4. Protocol Specification

The Information Retrieval application protocol specifies the formats and procedures governing the transfer of information between a Z39.50 origin/target pair. Sections 4.1 and 4.2 respectively describe the formats and rules for exchange of Z39.50 application protocol data units (APDUs). An APDU is a unit of information, transferred between origin and target, whose format is specified by the Z39.50 protocol, consisting of application-protocol-information and possibly application-user-data. Sections 4.3 and 4.4 respectively describe rules for extensibility and conformance requirements.

4.1 Abstract Syntax and ASN.1 Specification of Z39.50 APDUs

This section describes the abstract syntax of the Z39.50 APDUs, using the ASN.1 notation defined in ISO 8824. The comments included within the ASN.1 specification are part of the standard.

```
Z39-50-APDU-1995 -- OID for this definition, assigned in OID.3.1, is {Z39-50 2 1}
DEFINITIONS ::=
BEGIN   -- Z39.50 Maintenance Agency Official Text for ANSI/NISO Z39.50-1995 - July 1995
--
EXPORTS OtherInformation, Term, AttributeSetId, AttributeList, AttributeElement, ElementSetName, SortElement,
DatabaseName, CompSpec, Specification, Permissions, InternationalString, IntUnit, Unit, StringOrNumeric, Query,
Records, ResultSetId, DefaultDiagFormat, DiagRec;
--
PDU ::= CHOICE{
 initRequest                    [20] IMPLICIT InitializeRequest,
 initResponse                   [21] IMPLICIT InitializeResponse,
 searchRequest                  [22] IMPLICIT SearchRequest,
 searchResponse                 [23] IMPLICIT SearchResponse,
 presentRequest                 [24] IMPLICIT PresentRequest,
 presentResponse                [25] IMPLICIT PresentResponse,
 deleteResultSetRequest         [26] IMPLICIT DeleteResultSetRequest,
 deleteResultSetResponse        [27] IMPLICIT DeleteResultSetResponse,
 accessControlRequest           [28] IMPLICIT AccessControlRequest,
 accessControlResponse          [29] IMPLICIT AccessControlResponse,
 resourceControlRequest         [30] IMPLICIT ResourceControlRequest,
 resourceControlResponse        [31] IMPLICIT ResourceControlResponse,
 triggerResourceControlRequest  [32] IMPLICIT TriggerResourceControlRequest,
 resourceReportRequest          [33] IMPLICIT ResourceReportRequest,
 resourceReportResponse         [34] IMPLICIT ResourceReportResponse,
 scanRequest                    [35] IMPLICIT ScanRequest,
 scanResponse                   [36] IMPLICIT ScanResponse,
                                        -- [37] through [42] reserved
 sortRequest                    [43] IMPLICIT SortRequest,
 sortResponse                   [44] IMPLICIT SortResponse,
 segmentRequest                 [45] IMPLICIT Segment,
 extendedServicesRequest        [46] IMPLICIT ExtendedServicesRequest,
 extendedServicesResponse       [47] IMPLICIT ExtendedServicesResponse,
 close                          [48] IMPLICIT Close}


-- Initialize APDUs
--
```

```
  InitializeRequest ::= SEQUENCE{
        referenceId                         ReferenceId OPTIONAL,
        protocolVersion                     ProtocolVersion,
        options                             Options,
        preferredMessageSize       [5]      IMPLICIT INTEGER,
        exceptionalRecordSize      [6]      IMPLICIT INTEGER,
        idAuthentication           [7]      ANY OPTIONAL, -- see note below
        implementationId           [110]    IMPLICIT InternationalString OPTIONAL,
        implementationName         [111]    IMPLICIT InternationalString OPTIONAL,
        implementationVersion      [112]    IMPLICIT InternationalString OPTIONAL,
        userInformationField       [11]     EXTERNAL OPTIONAL,
        otherInfo                           OtherInformation OPTIONAL}
--Note:
-- For idAuthentication, the type ANY is retained for compatibility with earlier versions.
-- For interoperability, the following is recommended:
--   IdAuthentication [7] CHOICE{
--       open    VisibleString,
--       idPass  SEQUENCE {
--                   groupId       [0]      IMPLICIT InternationalString OPTIONAL,
--                   userId        [1]      IMPLICIT InternationalString OPTIONAL,
--                   password      [2]      IMPLICIT InternationalString OPTIONAL },
--       anonymous   NULL,
--       other       EXTERNAL
-- May use access control formats for 'other'.  See Appendix 7 ACC.
--
  InitializeResponse ::= SEQUENCE{
        referenceId                         ReferenceId OPTIONAL,
        protocolVersion                     ProtocolVersion,
        options                                     Options,
        preferredMessageSize       [5]      IMPLICIT INTEGER,
        exceptionalRecordSize      [6]      IMPLICIT INTEGER,
        result                     [12]     IMPLICIT BOOLEAN,    -- reject = FALSE; Accept = TRUE
        implementationId           [110]    IMPLICIT InternationalString OPTIONAL,
        implementationName         [111]    IMPLICIT InternationalString OPTIONAL,
        implementationVersion      [112]    IMPLICIT InternationalString OPTIONAL,
        userInformationField       [11]     EXTERNAL OPTIONAL,
        otherInfo                           OtherInformation OPTIONAL}
-- Begin auxiliary definitions for Init PDUs
  ProtocolVersion  ::=       [3]   IMPLICIT BIT STRING{
              version-1                     (0),     -- This bit should always be set, but does not
                                                     -- correspond to any Z39.50 version.
              version-2                     (1),     -- "Version 2 supported."
                                                     -- This bit should always be set.
              version-3                     (2)      -- "Version 3 supported."
-- Values higher than 'version-3' should be ignored. Both the Initialize request and Initialize Response APDUs
-- include a value string corresponding to the supported versions. The highest common version is selected
-- for use. If there are no versions in common, "Result" in the Init Response should indicate "reject."
-- Note: Versions 1 and 2 are identical. Systems supporting version 2 should indicate support for version
-- 1 as well, for interoperability with systems that indicate support for version 1 only (e.g. ISO 10163-1991
-- implementations).
        }
  Options  ::= [4] IMPLICIT BIT STRING{
                         search                    (0),
                         present                   (1),
                         delSet                    (2),
                         resourceReport            (3),
                         triggerResourceCtrl       (4),
                         resourceCtrl              (5),
                         accessCtrl                (6),
                         scan                      (7),
                         sort                      (8),
                         --                        (9) (reserved)
                         extendedServices          (10),
                         level-1Segmentation       (11),
                         level-2Segmentation       (12),
                         concurrentOperations      (13),
                         namedResultSets           (14)}
-- end auxiliary definitions for Init PDUs


--Search APDUs
  SearchRequest ::= SEQUENCE{
        referenceId                              ReferenceId OPTIONAL,
        smallSetUpperBound              [13]     IMPLICIT INTEGER,
        largeSetLowerBound              [14]     IMPLICIT INTEGER,
        mediumSetPresentNumber          [15]     IMPLICIT INTEGER,
        replaceIndicator                [16]     IMPLICIT BOOLEAN,
        resultSetName                   [17]     IMPLICIT InternationalString,
        databaseNames                   [18]     IMPLICIT SEQUENCE OF DatabaseName,
        smallSetElementSetNames         [100]    ElementSetNames OPTIONAL,
        mediumSetElementSetNames        [101]    ElementSetNames OPTIONAL,
        preferredRecordSyntax           [104]    IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        query                           [21]     Query,
             -- Following two parameters may be used only if version 3 is in force.
        additionalSearchInfo            [203]    IMPLICIT OtherInformation OPTIONAL,
        otherInfo                                OtherInformation OPTIONAL}


-- Query Definitions
        Query  ::=       CHOICE{
                         type-0       [0]      ANY,
                         type-1       [1]      IMPLICIT RPNQuery,
                         type-2       [2]      OCTET STRING,
                         type-100     [100]    OCTET STRING,
                         type-101     [101]    IMPLICIT RPNQuery,
                         type-102     [102]    OCTET STRING}
--
-- Definitions for RPN query
        RPNQuery ::= SEQUENCE{
                         attributeSet          AttributeSetId,
                         rpn                   RPNStructure}
--
```

```
RPNStructure ::= CHOICE{
                op              [0] Operand,
                rpnRpnOp        [1] IMPLICIT SEQUENCE{
                                        rpn1            RPNStructure,
                                        rpn2            RPNStructure,
                                        op              Operator }}
        Operand ::= CHOICE{
                attrTerm        AttributesPlusTerm,
                resultSet       ResultSetId,
                                        -- If version 2 is in force:
                                        --   - If query type is 1, one of the above two must be chosen;
                                        --   - resultAttr (below) may be used only if query type is 101.
                resultAttr      ResultSetPlusAttributes}

        AttributesPlusTerm ::= [102] IMPLICIT SEQUENCE{
                                attributes      AttributeList,
                                term            Term}
        ResultSetPlusAttributes ::= [214] IMPLICIT SEQUENCE{
                                resultSet       ResultSetId,
                                attributes      AttributeList}
        AttributeList ::=               [44]  IMPLICIT SEQUENCE OF AttributeElement
--
        Term ::= CHOICE{
                general                         [45]    IMPLICIT OCTET STRING,
                                                -- values below may be used only if version 3 is in force
                numeric                         [215]  IMPLICIT INTEGER,
                characterString                 [216]  IMPLICIT InternationalString,
                oid                             [217]  IMPLICIT OBJECT IDENTIFIER,
                dateTime                        [218]  IMPLICIT GeneralizedTime,
                external                        [219]  IMPLICIT EXTERNAL,
                integerAndUnit                  [220] IMPLICIT IntUnit,
                null                            [221] IMPLICIT NULL}

        Operator ::= [46] CHOICE{
                                 and            [0] IMPLICIT NULL,
                                 or             [1] IMPLICIT NULL,
                                 and-not        [2] IMPLICIT NULL,
                                                        -- If version 2 is in force:
                                                        --  - For query type 1, one of the above three must be chosen;
                                                        --  - prox (below) may be used only if query type is 101.
                                prox            [3] IMPLICIT ProximityOperator}
        AttributeElement  ::=  SEQUENCE{
                attributeSet            [1]             IMPLICIT AttributeSetId OPTIONAL,
                                                                -- Must be omitted if version 2 is in force.
                                                                -- If included, overrides value of attributeSet
                                                                -- in RPNQuery above, but only for this attribute.
                attributeType [120]  IMPLICIT INTEGER,
                attributeValue                  CHOICE{
                                                  numeric       [121]   IMPLICIT INTEGER,
                                                                        -- If version 2 is in force,
                                                                        -- Must select 'numeric' for attributeValue.
                complex     [224] IMPLICIT SEQUENCE{
                                  list                  [1] IMPLICIT SEQUENCE OF StringOrNumeric,
                                  semanticAction        [2] IMPLICIT SEQUENCE OF INTEGER OPTIONAL}}}


        ProximityOperator ::= SEQUENCE{
                exclusion               [1] IMPLICIT BOOLEAN OPTIONAL,
                distance                [2] IMPLICIT INTEGER,
                ordered                 [3] IMPLICIT BOOLEAN,
                relationType            [4] IMPLICIT INTEGER{
                                                lessThan                (1),
                                                lessThanOrEqual         (2),
                                                equal                   (3),
                                                greaterThanOrEqual      (4),
                                                greaterThan             (5),
                                                notEqual                (6)},
                proximityUnitCode       [5] CHOICE{
                                                known       [1] IMPLICIT KnownProximityUnit,
                                                private     [2] IMPLICIT INTEGER}}
--
    KnownProximityUnit ::= INTEGER{
                                  character     (1),
                                  word          (2),
                                  sentence      (3),
                                  paragraph     (4),
                                  section       (5),
                                  chapter       (6),
                                  document      (7),
                                  element       (8),
                                  subelement    (9),
                                  elementType   (10),
                                  byte          (11) -- Version 3 only
                                  }
-- End definitions for RPN Query


SearchResponse ::= SEQUENCE{
        referenceId                     ReferenceId OPTIONAL,
        resultCount             [23]    IMPLICIT INTEGER,
        numberOfRecordsReturned [24]    IMPLICIT INTEGER,
        nextResultSetPosition   [25]    IMPLICIT INTEGER,
        searchStatus            [22]    IMPLICIT BOOLEAN,
        resultSetStatus         [26]    IMPLICIT INTEGER{
                                                subset      (1),
                                                interim     (2),
                                                none        (3)} OPTIONAL,
        presentStatus                   PresentStatus  OPTIONAL,
