```
    serialNumber       CertificateSerialNumber }

CertificateSerialNumber ::= INTEGER

OCSPResponse ::= SEQUENCE {
   responseStatus         OCSPResponseStatus,
   responseBytes          [0] EXPLICIT ResponseBytes OPTIONAL }

OCSPResponseStatus ::= ENUMERATED {
    successful            (0),      --Response has valid confirmations
    malformedRequest      (1)       --Illegal confirmation request
}

ResponseBytes ::=       SEQUENCE {
    responseType   RESPONSE.&Type({SupportedResponses}),
    response       RESPONSE.&Value({SupportedResponses}{ @responceType })
}

RESPONSE ::= CLASS {
     &Type    OBJECT IDENTIFIER UNIQUE,
     &Value
}
WITH SYNTAX { SYNTAX &Value IDENTIFIED BY &Type }

SupportedResponses RESPONSE ::= { ..., -- extensible
                                  basicOCSPResponse }

basicOCSPResponse RESPONSE ::= {
  SYNTAX          BasicOCSPResponse
  IDENTIFIED BY   id-pkix-ocsp-basic
}

BasicOCSPResponse ::= SEQUENCE {
    tbsResponseData      ResponseData,
    signatureAlgorithm   AlgorithmIdentifier,
    signature            BIT STRING,
    certs                [0] EXPLICIT SEQUENCE OF Certificate }

ResponseData ::= SEQUENCE {
    version              [0] EXPLICIT Version DEFAULT v1,
    responderID              ResponderID,
    producedAt               GeneralizedTime,
    responses                SEQUENCE OF SingleResponse,
    responseExtensions   [1] EXPLICIT Extensions   -- nonce
}

ResponderID ::= CHOICE {
    byKey    [2] KeyHash }

KeyHash ::= OCTET STRING -- SHA-1 hash of responder's public key
                         -- (excluding the tag and length fields)

SingleResponse ::= SEQUENCE {
    certID                       CertID,
    certStatus                   CertStatus,
    thisUpdate                   GeneralizedTime,
    singleExtensions     [1]     EXPLICIT Extensions
                                 -- confirmationTime and ocspStatusCode
}

CertStatus ::= CHOICE {
    good                 [0]     IMPLICIT NULL,
    revoked              [1]     IMPLICIT RevokedInfo,
    unknown              [2]     IMPLICIT UnknownInfo }

RevokedInfo ::= SEQUENCE {
    revocationTime               GeneralizedTime,
    revocationReason     [0]     EXPLICIT CRLReason }

UnknownInfo ::= NULL

AlgorithmIdentifier     ::=  SEQUENCE {
   algorithm             ALGORITHM-ID.&id({SupportedAlgorithms}),
   parameters            ALGORITHM-ID.&Type({SupportedAlgorithms}
                                           { @algorithm }) }

ALGORITHM-ID ::= CLASS {
     &id    OBJECT IDENTIFIER UNIQUE,
     &Type
}
WITH SYNTAX { OID &id PARMS &Type }

SupportedAlgorithms ALGORITHM-ID ::= { ..., -- extensible
                                       rsaSHA-256 }

rsaSHA-256 ALGORITHM-ID ::= { OID sha256WithRSAEncryption PARMS NULL }

pkcs-1 OBJECT IDENTIFIER ::= {
     iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) 1 }

sha256WithRSAEncryption OBJECT IDENTIFIER  ::=  { pkcs-1 11 }
```