {
  &establish BOOLEAN DEFAULT FALSE,
  &EstablishParam OPTIONAL,
  &modify BOOLEAN DEFAULT FALSE,
  &ModifyParam OPTIONAL,
  &terminate BOOLEAN DEFAULT FALSE,
  &TerminateParam OPTIONAL
} WITH SYNTAX {
  [ESTABLISHMENT-INITIATOR &establish]
  [ESTABLISHMENT-PARAMETER &EstablishParam]
  [MODIFICATION-INITIATOR &modify]
  [MODIFICATION-PARAMETER &ModifyParam]
  [TERMINATION-INITIATOR &terminate]
  [TERMINATION-PARAMETER &TerminateParam] }  

In 27.3.3 Also, change item b) as shown:

b) The ESTABLISHMENT-PARAMETER field defines the ASN.1 type for the parameters exchanged by a DSA assuming the defined role when an instance of the operational binding type is established. If no parameters are to be exchanged, then the NULL ASN.1 type shall be specified.

Replace clauses 28.2, 28.3 and 28.4 with:

28.2 Establish Operational Binding operation

28.2.1 Establish Operational Binding syntax

The Establish Operational Binding operation allows establishment of an operational binding instance of a predefined type between two DSAs. This is achieved through the transfer of the establishment parameters and the terms of agreement which were defined in the definition of the operational binding type. The arguments of the operation may be signed (see 17.3) by the requestor. If the target component of the SecurityParameters (see 7.10 of Rec. ITU-T X.511 | ISO/IEC 9594-3) in the request is set to signed and a result is to be returned, the result may be signed. Otherwise, the result shall not be signed.
In the case of a symmetrical operational binding, either of the two DSAs may take the initiative to establish an operational binding instance of the predefined type.

In the case of an asymmetrical operational binding, just one of the roles are designated to initiate the establishment of an operational binding or either of the two DSAs may take the initiative depending on the definition of the operational binding type.

```
establishOperationalBinding OPERATION ::= {
  ARGUMENT EstablishOperationalBindingArgument
  RESULT EstablishOperationalBindingResult
  ERRORS {operationalBindingError | securityError}
  CODE id-op-establishOperationalBinding }

EstablishOperationalBindingArgument ::= 
  OPTIONALLY-PROTECTED-SEQ { EstablishOperationalBindingArgumentData }

EstablishOperationalBindingArgumentData ::= SEQUENCE {
  bindingType [0] OPERATIONAL-BINDING.&id({OpBindingSet}),
  bindingID [1] OperationalBindingID OPTIONAL,
  accessPoint [2] AccessPoint,
    -- symmetric, Role A initiates, or Role B initiates
  initiator CHOICE {
    symmetric [3] OPERATIONAL-BINDING.&both.&EstablishParam
      ({OpBindingSet}{@bindingType}),
      ({OpBindingSet}{@bindingType}),
    roleB-initiates [5] OPERATIONAL-BINDING.&roleB.&EstablishParam
      ({OpBindingSet}{@bindingType}),
  agreement [6] OPERATIONAL-BINDING.&Agreement
    ({OpBindingSet}{@bindingType}),
  valid [7] Validity DEFAULT {},
  securityParameters [8] SecurityParameters OPTIONAL,
}

OpBindingSet OPERATIONAL-BINDING ::= 
  {shadowOperationalBinding | hierarchicalOperationalBinding | 
    nonSpecificHierarchicalOperationalBinding}

OperationalBindingID ::= SEQUENCE {
  identifier INTEGER,
  version INTEGER,
}

Validity ::= SEQUENCE {
  validFrom [0] CHOICE {
    now [0] NULL,
    time [1] Time,
  } DEFAULT now:NULL,
  validUntil [1] CHOICE {
    explicitTermination [0] NULL,
    time [1] Time,
  } DEFAULT explicitTermination:NULL,
}

Time ::= CHOICE {
  utcTime UTCTime,
  generalizedTime GeneralizedTime,
}

EstablishOperationalBindingResult ::= 
  OPTIONALLY-PROTECTED-SEQ { EstablishOperationalBindingResultData }

EstablishOperationalBindingResultData ::= SEQUENCE {
  bindingType [0] OPERATIONAL-BINDING.&id({OpBindingSet}),
  bindingID [1] OperationalBindingID OPTIONAL,
  accessPoint [2] AccessPoint,
    -- symmetric, Role A replies, or Role B replies
  initiator CHOICE {
    symmetric [3] OPERATIONAL-BINDING.&both.&EstablishParam
      ({OpBindingSet}{@bindingType}),
```
28.2.2 Establish Operational Binding arguments