        records                         Records OPTIONAL,
                -- Following two parameters may be used only if version 3 is in force.
        additionalSearchInfo    [203]   IMPLICIT OtherInformation OPTIONAL,
        otherInfo                       OtherInformation OPTIONAL}
```

```
--Retrieval APDUs
  PresentRequest ::= SEQUENCE{
        referenceId                                  ReferenceId OPTIONAL,
        resultSetId                                  ResultSetId,
        resultSetStartPoint             [30]    IMPLICIT INTEGER,
        numberOfRecordsRequested        [29]    IMPLICIT INTEGER,
        additionalRanges                [212]   IMPLICIT SEQUENCE OF Range OPTIONAL,
                      -- additionalRanges may be included only if version 3 is in force.
        recordComposition               CHOICE{
                                                simple          [19]    ElementSetNames,
                                                -- must choose 'simple' if version 2 is in force
                                                complex         [209]   IMPLICIT CompSpec} OPTIONAL,
        preferredRecordSyntax           [104]   IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        maxSegmentCount                 [204]   IMPLICIT INTEGER OPTIONAL, -- level 1 or 2
        maxRecordSize                   [206]   IMPLICIT INTEGER OPTIONAL, -- level 2 only
        maxSegmentSize                  [207]   IMPLICIT INTEGER OPTIONAL, -- level 2 only
        otherInfo                               OtherInformation OPTIONAL}
--
  Segment ::= SEQUENCE{
                      -- Segment PDU may only be used when version 3 is in force,
                      -- and only when segmentation is in effect.
        referenceId                             ReferenceId OPTIONAL,
        numberOfRecordsReturned         [24]    IMPLICIT INTEGER,
        segmentRecords                  [0]     IMPLICIT SEQUENCE OF NamePlusRecord,
        otherInfo                               OtherInformation OPTIONAL}
--
  PresentResponse ::= SEQUENCE{
        referenceId                             ReferenceId OPTIONAL,
        numberOfRecordsReturned         [24]    IMPLICIT INTEGER,
        nextResultSetPosition           [25]    IMPLICIT INTEGER,
        presentStatus                           PresentStatus,
        records                                 Records OPTIONAL,
        otherInfo                               OtherInformation OPTIONAL}
-- begin auxiliary definitions for Search and Present APDUs

-- begin definition of records
  Records ::= CHOICE{
        responseRecords                 [28]    IMPLICIT SEQUENCE OF NamePlusRecord,
        nonSurrogateDiagnostic          [130]   IMPLICIT DefaultDiagFormat,
        multipleNonSurDiagnostics       [205]   IMPLICIT SEQUENCE OF DiagRec}
--
  NamePlusRecord  ::=  SEQUENCE{
        name            [0] IMPLICIT DatabaseName OPTIONAL,
        record          [1] CHOICE{
                              retrievalRecord             [1] EXTERNAL,
                              surrogateDiagnostic         [2] DiagRec,
                                    -- Must select one of the above two, retrievalRecord or
                                    -- surrogateDiagnostic, unless 'level 2 segmentation' is in effect.
                              startingFragment            [3] FragmentSyntax,
                              intermediateFragment        [4] FragmentSyntax,
                              finalFragment               [5] FragmentSyntax}}
        FragmentSyntax ::= CHOICE{
                externallyTagged                EXTERNAL,
                notExternallyTagged             OCTET STRING}

  DiagRec ::= CHOICE{
                        defaultFormat                   DefaultDiagFormat,
                                                                -- Must choose defaultFormat if version 2 is in effect.
                        externallyDefined               EXTERNAL}

  DefaultDiagFormat::= SEQUENCE{
        diagnosticSetId         OBJECT IDENTIFIER,
        condition               INTEGER,
        addinfo                 CHOICE{
                                        v2Addinfo    VisibleString,          -- version 2
                                        v3Addinfo    InternationalString    -- version 3
                                         }}
  -- end definition of records
        Range  ::= SEQUENCE{
                startingPosition                [1] IMPLICIT INTEGER,
                numberOfRecords                 [2] IMPLICIT INTEGER}
--
        ElementSetNames ::= CHOICE {
                genericElementSetName           [0] IMPLICIT InternationalString,
                databaseSpecific                [1] IMPLICIT SEQUENCE OF SEQUENCE{
                                                        dbName          DatabaseName,
                                                        esn             ElementSetName}}

        PresentStatus           ::=     [27]    IMPLICIT INTEGER{
                                                        success         (0),
                                                        partial-1       (1),
                                                        partial-2       (2),
                                                        partial-3       (3),
                                                        partial-4       (4),
                                                        failure         (5)}

-- begin definition of composition specification
  CompSpec ::= SEQUENCE{
        selectAlternativeSyntax         [1] IMPLICIT BOOLEAN,
                                                -- See comment for recordSyntax, below.
        generic                         [2] IMPLICIT Specification OPTIONAL,
        dbSpecific                      [3] IMPLICIT SEQUENCE OF SEQUENCE{
                                                        db      [1] DatabaseName,
                                                        spec    [2] IMPLICIT Specification} OPTIONAL,
                -- At least one of generic and dbSpecific must occur, and both may occur. If both, then for
                -- any record not in the list of databases within dbSpecific, generic applies.
        recordSyntax                    [4] IMPLICIT SEQUENCE OF OBJECT IDENTIFIER OPTIONAL
                                                -- For each record, the target selects the first record syntax
                                                -- in this list that it can support.  If the list is exhausted, the
                                                -- target may select an alternative syntax if
                                                -- selectAlternativeSyntax is 'true'.
                                }
  Specification ::= SEQUENCE{
        schema                [1] IMPLICIT OBJECT IDENTIFIER OPTIONAL,
        elementSpec           [2] CHOICE{
                                    elementSetName    [1] IMPLICIT InternationalString,
                                    externalEspec     [2] IMPLICIT EXTERNAL} OPTIONAL}
-- end definition of composition specification
-- end auxiliary definitions for search and response APDUs

-- Delete APDUs
  DeleteResultSetRequest ::= SEQUENCE{
        referenceId                              ReferenceId OPTIONAL,
        deleteFunction                  [32]     IMPLICIT INTEGER{
                                                        list    (0),
                                                        all     (1)},
        resultSetList                            SEQUENCE OF ResultSetId OPTIONAL,
        otherInfo                                OtherInformation OPTIONAL}
--
  DeleteResultSetResponse ::= SEQUENCE{
        referenceId                              ReferenceId OPTIONAL,
        deleteOperationStatus           [0]      IMPLICIT DeleteSetStatus,
        deleteListStatuses              [1]      IMPLICIT ListStatuses OPTIONAL,
        numberNotDeleted                [34]     IMPLICIT INTEGER OPTIONAL,
        bulkStatuses                    [35]     IMPLICIT ListStatuses OPTIONAL,
        deleteMessage                   [36]     IMPLICIT InternationalString OPTIONAL,
        otherInfo                                OtherInformation OPTIONAL}
        ListStatuses ::= SEQUENCE OF SEQUENCE{
                              id       ResultSetId,
                              status   DeleteSetStatus}

        DeleteSetStatus ::= [33] IMPLICIT INTEGER{
                    success                               (0),
                    resultSetDidNotExist                  (1),
                    previouslyDeletedByTarget             (2),
                    systemProblemAtTarget                 (3),
                    accessNotAllowed                      (4),
                    resourceControlAtOrigin               (5),
                    resourceControlAtTarget               (6),
                    bulkDeleteNotSupported                (7),
                    notAllRsltSetsDeletedOnBulkDlte       (8),
                    notAllRequestedResultSetsDeleted      (9),
                    resultSetInUse                        (10)}
--

--Access- and Resource-control APDUs
--
  AccessControlRequest ::= SEQUENCE{
        referenceId                     ReferenceId OPTIONAL,
        securityChallenge               CHOICE{
                                              simpleForm           [37] IMPLICIT OCTET STRING,
                                              externallyDefined    [0]  EXTERNAL},
        otherInfo                       OtherInformation OPTIONAL}
```

```
AccessControlResponse ::= SEQUENCE{
        referenceId                          ReferenceId OPTIONAL,
        securityChallengeResponse  CHOICE{
                                                   simpleForm                [38]    IMPLICIT OCTET STRING,
                                                   externallyDefined         [0]     EXTERNAL} OPTIONAL,
                                                   -- Optional only in version 3; mandatory in version 2. If
                                                   -- omitted (in version 3) then diagnostic must occur.
        diagnostic              [223]        DiagRec OPTIONAL, -- Version 3 only.
        otherInfo                            OtherInformation OPTIONAL}



ResourceControlRequest ::= SEQUENCE{
        referenceId                                ReferenceId OPTIONAL,
        suspendedFlag                     [39]     IMPLICIT BOOLEAN OPTIONAL,
        resourceReport                    [40]     ResourceReport OPTIONAL,
        partialResultsAvailable           [41]     IMPLICIT INTEGER{
                                                           subset          (1),
                                                           interim         (2),
                                                           none            (3)} OPTIONAL,
        responseRequired                  [42]     IMPLICIT BOOLEAN,
        triggeredRequestFlag              [43]     IMPLICIT BOOLEAN OPTIONAL,
        otherInfo                                  OtherInformation OPTIONAL}


ResourceControlResponse ::= SEQUENCE{
        referenceId                                ReferenceId OPTIONAL,
        continueFlag                      [44]     IMPLICIT BOOLEAN,
        resultSetWanted                   [45]     IMPLICIT BOOLEAN OPTIONAL,
        otherInfo                                  OtherInformation OPTIONAL}



TriggerResourceControlRequest ::= SEQUENCE{
        referenceId                                ReferenceId OPTIONAL,
        requestedAction                   [46]     IMPLICIT INTEGER{
                                                           resourceReport      (1),
                                                           resourceControl     (2),
                                                           cancel              (3)},
        prefResourceReportFormat          [47]     IMPLICIT ResourceReportId OPTIONAL,
        resultSetWanted                   [48]     IMPLICIT BOOLEAN OPTIONAL,
        otherInfo                                  OtherInformation OPTIONAL}



ResourceReportRequest ::= SEQUENCE{
        referenceId                                ReferenceId OPTIONAL,
        opId                              [210]    IMPLICIT ReferenceId OPTIONAL,
        prefResourceReportFormat          [49]     IMPLICIT ResourceReportId OPTIONAL,
        otherInfo                                  OtherInformation OPTIONAL}
--
```

```
  ResourceReportResponse ::= SEQUENCE{
        referenceId                                  ReferenceId OPTIONAL,
        resourceReportStatus             [50]        IMPLICIT INTEGER{
                                                             success              (0),
                                                             partial              (1),
                                                             failure-1            (2),
                                                             failure-2            (3),
                                                             failure-3            (4),
                                                             failure-4            (5),
                                                             failure-5            (6),
                                                             failure-6            (7)},
        resourceReport                   [51]        ResourceReport OPTIONAL,
        otherInfo                                    OtherInformation OPTIONAL}
--
        ResourceReport            ::=    EXTERNAL
        ResourceReportId          ::=    OBJECT IDENTIFIER

--Scan APDUs
  ScanRequest ::= SEQUENCE{
        referenceId                                          ReferenceId OPTIONAL,
        databaseNames                            [3]         IMPLICIT SEQUENCE OF DatabaseName,
        attributeSet                                         AttributeSetId OPTIONAL,
        termListAndStartPoint                                AttributesPlusTerm,
        stepSize                                 [5]         IMPLICIT INTEGER OPTIONAL,
        numberOfTermsRequested                   [6]         IMPLICIT INTEGER,
        preferredPositionInResponse              [7]         IMPLICIT INTEGER OPTIONAL,
        otherInfo                                            OtherInformation OPTIONAL}

  ScanResponse ::= SEQUENCE{
        referenceId                                          ReferenceId OPTIONAL,
        stepSize                                 [3]         IMPLICIT INTEGER OPTIONAL,
        scanStatus                               [4]         IMPLICIT INTEGER {
                                                             success       (0),
                                                             partial-1     (1),
                                                             partial-2     (2),
                                                             partial-3     (3),
                                                             partial-4     (4),
                                                             partial-5     (5),
                                                             failure       (6) },
        numberOfEntriesReturned                  [5]         IMPLICIT INTEGER,
        positionOfTerm                           [6]         IMPLICIT INTEGER OPTIONAL,
        entries                                  [7]         IMPLICIT ListEntries  OPTIONAL,
        attributeSet                             [8]         IMPLICIT AttributeSetId OPTIONAL,
        otherInfo                                            OtherInformation OPTIONAL}

-- begin auxiliary definitions for Scan
  ListEntries ::= SEQUENCE{
        entries                          [1]         IMPLICIT SEQUENCE OF Entry OPTIONAL,
        nonsurrogateDiagnostics          [2]         IMPLICIT SEQUENCE OF DiagRec OPTIONAL
              -- At least one of entries and nonsurrogateDiagnostics must occur
                                 }
  Entry  ::= CHOICE {
     termInfo                    [1]    IMPLICIT TermInfo,
     surrogateDiagnostic         [2]    DiagRec}
--
  TermInfo ::= SEQUENCE {
        term                            Term,
        displayTerm              [0]    IMPLICIT InternationalString OPTIONAL,
                                            -- Presence of displayTerm means that term is not considered by
                                            -- the target to be suitable for display, and displayTerm should
                                            -- instead be displayed. 'term' is the actual term in the term list;
                                            -- 'displayTerm' is for display purposes only, and is not an actual
                                            -- term in the term list.
        suggestedAttributes             AttributeList OPTIONAL,
        alternativeTerm          [4]    IMPLICIT SEQUENCE OF AttributesPlusTerm OPTIONAL,
        globalOccurrences        [2]    IMPLICIT INTEGER OPTIONAL,
        byAttributes             [3]    IMPLICIT OccurrenceByAttributes OPTIONAL,
        otherTermInfo                   OtherInformation OPTIONAL}

  OccurrenceByAttributes ::= SEQUENCE OF SEQUENCE{
        attributes          [1]    AttributeList,
        occurrences                CHOICE{
                                        global        [2] INTEGER,
                                        byDatabase    [3] IMPLICIT SEQUENCE OF SEQUENCE{
                                            db                    DatabaseName,
                                            num            [1]    IMPLICIT INTEGER OPTIONAL,
                                            otherDbInfo           OtherInformation OPTIONAL}} OPTIONAL,
        otherOccurInfo             OtherInformation OPTIONAL}
-- end auxiliary definitions for Scan

-- Sort APDUs
SortRequest  ::= SEQUENCE{
        referenceId                     ReferenceId OPTIONAL,
        inputResultSetNames      [3]    IMPLICIT SEQUENCE OF InternationalString,
        sortedResultSetName      [4]    IMPLICIT InternationalString,
        sortSequence             [5]    IMPLICIT SEQUENCE OF SortKeySpec,
                                            -- order of occurrence is from major to minor
        otherInfo                       OtherInformation OPTIONAL}

SortResponse  ::= SEQUENCE{
        referenceId                     ReferenceId OPTIONAL,
        sortStatus               [3]    IMPLICIT INTEGER{
                                            success       (0),
                                            partial-1     (1),
                                            failure       (2)},
        resultSetStatus          [4]    IMPLICIT INTEGER{
                                            empty         (1),
                                            interim       (2),
                                            unchanged     (3),
                                            none          (4)} OPTIONAL,
        diagnostics              [5]    IMPLICIT SEQUENCE OF DiagRec OPTIONAL,
        otherInfo                       OtherInformation OPTIONAL}
```

```
-- begin auxiliary definitions for Sort
        SortKeySpec ::= SEQUENCE{
          sortElement                           SortElement,
          sortRelation               [1]        IMPLICIT INTEGER{
                                                        ascending               (0),
                                                        descending              (1),
                                                        ascendingByFrequency    (3),
                                                        descendingByfrequency   (4)},
          caseSensitivity            [2]        IMPLICIT INTEGER{
                                                        caseSensitive           (0),
                                                        caseInsensitive         (1)},
      missingValueAction             [3]        CHOICE{
                                         abort              [1] IMPLICIT NULL,
                                         null               [2] IMPLICIT NULL,
                                                                    --supply a null value for missing value
                                     missingValueData       [3] IMPLICIT OCTET STRING} OPTIONAL}

        SortElement ::=       CHOICE{
          generic                    [1] SortKey,
          datbaseSpecific            [2] IMPLICIT SEQUENCE OF SEQUENCE{
                                        databaseName      DatabaseName,
                                        dbSort            SortKey}}

        SortKey ::= CHOICE{
                sortfield            [0]        IMPLICIT InternationalString,
                                                -- An element, element-group-tag, or alias supported by the target
                                                -- and denoting a set of elements associated with each record.
                elementSpec          [1]        IMPLICIT Specification,
        sortAttributes               [2]        IMPLICIT SEQUENCE{
                                                        id      AttributeSetId,
                                                        list    AttributeList}}
-- end auxiliary definitions for sort



-- Extended Service APDUs
  ExtendedServicesRequest  ::= SEQUENCE{
        referenceId                             ReferenceId OPTIONAL,
        function                     [3]        IMPLICIT INTEGER {
                                                        create          (1),
                                                        delete          (2),
                                                        modify          (3)},
        packageType                  [4]        IMPLICIT OBJECT IDENTIFIER,
        packageName                  [5]        IMPLICIT InternationalString OPTIONAL,
                                                        -- PackageName mandatory for 'modify' or 'delete'; optional for
                                                        -- 'create'. Following four parameters mandatory for 'create'; should
                                                        -- be included on 'modify' if being modified; not needed on 'delete'.
        userId                       [6]        IMPLICIT InternationalString OPTIONAL,
        retentionTime                [7]        IMPLICIT IntUnit OPTIONAL,
        permissions                  [8]        IMPLICIT Permissions OPTIONAL,
        description                  [9]        IMPLICIT InternationalString OPTIONAL,
-- (ExtendedServiceRequest APDU continued)
        taskSpecificParameters     [10]   IMPLICIT EXTERNAL OPTIONAL,
                                                  -- Mandatory for 'create'; included on 'modify' if specific
                                                  -- parameters being modified; not necessary on 'delete'. For the
                                                  -- 'EXTERNAL,' use OID of specific ES definition and select
                                                  --  CHOICE [1]: 'esRequest'.
        waitAction                 [11]   IMPLICIT INTEGER{
                                                  wait                 (1),
                                                  waitIfPossible       (2),
                                                  dontWait             (3),
                                                  dontReturnPackage    (4)},
        elements                                  ElementSetName OPTIONAL,
        otherInfo                                 OtherInformation OPTIONAL}
--

ExtendedServicesResponse ::= SEQUENCE{
        referenceId                               ReferenceId OPTIONAL,
        operationStatus            [3]            IMPLICIT INTEGER{
                                                          done              (1),
                                                          accepted          (2),
                                                          failure           (3)},
        diagnostics                [4]            IMPLICIT SEQUENCE OF DiagRec OPTIONAL,
        taskPackage                [5]            IMPLICIT EXTERNAL OPTIONAL,
                                                          -- Use OID: {Z39-50-recordSyntax (106)} and corresponding
                                                          -- syntax. For the EXTERNAL, 'taskSpecific,' within that
                                                          -- definition, use OID of the specific es, and choose [2],
                                                          -- 'taskPackage'.
        otherInfo                                 OtherInformation OPTIONAL}

 Permissions ::= SEQUENCE OF SEQUENCE{
        userId                [1] IMPLICIT InternationalString,
        allowableFunctions    [2] IMPLICIT SEQUENCE OF INTEGER{
                                                  delete               (1),
                                                  modifyContents       (2),
                                                  modifyPermissions    (3),
                                                  present              (4),
                                                  invoke               (5)}}

Close ::= SEQUENCE{
        referenceId                               ReferenceId OPTIONAL,  -- See 3.2.11.1.5.
        closeReason                               CloseReason,
        diagnosticInformation            [3]      IMPLICIT InternationalString OPTIONAL,
        resourceReportFormat             [4]      IMPLICIT ResourceReportId OPTIONAL,
                                                          -- For use by origin only, and only on Close request;
                                                          -- origin requests target to include report in response.
        resourceReport                   [5]      ResourceReport OPTIONAL,
                                                          -- For use by target only, unilaterally on Close request;
                                                          -- on Close response may be unilateral or in response
                                                          -- to origin request.
        otherInfo                                 OtherInformation OPTIONAL}
        CloseReason ::=      [211]  IMPLICIT INTEGER{
                                        finished                    (0),
                                        shutdown                    (1),
                                        systemProblem               (2),
                                        costLimit                   (3),
                                        resources                   (4),
                                        securityViolation           (5),
                                        protocolError               (6),
                                        lackOfActivity              (7),
                                        peerAbort                   (8),
                                        unspecified                 (9)}


-- Global auxiliary definitions
        ReferenceId                 ::=          [2]    IMPLICIT OCTET STRING
        ResultSetId                 ::=          [31]   IMPLICIT InternationalString
        ElementSetName              ::=          [103]  IMPLICIT InternationalString
        DatabaseName                ::=          [105]  IMPLICIT InternationalString
        AttributeSetId              ::=                 OBJECT IDENTIFIER


-- OtherInformation
        OtherInformation     ::= [201] IMPLICIT SEQUENCE OF SEQUENCE{
                category                                [1]   IMPLICIT InfoCategory OPTIONAL,
                information                 CHOICE{
                        characterInfo                   [2]     IMPLICIT InternationalString,
                        binaryInfo                      [3]     IMPLICIT OCTET STRING,
                        externallyDefinedInfo           [4]     IMPLICIT EXTERNAL,
                        oid                             [5]     IMPLICIT OBJECT IDENTIFIER}}
--
        InfoCategory ::= SEQUENCE{
                        categoryTypeId      [1]     IMPLICIT OBJECT IDENTIFIER OPTIONAL,
                        categoryValue       [2]     IMPLICIT INTEGER}


-- Units
        -- IntUnit is used when value and unit are supplied together. Unit, alone, is used when just
        -- specifying a unit (without a value).  For example, IntUnit is used in Term, in an RPNQuery, or
        -- it can be the datatype of an element within a retrieval record. Unit (alone) would be used in an
        -- element request, when requesting data be returned according to a particular unit.

        IntUnit ::= SEQUENCE{
                value           [1] IMPLICIT INTEGER,
                unitUsed        [2] IMPLICIT Unit}
--
        Unit ::= SEQUENCE{
                unitSystem          [1] InternationalString OPTIONAL,           -- e.g. 'SI'
                unitType            [2] StringOrNumeric OPTIONAL,               -- e.g. 'mass'
                unit                [3] StringOrNumeric OPTIONAL,               -- e.g. 'kilograms'
                scaleFactor         [4] IMPLICIT INTEGER OPTIONAL               -- e.g. 9 means 10**9
                            }
```

```
--CharacterString
        InternationalString ::= GeneralString
                -- When version 2 is in force, this collapses to VisibleString. That is, only characters in the
                -- visibleString repertoire may be used. (Datatype compatibility with version 2 is not affected,
                -- because references are IMPLICIT.)  When version 3 is in force, the semantics of the
                -- GeneralString content may be altered by negotiation during initialization. If no such
                -- negotiation is in effect, then GeneralString semantics are in force.

StringOrNumeric ::= CHOICE{
        string          [1] IMPLICIT InternationalString,
        numeric         [2] IMPLICIT INTEGER}

END -- IR DEFINITIONS
```