The `bindingType` component shall specify which type of operational binding is to be established. An operational binding type is defined by an instance of the `OPERATIONAL-BINDING` information object class which assigns an object identifier value to the operational binding type. If the receiver does not recognize or support the operational binding type, it shall return an `operationalBindingError` with problem `unsupportedBindingType`.

The `bindingID` component, when present, shall hold an identification of the new operational binding instance. If the `bindingID` is absent within the operation argument, the responding DSA shall assign an ID to the operational binding instance and return it in the `bindingID` component of the `EstablishOperationalBindingResult` data type. In either case, when establishing an operational binding, both the `identifier` and `version` components of the `OperationalBindingID` data type shall be unique for all operational bindings between any two DSAs. However, the DSA not making the assignment shall accept an `identifier` component that is only unique within a specific operational binding type. If the identifier component specifies an identifier already in use for the particular binding type, the responding DSA shall return an `operationalBindingError` with problem `duplicateID`.

NOTE – A pre-edition 5 system may not follow the above rule for assigning identifiers.

The `accessPoint` component shall specify the access point of the initiator for subsequent interactions.

The `initiator` component shall specify the role the DSA issuing the Establish Operational Binding operation assumes. The semantics of the roles are defined as part of the definition of the operational binding type. It is a choice of three alternatives:

- **The symmetric alternative** shall be taken, if the type of operational binding requires identical roles for the two DSAs. The establishment parameter for the initiating DSA is determined by the `OP-BIND-ROLE` associated with the `SYMMETRIC` field of the instance of `OPERATIONAL-BINDING` information object class. If this alternative is chosen in the request, but the operational binding type specifies asymmetric roles, then the responding DSA shall return an `operationalBindingError` with problem `notAllowedForRole`.

- **The roleA-initiates alternative** may be taken if both roles may be the initiator of an asymmetric operational binding and it shall be taken when only the initiating DSA may take ROLE-A. The establishment parameter for the initiating DSA is determined by the `OP-BIND-ROLE` associated with ROLE-A field of the instance of `OPERATIONAL-BINDING` information object class. If the DSA in ROLE-A is not allowed to initiate the operational binding, the responding DSA shall return an `operationalBindingError` with problem `notAllowedForRole`. If the responding system does not accept the role allocation, it shall return an `operationalBindingError` with problem `roleAssignment`.

- **The roleB-initiates alternative** may be taken if both roles may be the initiator of an asymmetric operational binding and it shall be taken when only the initiating DSA may take ROLE-B. The establishment parameter for the initiating DSA is determined by the `OP-BIND-ROLE` associated with ROLE-B field of the instance of `OPERATIONAL-BINDING` information object class. If the DSA in ROLE-B is not allowed to initiate the operational binding, the responding DSA shall return an `operationalBindingError` with problem `notAllowedForRole`. If the responding DSA does not accept the role allocation, it shall return an `operationalBindingError` with problem `roleAssignment`.

If for any of the three alternatives the data type for establishment parameters is the `NULL` ASN.1 type, where according to the operational binding type should be another data type, then the responding DSA shall return an `operationalBindingError` with problem `parametersMissing`.

The `agreement` component, when present, shall specify the terms of agreement governing the operational binding instance. Its actual content depends on the type of operational binding to be established. The ASN.1 type for this parameter is defined by the `AGREEMENT` field of the `OPERATIONAL-BINDING` information object for the operational binding type.
The valid component shall specify the duration of the operational binding.

- The validFrom subcomponent shall specify the starting time of the operational binding instance. If the now alternative is taken, the operational binding becomes active when the operation has successfully completed. If the time alternative is taken, the operational binding becomes active at the specified time. If the receiving DSA cannot accept the starting time, e.g., the starting time makes no sense or for other reasons, it shall return an operationalBindingError with problem invalidStartTime.