4.2 Protocol Procedures

Protocol procedures are described in this section. Rules for extensibility and conformance requirements are specified in sections 4.3 and 4.4 respectively.

4.2.1 Presentation and Association Control Services

The Information Retrieval protocol may be used in conjunction with the presentation layer and the association control service element (ACSE).

4.2.1.1 Service Provided by the Presentation Layer

Z39.50 may use the presentation service as defined in ISO 8822 to provide a presentation connection for communication between a Z39.50 origin/target pair. The communication service that supports this protocol is a connection-oriented service defined in ISO 8822 in an established application association, in combination with ACSE, ISO 8649.

A Z39.50 origin establishes application-associations as necessary with the target. The Z39.50 application-service-element (ASE) may then use the P-DATA service defined in ISO 8822 directly to transmit Z39.50 APDUs. This provides a connection-oriented interaction between Z39.50 systems.

4.2.1.2 Association Control Services

The complete application service may include ACSE, and one or more specific application services, such as the Information Retrieval application service.

ACSE, defined in ISO 8649, is used to establish an A-association, and provides association management. The life of an A-association has three distinct phases: establishment, information transfer, and termination. ACSE provides services for the establishment and termination phases, including the selection of an application context, specifying information including the set of service elements that are valid during the information transfer phase. Prior to the exchange of Z39.50 APDUs, the Information Retrieval service user invokes the association control services required to establish an association with an application context encompassing the Information Retrieval service. The application context "basic-Z39.50-ac" is defined and registered in Appendix 2, CTX.

A single application-association can be used to support a series of Z-Associations. A single system can be engaged in multiple application associations with multiple remote systems simultaneously.

4.2.2 Protocol Model

To specify protocol procedure, the abstract, implementation-independent concepts of service-user, service-provider, and service primitive are used.

A service-provider provides a communication path between two service users. In this model, the service-provider is analogous to the application layer composed of the Z39.50 origin/target pair. The client is modeled as a service-user together with an origin, and the server is modeled as a service-user together with a target. The two service users are referred to as the origin service-user and target service-user.

A service primitive is an element of interaction between a service-user and the service-provider. There are four types of service primitives: Request, Indication, Response and Confirmation. For a confirmed service initiated by the origin (i.e., for Z39.50: Init, Search, Present, Delete, Resource-report, Sort, Scan, Extended-services) they are used as follows:

- Request - A primitive issued by the origin to the service-provider in order to invoke some procedure.
- Indication - A primitive issued by the service-provider to the target service-user to indicate that a procedure has been invoked by its peer.
- Response - A primitive issued by the target service-user to the service-provider at the completion of the procedure previously invoked by an indication.
- Confirmation - A primitive issued by the service-provider to the origin service-user to complete the procedure previously invoked by a request.

Notes:

1. For a confirmed service initiated by the target (i.e., for Z39.50: Access-control and Resource-control) the roles of origin and target are reversed.
2. For a non-confirmed service (i.e., for Z39.50: Segment, Trigger-resource-control, Close) only the Request and Indication primitives are used.

Primitives are conceptual and their use neither specifies nor precludes any specific implementation of a service. Only primitives that correspond to some element of the service involving the exchange of information between systems are defined.

From the perspective of the service-user, the service-provider is system-independent. For the exchange of protocol however, a distinction is drawn between the portion of the service-provider residing on the client and the portion of the service-provider residing on server (respectively, the origin and the target). The sequence of interactions for a confirmed service initiated by the origin is:

1. Request Primitive from origin service-user to service-provider.
2. Protocol Message from origin to target.
3. Indication Primitive from service-provider to target service-user.
4. Response Primitive from target service-user to service-provider.
5. Protocol Message from target to origin.
6. Confirmation Primitive from service-provider to origin service-user.

Notes:

1. For a confirmed service initiated by the target, the roles of origin and target are reversed.
2. For a non-confirmed service, only steps 1 through 3 apply.

The following illustrates the sequence of interactions that occur for a Search operation:

1. Search request from origin service-user to service-provider.
2. Search APDU from origin to target.
3. Search indication from service-provider to target service-user.
4. Search response from target service-user to service-provider.
5. Search-response APDU from target to origin.
6. Search confirm from service-provider to origin service-user.

The interactions between service user and service-provider, as represented by steps 1 and 6 for the client, and by steps 3 and 4 for the server, are described solely to facilitate the specification of protocols. These steps do not represent intersystem communication, and the means by which they are implemented are not constrained by this specification. For example, in an actual implementation the target service-user and service-provider might be combined in a single program, and steps 3 and 4 might not have any real physical manifestation.

4.2.3 State Tables

This section defines Information Retrieval Protocol Machines (IRPMs) in terms of state tables. For both origin and target, there are three protocol machines defined, one for the Z-Association (called the "Z-machine") and two for Z39.50 operations (called "O-machines"). One O-machine is for a Present operation and one O-machine is for any other type operation, excluding Init which is included in the Z-machine.

There is one instance of the Z-machine (within a given application association) each for the origin and target; there may be multiple concurrent instances of the O-machines.

Each state table shows the interrelationship between the state of an operation or Z-Association, the incoming events that occur in the protocol, the actions taken, and, finally, the resulting state. The state tables do not constitute a formal definition of the IRPM. They are included to provide a more precise specification of the protocol procedures. The following conventions are used in the state tables:

State Table Cells. The intersection of an incoming event (row) and a state (column) forms a cell. A blank cell represents the combination of an incoming event and a state that is not defined for the IRPM. A non-blank cell represents an incoming event and state that is defined for the IRPM. Such a cell contains one or more actions, separated by semi-colons (;). The last such action specified is always a transition to the resulting state, in parentheses.

Invalid Intersections. Blank cells indicate an invalid intersection of an incoming event and state. The state tables define correct operation only. They do not specify actions to be taken in response to incorrect operation (for example, erroneous protocol control information, incorrect protocol control actions, etc.). Such actions are not within the scope of the specification, although implementations must consider them.

Predicates. Some actions are predicated on a certain condition, or "predicate." The notation for these actions takes one of the following two forms:

:[predicate] actions:

or

:[predicate] actions else actions:

where "actions" is either a single action or multiple actions separated by semicolon. The following predicates are defined:

Predicate	Meaning
resp	"Response required" on a Resource Control PDU.
noResp	"No response required" on a Resource Control PDU.
conc	Concurrent operations in effect.
noOps	No active operations.

Variables. The following variables are used:

Variable	Meaning
<op>	An operation type (other than Init): search, present, delete, scan, sort, resource-report, Extended-services.
opCnt	Number of active operations.
retSt	Return state. An integer; the action "(retSt)" means "go to the state whose value is retSt".

Notes pertaining to the tables:

1. Access-control and resource-control events, actions, and states are distinguished according to whether they pertain to an operation or to the Z-association. (If concurrent operation is not in effect, all pertain to an operation. During initialization, all pertain to the initialization operation.) Those that pertain to an operation are reflected in the operation state table, except for those that occur during initialization (those are shown in part 1 of the Z-association table) and those that pertain to an aborted operation (those apply to part 3 of the Z-association table). Those that pertain to the Z-association are shown in part 2 of the Z-association table (except as noted in notes 4 and 5). All abbreviations for states, events, and actions for access- or resource-control beginning with "Z-" (e.g. "Z-Acc PDU") pertain to the Z-association. All others (e.g. "Acc PDU") pertain to an operation.
2. During initialization, access control or resource control requests may be received by the origin but only if the origin has indicated support (though this is not reflected in the state tables). The origin may not send Trigger-resource-control, because initialization is not complete so it has not yet been successfully negotiated. Neither the origin nor target may initiate Close during initialization.
3. "End-operation indication" is a pseudo-action by the O-machine and corresponding event to the Z-machine. The O-machine issues the indication to the Z-machine, which receives it also as an indication. Its meaning is that an operation has ended (it is necessary for the Z-machine to keep track of the number of active operations so that it will know whether there are zero, one, or multiple concurrent active operations).
4. After the origin sends a Close PDU, PDUs may arrive that were sent before the target receives the Close PDU. When the origin is in "Close sent" state, it ignores all such PDUs if they pertain to an (aborted) operation. If an Access-control request pertaining to the Z-association is received, it is similarly ignored. However, if a Resource-control request pertaining to the application is received, and if it specifies that "no response is required" it is passed to the application, because it may include useful information. If a resource-control request specifies "response required" it is ignored.
5. After the target sends a Close PDU, it ignores any received PDUs until it receives a Close PDU. When the target is in "Close Recvd" state, it may send one or more Resource-control requests before sending the Close PDU, but they must indicate "no response required".

Definition of States

Origin States

Origin States for Z-association

0. Closed: The origin is awaiting an Init request from the service-user.
1. Init Sent: The origin is awaiting an Init-response PDU from the target.
2. Acc Recvd: During initialization the origin has received an Access-control PDU and is awaiting an Access-control response from the service-user.
3. Rsc Recvd: During initialization the origin has received a resource-control PDU and is awaiting a Resource-control response from the service-user.
4. Serial Idle: The Z-association is established, there are no active operations, and ’serial operations’ is in effect.
5. Concurrent Idle: The Z-association is established, there are no active operations, and ’concurrent operations’ is in effect.
6. Serial Active: There is an active operation and ’serial operations’ is in effect.
7. Concurrent Active: There is at least one active operation, and ’concurrent operations’ is in effect.
8. Z-Acc recvd: The origin has received an Access-control PDU pertaining to the Z-association and is awaiting an Access-control response from the service-user.
9. Z-Rsc recvd: The origin has received a Resource-control PDU pertaining to the Z-association and is awaiting a Resource-control response from the service-user.
10. Close sent: The origin is awaiting a Close PDU from the target.
11. Close Received: The origin is awaiting a Close response from the service-user.

Origin States for Operation

1. *For Present operation:* Present sent: The origin is awaiting a Present-response PDU from the target. *For operation other than Present:* <Op> sent: The origin is awaiting an <Op>-response PDU from the target.
2. Rsc recvd: The origin has received a Resource-control-request PDU pertaining to the operation and is awaiting a Resource-control response from the service-user.
3. Acc recvd: The origin has received an Access-control-request PDU pertaining to the operation and is awaiting an Access-control response from the service-user.

Target States

Target States for Z-association

0. Closed: The target is awaiting an Init PDU from the origin.
1. Init recvd: The target is awaiting an Init Response from the service-user.
2. Acc Sent: During initialization the target has sent an Access-control PDU and is awaiting an Access-control-response PDU from the origin.
3. Rsc sent: During initialization the target has sent a Resource-control PDU and is awaiting a Resource-control-response PDU from the origin.
4. Serial Idle: The Z-association is established, there are no active operations, and ’serial operations’ is in effect.
5. Concurrent Idle: The Z-association is established, there are no active operations, and ’concurrent operations’ is in effect.
6. Serial Active: There is an active operation and ’serial operations’ is in effect.
7. Concurrent Active: There is at least one active operation, and ’concurrent operations’ is in effect.
8. Z-Acc sent: The target has sent an Access-control PDU pertaining to the Z-association and is awaiting an Access-control-response PDU from the origin.
9. Z-Rsc sent: The target has sent a Resource-control PDU pertaining to the Z-association and is awaiting a Resource-control-response PDU from the origin.
10. Close sent: The target is awaiting a Close PDU from the origin.
11. Close Received: The target is awaiting a Close response from the service-user.