- The validUntil shall specify the time that the operational binding instance is terminated. If the explicitTermination alternative is taken, the operational binding is active until explicitly terminated. If the time alternative is taken, the operational binding is terminated at the specified time. If the receiving DSA cannot accept the ending time, e.g., the ending time makes no sense or for other reasons, it shall return an operationalBindingError with problem invalidEndTime.

When a value of Time in the UTCTime syntax, the value of the two-digit year field shall be normalised into a four-digit year value as follows:

- If the 2-digit value is 00 through 49 inclusive, the value shall have 2000 added to it.
- If the 2-digit value is 50 through 99 inclusive, the value shall have 1900 added to it.

The use of GeneralizedTime may prevent interworking with implementations unaware of the possibility of choosing either UTCTime or GeneralizedTime. It is the responsibility of those specifying the domains in which this Directory Specification will be used, e.g., profiling groups, as to when the GeneralizedTime may be used. In no case shall UTCTime be used for representing dates beyond 2049.

If the Validity data type is an empty sequence or if the valid component is not present, then the operational binding is valid from the current time and until it is explicitly terminated.

The securityParameters component shall be present if the request is signed or if the result or error is requested to be signed.

28.2.3 Establish Operational Binding results

If the Establish Operational Binding operation succeeds, the result shall be returned.

The bindingType component shall have the same value as that provided by the establishment initiator.

The bindingID component shall hold a valid identification of the established operational binding instance if the corresponding component of the request was absent (see 28.2.2). Otherwise, it may be present, but shall then echo the value in the request.

The accessPoint component shall specify the access point of the responding DSA for subsequent interactions.

The initiator component shall specify the role that the responding DSA assumes. The semantics of the roles are defined as part of the definition of the operational binding type. It is a choice of three alternatives:

- The symmetric alternative shall be taken if the corresponding alternative was taken in the received request. The establishment parameter for the responding DSA is the same as given in the request.
- The roleA-replies alternative shall be taken, if the initiating DSA took the ROLE-B. The establishment parameter for the responding DSA is determined by the OP-BIND-ROLE associated with ROLE-A field of the instance of OPERATIONAL-BINDING information object class.
- The roleB-replies alternative shall be taken if the initiating DSA took ROLE-A. The establishment parameter for the responding DSA is determined by the OP-BIND-ROLE associated with ROLE-B field of the instance of OPERATIONAL-BINDING information object class.

If the result is to be signed by the responding DSA, the securityParameters component of CommonResultsSeq shall be present.

28.3 Modify Operational Binding operation

28.3.1 Modify Operational Binding syntax

The Modify Operational Binding operation is used to modify an established operational binding. The right to modify is indicated by the MODIFICATION INITIATOR field(s) within the definition of the operational binding type using the OP-BIND-ROLE and OPERATIONAL-BINDING information object.

The components of an operational binding that can be modified are the content of the agreement for the operational binding and its period of validity. Further, a modification parameter can be specified by the initiator of the Modify
Operational Binding operation. The arguments of the operation may be signed (see 17.3) by the requestor. If the target component of the SecurityParameters (see 7.10 of Rec. ITU-T X.511 | ISO/IEC 9594-3) in the request is set to signed and a result is to be returned, the result may be signed. Otherwise, the result shall not be signed.

If the initiator of the Modify Operational Binding operation according to the operational binding type is not allowed to be the initiator, the responding DSA shall return an operationalBindingError with problem notAllowedForRole.

modifyOperationalBinding OPERATION ::= {
  ARGUMENT ModifyOperationalBindingArgument
  RESULT ModifyOperationalBindingResult
  ERRORS {operationalBindingError | securityError}
  CODE id-op-modifyOperationalBinding }

ModifyOperationalBindingArgument ::= OPTIONALLY-PROTECTED-SEQ { ModifyOperationalBindingArgumentData }