Target States for Operation

1. *For Present operation:* Present sent: The target is awaiting a Present response from the service-user. *For operation other than Present:* <Op> sent: The target is awaiting an <Op>-response PDU from the service-user.
2. Rsc sent: The target has sent a Resource-control PDU pertaining to the operation and is

awaiting a Resource-Control-response PDU from the origin
3. Acc sent: The target has sent an Access-control PDU pertaining to the operation and is awaiting an Access-control-response PDU from the origin.

Events and Actions

Listed below are the events and actions that appear in the tables. Those corresponding to a service primitive or APDU are listed first (in alphabetical order by the abbreviation used in the tables) followed by miscellaneous actions.

Abbreviation	**Meaning**
<op> PDU	<operation type> PDU
<op> req	<operation type> request
<op> resp	<operation type> response
<op> conf	<operation type> confirm
<op> resp PDU	<operation type> Response PDU
Acc conf	Access-control confirm
Acc ind	Access-control indication
Acc PDU	Access-control PDU
Acc req	Access-control request
Acc resp	Access-control response
Acc Resp PDU	Access-control-response PDU
AnyOpPdu	Any PDU belonging to an operation
AnyPdu	Any PDU except Close
Close conf	Close confirm
Close ind	Close Indication
Close PDU	Close PDU
Close req	Close request
Close resp	Close response
EndOp ind	End-operation indication
Init conf+	Init confirm (accept)
Init conf-	Init confirm (reject)
Init ind	Init indication
Init PDU	Init PDU
Init req	Init request
Init resp PDU+	Init-response PDU (accept)
Init resp PDU-	Init-response PDU (reject)
Init resp+	Init response (accept)
Init resp-	Init response (reject)
Prsnt conf	Present confirm
Prsnt resp PDU	Present-response PDU
Prsnt resp	Present response
Rsc conf	Resource-control confirm
Rsc ind	Resource-control indication
Rsc PDU	Resource-control PDU
Rsc req	Resource-control request
Rsc resp	Resource-control response
Rsc resp PDU	Resource-control-response PDU
Seg ind	Segment Indication
Seg PDU	Segment PDU
Seg req	Segment request
Trigrc PDU	Trigger-resource-control PDU
Trigrc req	Trigger-resource-control request
Z-Acc conf	Access-control confirm (Z-association)
Z-Acc PDU	Access-control PDU (Z-association)
Z-Acc req	Access-control request (Z-association)
Z-Acc resp	Access-control response (Z-association)
Z-Acc resp PDU	Access-control-response PDU (Z-association)
Z-Rsc conf	Resource-control confirm (Z-association)
Z-Rsc PDU	Resource-control PDU (Z-association)
Z-Rsc req	Resource-control request (Z-association)
Z-Rsc req noResp	Resource-control request, "no response" (Z-association)
Z-Rsc resp	Resource-control response (Z-association)
Z-Rsc resp PDU	Resource-control-response PDU (Z-association)

Miscellaneous actions

Abbreviation	**Meaning**
Initiate <op> operation	1. Initiate an O-machine for an operation of type <op>. If <op> is Present, table 2 or 5 applies (for origin or target respectively); otherwise table 3 or 6 applies. 2. Origin: send <op> PDU. Target: issue <op> indication. 3. Set initial state for operation to 1. 4. If concurrent operations is in effect, increment opCnt by 1.
KillOps	Immediately terminate any active operations; all further PDUs pertaining to any of those operations are input to the Z-machine.
Set <variable> = <x>	Set the value of the specified variable to x.
(x)	Go to state x.
Decr	Decrement the variable opCnt by 1.
Exit	Terminate the O-machine.

Table 1, part 1: State Table for Origin Z39.50 Association: Initialization Phase

State **Event**	**Closed** **0**	**Init sent** **1**	**Acc recvd** **2**	**Rsc recvd** **3**
Init req	Init PDU; (1)			
Init resp PDU+		Init conf+; set opCnt = 0; :[conc] (5) else (4):		
Init resp PDU-		Init conf-; (0)		
Acc PDU		Acc ind; (2)		
Acc resp			Acc resp PDU; (1)	
Rsc PDU		Rsc ind; :[resp] (3) else (1):		
Rsc resp				Rsc resp PDU; (1)

Table 1, part 2: State Table for Origin Z39.50 Association: Processing Phase

State **Event**	**Serial Idle** **4**	**Concurrent Idle** **5**	**Serial Active** **6**	**Concurrent Active** **7**	**Z-Acc recvd** **8**	**Z-Rsc recvd** **9**
<op> req	Initiate <op> operation; (6)	Initiate <op> operation; (7)		Initiate <op> operation; (7)	Initiate <op> operation; set RetSt = 7; (8)	Initiate <op> operation; set RetSt = 7; (9)
EndOp ind			(4)	Decr; :[noOps] (5) else (7):	Decr; :[noOps] set RetSt = 5:; (8)	Decr; :[noOps] set RetSt = 5:; (9)
Z-Acc PDU		Acc ind; set RetSt = 5; (8)		Acc ind; set RetSt = 7; (8)		
Z-Acc resp					Acc resp PDU; (RetSt)	
Z-Rsc PDU		Rsc ind; :[resp] set RetSt = 5; (9) else (5):		Rsc ind; :[resp] set RetSt = 7; (9) else (7):		
Z-Rsc resp						Rsc Resp PDU; (RetSt)
Close req	Close PDU; (10)	Close PDU; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)
Close PDU	Close ind; (11)	Close ind; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)

Table 1, part 3: State Table for Origin Z39.50 Association: Termination Phase

State / Event	Close sent 10	Close Recvd 11
AnyOpPdu	(10)	
Z-Rsc PDU	:[noResp] Rsc ind:; (10)	
Z-Acc PDU	(10)	
Close resp		Close PDU; (0)
Close PDU	Close conf; (0)	

Table 2: State Table for Origin Present Operation

State / Event	Present sent 1	Rsc recvd 2	Acc recvd 3
Rsc PDU	Rsc ind; :[resp] (2) else (1):		
Rsc resp		Rsc resp PDU; (1)	
Acc PDU	Acc ind; (3)		
Acc resp			Acc resp PDU; (1)
Trigrc req	Trigrc PDU; (1)		
Seg PDU	Seg ind; (1)		
Prsnt resp PDU	Prsnt conf; EndOp ind; exit		

Table 3: State Table for Origin Operation Other Than Present

State / Event	<op> sent 1	Rsc recvd 2	Acc recvd 3
Rsc PDU	Rsc ind; :[resp] (2) else (1):		
Rsc resp		Rsc resp PDU; (1)	
Acc PDU	Acc ind; (3)		
Acc resp			Acc resp PDU; (1)
Trigrc req	Trigrc PDU; (1)		
<op> resp PDU	<op> conf; EndOp ind; exit		

Table 4, part 1: State Table for Target Z39.50 Association: Initialization Phase

State **Event**	**Closed** **0**	**Init recvd** **1**	**Acc sent** **2**	**Rsc sent** **3**
Init PDU	Init ind; (1)			
Init resp+		Init resp PDU+; set opCnt =0; :[conc] (5) else (4):		
Init resp-		Init resp PDU-; (0)		
Acc req		Acc PDU; (2)		
Acc resp PDU			Acc conf; (1)	
Rsc req		Rsc PDU; :[resp] (3) else (1):		
Rsc resp PDU				Rsc conf; (1)

Table 4, part 2: State Table for Target Z39.50 Association: Processing Phase

State **Event**	**Serial Idle** **(4)**	**Concur-rent Idle** **5**	**Serial Active** **6**	**Concurrent Active** **7**	**Z-Acc sent** **8**	**Z-Rsc sent** **9**
<op> PDU	Initiate <op> operation; (6)	Initiate <op> operation; (7)		Initiate <op> operation; (7)	Initiate <op> operation; set RetSt = 7; (8)	Initiate <op> operation; set RetSt = 7; (9)
EndOp ind			(4)	Decr; :[noOps] (5) else (7):	Decr; :[noOps] set RetSt = 5:; (8)	Decr; :[noOps] set RetSt = 5:; (9)
Z-Acc req		Acc PDU; set RetSt = 5; (8)		Acc PDU; set RetSt = 7; (8)		
Z-Acc resp PDU					Acc conf; (RetSt)	
Z-Rsc req		Rsc PDU; :[resp] set RetSt = 5; (9) else (5):		Rsc PDU; :[resp] set RetSt = 7; (9) else (7):		
Z-Rsc resp PDU						Rsc conf; (RetSt)
Close req	Close PDU; (10)	Close PDU; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)
Close PDU	Close ind; (11)	Close ind; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)

Table 4, part 3: State Table for Target Z39.50 Association: Termination Phase		
State Event	**Close sent 10**	**Close Recvd 11**
AnyPdu	(10)	
Z-Rsc req noResp		Rsc PDU; (10)
Close resp		Close PDU; (0)
Close PDU	Close conf; (0)	

Table 5: State Table for Target Present Operation			
State Event	**Present recvd 1**	**Rsc sent 2**	**Acc sent 3**
Rsc req	Rsc PDU; :[resp] (2) else (1):		
Rsc resp PDU		Rsc conf; (1)	
Acc req	Acc PDU; (3)		
Acc resp PDU			Acc conf; (1)
Trigrc PDU	Trigrc ind; (1)		
Seg req	Seg PDU; (1)		
Prsnt resp	Prsnt resp PDU; EndOp ind; exit		

Table 6: State Table for Target Operation Other Than Present			
State Event	**<op> recvd 1**	**Rsc sent 2**	**Acc sent 3**
Rsc req	Rsc PDU; :[resp] (2) else (1):		
Rsc resp PDU		Rsc conf; (1)	
Acc req	Acc PDU; (3)		
Acc resp PDU			Acc conf; (1)
Trigrc PDU	Trigrc ind; (1)		
<op> resp	<op> resp PDU; EndOp ind; exit		

4.2.4 Protocol Errors

Any event not listed in the tables of section 4.2.3 is not valid and is considered to be a protocol error. With exceptions specified in section 4.3, incorrectly formatted APDUs or APDUs with invalid data are also considered to be protocol errors. This standard does not specify the actions to be taken upon detection of protocol errors. An application context may contain such a specification.