ModifyOperationalBindingArgumentData ::= SEQUENCE {
  bindingType       [0]  OPERATIONAL-BINDING.&id({OpBindingSet}),
  bindingID         [1]  OperationalBindingID,
  accessPoint       [2]  AccessPoint OPTIONAL,
  -- symmetric, Role A initiates, or Role B initiates
  initiator              CHOICE {
    symmetric         [3]  OPERATIONAL-BINDING.&both.&ModifyParam
      (OpBindingSet){@bindingType},
      (OpBindingSet){@bindingType}),
    roleB-initiates   [5]  OPERATIONAL-BINDING.&roleB.&ModifyParam
      (OpBindingSet){@bindingType}),
  newBindingID      [6]  OperationalBindingID,
  newAgreement      [7]  OPERATIONAL-BINDING.&Agreement
      (OpBindingSet){@bindingType}) OPTIONAL,
  valid               [8]  ModifiedValidity OPTIONAL,
  securityParameters  [9]  SecurityParameters OPTIONAL,
...}

ModifiedValidity ::= SEQUENCE {
  validFrom            [0]  CHOICE {
    now                  [0]  NULL,
    time                 [1]  Time,
    -- DEFAULT now:NULL,
    validUntil           [1]  CHOICE {
      explicitTermination [0]  NULL,
      time                 [1]  Time,
      unchanged            [2]  NULL,
      -- DEFAULT unchanged:NULL,
    }
  }

ModifyOperationalBindingResult ::= CHOICE {
  null       [0]  NULL,
  protected  [1]  OPTIONALLY-PROTECTED-SEQ{ ModifyOperationalBindingResultData }
}

ModifyOperationalBindingResultData ::= SEQUENCE {
  newBindingID   OperationalBindingID,
  bindingType    OPERATIONAL-BINDING.&id({OpBindingSet}),
  newAgreement   OPERATIONAL-BINDING.&Agreement
      (OpBindingSet){@.bindingType}),
  valid           Validity OPTIONAL,
  ...}

COMPONENTS OF CommonResultsSeq

28.3.2 Modify Operational Binding argument

The bindingType component shall specify which type of operational binding is to be modified. If no operational binding of the specified type has been established between the two DSAs, the responding DSA shall return an operationalBindingError with problem invalidBindingType.
The bindingID component shall specify the operational binding instance to be modified. If the bindingID is unknown to the responding DSA, it shall return an operationalBindingError with problem invalidID.

The accessPoint component, if present, shall specify the initiator's access point for subsequent interactions. This component shall be present, if the access point is changed.

The initiator component, when present, shall specify the role that the DSA issuing the Modify Operational Binding operation assumed during the Establish Operational Binding operation. This component shall be present if the MODIFICATION-PARAMETER of the initiator's OP-BIND-ROLE information object for the taken alternative is present. Otherwise, it shall be absent. If the chosen role is not the correct one, the responding DSA shall return an operationalBindingError with problem roleAssignment.

The newBindingID component shall hold the revised identifier of the operational binding instance. The version component of newBindingID shall be greater than that of bindingID. The identifier subcomponent shall remain unchanged. If the identifier subcomponent in this component is different from the identifier subcomponent of bindingID component, the responding DSA shall return an operationalBindingError with problem invalidNewID.

The newAgreement component, if present, shall contain the modified terms of agreement governing the operational binding instance. The ASN.1 type for this parameter is defined by the AGREEMENT field of the OPERATIONAL-BINDING information object class template of the operational binding type. If newAgreement is not present, the agreement is not changed by the operation.

The valid component, if present, may be used to indicate a revised period of validity for the altered agreement. If the valid component is absent, the validFrom component is presumed to have the value now and the validUntil component is assumed unchanged. If the validFrom component is present and refers to an instant of time in the future, the current agreement remains in effect until that time, unless operational binding is explicitly terminated before that time.

The securityParameters component shall be present if the request is signed or if the result or error is requested to be signed.

28.3.3 Modify Operational Binding results

If the Modify Operational Binding operation succeeds, the result shall be returned.

The newBindingID component shall echo the newBindingID component in the request.

The bindingType component shall echo the bindingType component in the request.

The newAgreement component shall echo the newAgreement component in the request.

The valid component shall echo the valid component in the request.

If the result is to be signed by the responding DSA, the securityParameters component of CommonResultsSeq shall be present.

It is not possible for the responding DSA to return the modification parameter defined for its role to the modification initiator.