Additional conditions that may be treated as protocol errors are described in 4.4.2.2.

4.3 Rules for Extensibility

All syntactical errors in received APDUs are considered to be protocol errors except for the following case: Unknown data elements, and unknown options within the Options data element, will be ignored on received Init APDUs.

4.4 Conformance

A system claiming to implement the procedures in this standard shall comply with the conformance requirements in 4.4.1. These requirements are elaborated in 4.4.2.

4.4.1 General Conformance Requirements

The system shall:

a) Act in the role of origin or target.
b) Support the Init, Search, and Present services. See 4.4.2.2.1.
c) Support the syntax in 4.1.
d) Support the Type-1 Query. See 4.4.2.2.2.
e) Support (at minimum) version 2 of the protocol.
f) Follow the procedures specified in sections 3, 4.1, 4.2, and 4.3.
g) Assign values to APDU data elements according to the procedures of sections 3 and 4.1.

4.4.2 Specific Conformance Requirements

4.4.2.1 provides a table of Z39.50 features for which 4.4.2.2 specifies conformance requirements. In particular, conformance requirements are described as they pertain to version 2 and version 3 respectively.

4.4.2.1 Z39.50 Features

The following table of Z39.50 features indicates the applicable protocol version (2 or 3), a reference to a description of the feature, and a reference to the section within 4.4.2.2 that describes conformance requirements for the feature. The "item" column is used by the sections within 4.4.2.2 to refer back to the table.

Item	Feature	Version	Reference	Conformance
1	**Init Service**	V2 and V3	3.2.1.1	4.4.2.2.1
2	**Search Service**	V2 and V3	3.2.2.1	4.4.2.2.1
3	Query type-1	V2 and V3	3.7	4.4.2.2.2
4	Multiple attribute sets	V3	Note 1	4.4.2.2.3
5	Multiple data types for search term	V3	Note 2	4.4.2.2.3
6	Complex attribute values	V3	Note 3	4.4.2.2.3
7	Result set restriction	V3	3.7	4.4.2.2.3
8	Proximity	V3	3.7.2	4.4.2.2.4
9	Query type-101	V2 and V3	3.7	4.4.2.2.4
10	Query types 0, 2, 100	V2 and V3	3.2.2.1.1	4.4.2.2.4
11	Query type 102	V3	3.2.2.1.1	4.4.2.2.5
12	Additional-search-information parameter in Search request and response	V3	3.2.2.1.12	4.4.2.2.6

Item	Feature	Version	Reference	Conformance
13	Named result sets	V2 and V3	3.2.2.1.3	4.4.2.2.23
14	**Present Service**	V2 and V3	3.2.3.1	4.4.2.2.1
15	Additional-ranges and Comp-spec parameters on Present request	V3	3.2.3.1.2, 3.2.3.1.6	4.4.2.2.7
16	Max- segment-count, -segment-size, -record-size parameters on Present request	V3	3.2.3.1.7	4.4.2.2.8
17	Diagnostic format -- default form	V2 and V3	Note 4	4.4.2.2.9
18	Diagnostic format -- external form	V3	Note 4	4.4.2.2.9
19	addinfo type VisibleString	V2, V3	Note 5	4.4.2.2.10
20	addinfo type InternationalString	V3	Note 5	4.4.2.2.10
21	Multiple non-surrogates in Search or Present response	V3	Note 6	4.4.2.2.11
22	**Segment Service**	V3	3.2.3.2	4.4.2.2.12
23	Level-1 segmentation	V3	3.3.2	4.4.2.2.12
24	Level-2 segmentation	V3	3.3.3	4.4.2.2.12
25	**Delete Service**	V2 and V3	3.2.4.1	4.4.2.2.13
26	failure-10 value of Delete-list-status on Delete response	V3	3.2.4.1.4	4.4.2.2.15
27	**Access-control Service**	V2 and V3	3.2.5.1	4.4.2.2.14
28	Security-challenge-response and diagnostic in Access-control response	V3	Note 7	4.4.2.2.16
29	**Resource-control Service**	V2 and V3	3.2.6.1	4.4.2.2.14
30	**Trigger-resource-control Service**	V2 and V3	3.2.6.2	4.4.2.2.13
31	**Resource-report Service**	V2 and V3	3.2.6.3	4.4.2.2.13
32	Op-id parameter of Resource-report-request	V3	3.2.6.3.2	4.4.2.2.17
33	failure-5 and failure-6 values of Resource-report-status in Resource-report response	V3	3.2.6.3.3	4.4.2.2.18
34	**Sort Service**	V2 and V3	3.2.7.1	4.4.2.2.13
35	**Scan Service**	V2 and V3	3.2.8.1	4.4.2.2.13
36	**Extended-Services Service**	V2 and V3	3.2.9.1	4.4.2.2.13
37	**Close Service**	V3	3.2.11.1	4.4.2.2.19
38	Explain facility	V2 and V3	3.2.10	4.4.2.2.20
39	Other-information (in a request or response other than Scan, Sort, or Extended Services)	V3	Note 8	4.4.2.2.6
40	Other-information in Scan, Sort, and ES	V2 and V3	Note 8	4.4.2.2.21

Item	Feature	Version	Reference	Conformance
41	Concurrent Operations	V3	3.5	4.4.2.2.22
42	InternationalString full use of GeneralString repertoire	V3	Note 9	4.4.2.2.24
43	Reference Id	V2 and V3	3.4	4.4.2.2.25

Notes:

(1) In version 2 a type-1 query includes a single, global attribute set id, which identifies an attribute set definition that pertains to all of the attributes within the query. In version 3 a type-1 query also includes a global attribute set id, but in addition, each attribute within the query may also be qualified with an attribute set id (which, If included, overrides the global attribute set id).

(2) In version 2 a search term must be of ASN.1 type OCTET STRING. In version 3 it may be any of the following: OCTET STRING, INTEGER, InternationalString, OBJECT IDENTIFIER, GeneralizedTime, EXTERNAL, IntUnit, or NULL.

(3) In version 2, in a type-1 query, an attribute value must be numeric (i.e. ASN.1 type INTEGER). In version 3, an attribute value may be numeric or 'complex'. The complex form may include multiple values, each either numeric or character string, and a semantic action indicator (corresponding to some semantic action defined within the attribute set definition).

(4) See introductory text of Appendix ERR.

(5) In version 2, when using default diagnostic format, the addInfo parameter must be ASN.1 type VisibleString. In version 3 it may be type InternationalString.

(6) In version 2, a Search or Present response may include at most a single non-surrogate diagnostic record. In version 3 a Search or Present response may include multiple non-surrogate diagnostic records. (Responses other than Search or Present that include diagnostics may include multiple non-surrogate diagnostics regardless of version.)

(7) In version 2, in the Access control response, securityChallengeResponse must occur, and no diagnostic may occur. In version 3, securityChallengeResponse may be omitted, if the parameter 'diagnostic' is present.

(8) In version 2, the parameter otherInformation may be used only in Scan, Sort, and Extended Services requests and responses. In version 3 it may be used in any request or response.

(9) See definition of InternationalString in ASN.1 for APDUs.

4.4.2.2 Detailed Requirements

4.4.2.2.1 Init, Search, and Present Services *(See items 1, 2 and 14 above.)*

A system must support the Init, Search, and Present services.

This means that an origin must be capable of sending Init, Search, and Present requests and receiving the respective responses. A target must respond properly to Init, Search, and Present requests with respective responses.

An origin may indicate (via option bits) during initialization that it does not intend to utilize the Present service during the Z-association; this does not constitute non-conformance. If, however, an origin indicates that it does intend to utilize the Present service, and the target refuses, this does constitute non-conformance on the part of the target.

This requirement is independent of version.

4.4.2.2.2 Type-1 Query *(See item 3 above.)*

An origin must be capable of formulating a type-1 query within a Search request, and a target should expect to receive a type-1 query.

An origin or target may support other query types. If the origin fails to send a type-1 query during a Z-association, this does not constitute non-conformance on the part of the origin. If, however, the origin does send a type-1 query and the target responds with a diagnostic indicating "query type not supported" this does constitute non-conformance on the part of the target.

This requirement does not mean that any specific feature of the type-1 query must be supported. A target that receives a type-1 query that conforms to the type-1 query syntax but which includes a feature that it does not support must not treat this condition as a protocol error (but instead should return an appropriate diagnostic, however, that diagnostic must not indicate "query type not supported").

This requirement is independent of version.

4.4.2.2.3 Multiple attribute sets, Multiple data types for search term, Complex attribute values, result set restriction, and Proximity *(See items 4, 5, 6, 7, and 8 above.)*

For version 2, the origin may not use any of these features in a type-1 query. If target receives a type-1 query with any of these features, it may treat this condition as a protocol error.

For version 3, the origin may but is not required to use any of these features in a type-1 query. The target should expect type-1 queries to include any or all of these features, but is not required to support any of these features. If the target receives a type-1 query which includes any of these feature that it does not support, it must not treat this condition as a protocol error (but rather should return an appropriate diagnostic).

4.4.2.2.4 Query types 0, 2, 100, and 101 *(See items 9 and 10 above.)*

An origin is not required to support queries of any of these types. A target should expect to receive, but need not support queries of these types. If a target receives a query of one of these types that it does not support it must not treat this condition as a protocol error but instead should return a diagnostic indicating that the query type is not supported.

This requirement is independent of version.

4.4.2.2.5 Query Type-102 *(See item 11 above.)*

For version 2, an origin may not use the type-102 query. If a target receives a type-102 query it may treat this condition as a protocol error.

For version 3, an origin may, but need not support the type-102 query. A target should expect to receive, but need not support, type-102 queries; if it receives a type-102 query it must not treat this condition as a protocol error.

Note: Z39.50-1995 lists type-102 as a valid query type (for version 3) but does not include a definition.

4.4.2.2.6 Additional-search-information parameter in Search request or response; Other-information parameter in any request or response other than Scan, Sort, or Extended Services *(See items 12 and 39 above.)*

For version 2, a system may not use these parameters; if a system receives one of these parameters it may treat this condition as a protocol error.

For version 3, a system is never required to use any of these parameters. However, a system should expect to receive these parameters, but is not required to interpret or process the information contained within the any of these parameter.

4.4.2.2.7 Additional-ranges and Comp-spec parameters on Present request *(See item 15 above.)*

For version 2, the origin may not use these parameters. If the target receives one of these parameters it may treat this condition as a protocol error.

For version 3, the origin is not required to, but may use either of these parameters. The target should expect to receive, but need not support either of these parameters. If the target receives but does not support one of these parameters, it should not treat this condition as a protocol error (but instead should return an appropriate status value and/or diagnostic).

4.4.2.2.8 Max-segment-count, Max-segment-size, and Max-record-size parameters on Present request *(See item 16 above.)*

For version 2, as well as for version 3 when segmentation is not in effect, the origin may not use these parameters; if the target receives any of these parameters it may treat this condition as a protocol error.

For version 3:

- If level-1 segmentation is in effect:
  - -- The origin may but is not required to support Max-segment-count. The target should expect to receive, but need not support Max-segment-count. If the target receives but does not support Max-segment-count, it must not treat this condition as a protocol error (but instead should return an appropriate status value and/or diagnostic).

  - -- The origin may not use Max-segment-size or Max-record-size. If target receives either it may treat this condition as a protocol error.
- If level-2 segmentation is in effect:
  - -- The origin may but is not required to support any of these three parameters. The target should expect to receive, but need not support any of these parameters. If the target receives but does not support a parameter, it must not treat this condition as a protocol error (but instead should return an appropriate status value and/or diagnostic).

4.4.2.2.9 Diagnostic format *(See items 17 and 18 above.)*

For version 2, the target may send diagnostics in a Search or Present response using the default form only. If the origin receives a diagnostic which does not conform to the default form, it may treat this condition as a protocol error.

Note: This rule applies to Search and Present responses only. Responses other than Search or Present that include diagnostics are not affected.

For version 3, the target may send diagnostics using the default or external form. The origin should expect to receive diagnostics in either form.

4.4.2.2.10 Addinfo of default diagnostic format *(See items 19 and 20 above.)*

For version 2, when the target sends a diagnostic in a Search or Present response using the default form, the addinfo parameter must be of ASN.1 type VisibleString. If the origin receives a diagnostic that violates this rule, it may treat this condition as a protocol error.

For version 3 the addinfo parameter may be of either type VisibleString or InternationalString.

4.4.2.2.11 Multiple non-surrogates in Search or Present response *(See item 21 above.)*

For version 2, the target must not include multiple non-surrogate diagnostics in a Search or Present response; if it does so, the origin may treat this condition as a protocol error.

Note: This rule applies to Search and Present responses only. There are responses other than Search or Present that include diagnostics, and these are not affected.

For version 3, the target may (but is not required to) include multiple non-surrogate diagnostics in a Search or Present response and if it does, the origin must not treat this condition as a protocol error.

4.4.2.2.12 Segmentation *(See items 22, 23, and 24 above.)*

For version 2, as well as for version 3 when segmentation is not in effect, the target may not send a Segment request, and if it does, the origin may treat this condition as a protocol error.

For version 3, level-1 or level-2 segmentation may be negotiated, however neither the target not the origin is required to support segmentation.

4.4.2.2.13 Delete service, Trigger-resource-control service, Resource-report service, Sort service, Scan service, and Extended-Services service *(See items 25, 30, 31, 34, 35, and 36 above.)*

A system is not required to support any of these services. They are independently negotiable. If the target receives a request of one of these types and the respective service is not in effect, it may treat this condition as a protocol error.

This requirement is independent of version.

4.4.2.2.14 Access-control and Resource-control services *(See items 27 and 29 above.)*

A system is not required to support either of these services. They are independently negotiable. If the origin receives an Access-control or Resource-control request and the respective service is not in effect (or if the request occurs while the origin is awaiting an Init response and the origin has not proposed the respective option in the Init request), it may treat this condition as a protocol error.

This requirement is independent of version.

4.4.2.2.15 'failure-10' value of Delete-list-status on Delete response *(See item 26 above.)*

For version 2, the target may not return this value; if it does the origin may treat this condition as a protocol error.

For version 3, the target may return this value.

4.4.2.2.16 Security-challenge-response and Diagnostic in Access-control response *(See item 28 above.)*

For version 2, the origin must include in the Access-control response the parameter Security-challenge-response, and may not include a diagnostic.

If the target receives an Access-control response that violates this rule it may treat this condition as a protocol error.

For version 3, the origin may include a diagnostic, and if so, the parameter securityChallengeResponse may be omitted.

4.4.2.2.17 Op-id parameter of Resource-report request *(See item 32 above.)*

For version 2, the origin may not use this parameter; if the target receives this parameter it may treat this condition as a protocol error.

For version 3, the origin may, but is not required to include this parameter. The target should expect to receive, but need not support the parameter. If the target receives but does not support this parameter, it should not treat this condition as a protocol error (but instead should return an appropriate status).

4.4.2.2.18 failure-5 and failure-6 Resource-report-status in Resource-report response *(See item 33 above.)*

For version 2, the target may not return either value for this status; if it does the origin may treat this condition as a protocol error.

For version 3, the target may return either value.

4.4.2.2.19 Close service *(See item 37 above.)*

For version 2, the Close service may not be used. If a system receives a Close request, it may treat this condition as a protocol error.

For version 3, a system must expect to receive a Close request, and must be capable of responding with a Close response. A system is not required to send a Close request.

4.4.2.2.20 Explain facility *(See item 38 above.)*

There are no conformance requirements pertaining to the Explain facility, either for version 2 or version 3. A system may choose to support or not support Explain.

Note that implementation of Explain requires, at minimum, support for searching the Explain database and for the Explain record syntax. This standard does not require support for searching any particular database or support for any particular record syntax.

4.4.2.2.21 Other-information parameter in Scan, Sort, and Extended Services request *(See item 40 above.)*

The parameter Other-information may occur in a Scan, Sort, or Extended Services request or response. A system should expect to receive this parameter, but is not required to interpret or process the information contained within the parameter.

This requirement is independent of version.

4.4.2.2.22 Concurrent Operations *(See item 41 above.)*

For version 2, as well as for version 3 when concurrent operations is not in effect, if an origin attempts to initiate concurrent operations (i.e. attempts to initiate an operation when an operation is already active), the target may treat this as a protocol error.

For version 3, a system may choose to support or not to support concurrent operations.

4.4.2.2.23 Named Result sets *(See item 13 above.)*

A system may choose to support or not support named result sets. If the target receives a Search request where the value of the parameter Result-set-id is other than ’default’ and the target does not support named result sets, the target should not treat this condition as a protocol error but should instead return an appropriate diagnostic.

This requirement is independent of version.

4.4.2.2.24 InternationalString Definition *(See item 42 above.)*

For version 2, a value of a parameter of ASN.1 type InternationalString must conform to the VisibleString definition. A system which receives a value that violates this rule may treat this condition as a protocol error.

For version 3, a value of a parameter of ASN.1 type InternationalString must conform to the GeneralString definition. A system which receives a value that does not conform to the VisibleString definition (but does conform to the GeneralString definition) must not treat this condition as a protocol error.

4.4.2.2.25 Reference-id *(See item 43 above.)*

For both version 2 and version 3, an origin may choose to support or not support the Reference-id parameter; a target must support the Reference-id parameter. Note, however, for version 3, origin support of concurrent operations (see 4.4.2.2.23) implies support for the reference-id parameter.