28.4 Terminate Operational Binding operation

28.4.1 Terminate Operational Binding syntax

The Terminate Operational Binding operation is used to request the termination of an established operational binding instance. The right to request termination is indicated by the TERMINATION INITIATOR field(s) within the definition of the operational binding type using the OP-BIND-ROLE and OPERATIONAL-BINDING information object class templates. The arguments of the operation may be signed (see 17.3) by the requestor. If the target component of the SecurityParameters (see 7.10 of Rec. ITU-T X.511 | ISO/IEC 9594-3) in the request is set to signed and a result is to be returned, the result may be signed. Otherwise, the result shall not be signed.

If the initiator of the Terminate Operational Binding operation according to the operational binding type is not allowed to be the initiator, the responding DSA shall return an operationalBindingError with problem notAllowedForRole.
ERRORS    {operationalBindingError | securityError} 
CODE      id-op-terminateOperationalBinding 

TerminateOperationalBindingArgument ::= 
  OPTIONALLY-PROTECTED-SEQ { TerminateOperationalBindingArgumentData }

TerminateOperationalBindingArgumentData ::= SEQUENCE {
  bindingType         [0]  OPERATIONAL-BINDING.&id({OpBindingSet}),
  bindingID           [1]  OperationalBindingID,
  -- symmetric, Role A initiates, or Role B initiates 
  initiator                CHOICE {
    symmetric           [2]  OPERATIONAL-BINDING.&both.&TerminateParam 
      ({OpBindingSet}{@bindingType}),
      ({OpBindingSet}{@bindingType}),
    roleB-initiates     [4]  OPERATIONAL-BINDING.&roleB.&TerminateParam 
      ({OpBindingSet}{@bindingType}) OPTIONAL,
  terminateAt         [5]  Time OPTIONAL,
  securityParameters  [6]  SecurityParameters OPTIONAL,
...}

TerminateOperationalBindingResult ::= CHOICE {
  null       [0]  NULL,
  protected  [1]  OPTIONALLY-PROTECTED-SEQ{ TerminateOperationalBindingResultData },
... }

TerminateOperationalBindingResultData ::= SEQUENCE {
  bindingID       OperationalBindingID,
  bindingType     OPERATIONAL-BINDING.&id({OpBindingSet}),
  terminateAt     GeneralizedTime OPTIONAL,
  COMPONENTS OF   CommonResultsSeq }

28.4.2 Terminate Operational Binding argument

The bindingType component shall specify which type of operational binding is to be terminated. If no operational 
binding of the specified type has been established between the two DSAs, the responding DSA shall return an 
operationalBindingError with problem invalidBindingType.

The bindingID component shall specify the operational binding instance to be terminated. The version component 
present in the bindingID shall be ignored. If there are supplicate IDs for different binding types, then the combination 
of bindingType and bindingID components shall be used for identifying the operational binding to be terminated. If 
it is not possible to locate an existing operational binding between the two DSAs where the binding type and the 
binding id fit the combination of the bindingType and bindingID components in the request, the responding DSA 
shall return an operationalBindingError with problem invalidBindingType.

The initiator component, when present, shall specify the role that the DSA issuing the Terminate Operational 
Binding operation assumed during the Establish Operational Binding operation. This component shall be present if the 
termination-parameter of the initiator's OP-BIND-ROLE information object for the taken alternative is present. 
Otherwise, it shall be absent.

The terminateAt component, when present, shall specify a time at which the operational binding shall terminate. If 
this component is not present, the operational binding terminates at the completion of the operation.

The securityParameters component shall be present if the request is signed or if the result or error is requested to 
be signed.

28.4.3 Terminate Operational Binding result

If the Terminate Operational Binding operation succeeds, the result shall be returned.

The newBindingID component shall echo the newBindingID component in the request.

The bindingType component shall echo the bindingType component in the request.

The terminateAt component shall echo the terminateAt component in the request.
If the result is to be signed by the responding DSA, the `securityParameters` component of `CommonResultsSeq` shall be present.

It is not possible for the responding DSA to return the termination parameter defined for its role to the termination initiator.

*In 28.5 add new problem codes:*

```
modificationNotAllowed (10),
invalidBindingType (11),
invalidNewID (12)
```

*and add the following text:*

l) `invalidBindingType`: A `modifyOperationalBinding` or a `terminateOperationalBinding` request specifies an operational binding type not established between the two DSAs in question.

m) `invalidNewID`: The new binding ID given in the request is invalid.