NFC Data Exchange Format (NDEF)

Technical Specification

NFC Forum^{TM}

NDEF 1.0

NFCForum-TS-NDEF_1.0

2006-07-24

Contents

Figures

Tables

Test Requirements

1 Overview

The International Standard ISO/IEC 18092, Near Field Communication – Interface and Protocol (NFCIP-1), defines an interface and protocol for simple wireless interconnection of closely coupled devices operating at 13.56 MHz.

The NFC Data Exchange Format (NDEF) specification defines a message encapsulation format to exchange information, e.g. between an NFC Forum Device and another NFC Forum Device or an NFC Forum Tag.

NDEF is a lightweight, binary message format that can be used to encapsulate one or more application-defined payloads of arbitrary type and size into a single message construct. Each payload is described by a type, a length, and an optional identifier.

Type identifiers may be URIs, MIME media types, or NFC-specific types. This latter format permits compact identification of well-known types commonly used in NFC Forum applications, or self-allocation of a name space for organizations that wish to use it for their own NFC-specific purposes.

The payload length is an unsigned integer indicating the number of octets in the payload. A compact, short-record layout is provided for very small payloads.

The optional payload identifier enables association of multiple payloads and cross-referencing between them.

NDEF payloads may include nested NDEF messages or chains of linked chunks of length unknown at the time the data is generated.

NDEF is strictly a message format, which provides no concept of a connection or of a logical circuit, nor does it address head-of-line problems.

1.1 Objectives

The NFC Data Exchange Format (NDEF) specification is a common data format for NFC Forum Devices and NFC Forum Tags.

The NFC Data Exchange Format specification defines the NDEF data structure format as well as rules to construct a valid NDEF message as an ordered and unbroken collection of NDEF records. Furthermore, it defines the mechanism for specifying the types of application data encapsulated in NDEF records.

The NDEF specification defines only the data structure format to exchange application or service specific data in an interoperable way, and it does not define any record types in detail—record types are defined in separate specifications.

This NDEF specification assumes a reliable underlying protocol and therefore this specification does not specify the data exchange between two NFC Forum Devices or the data exchange between an NFC Forum Device and an NFC Forum Tag. Readers are encouraged to review the NFCIP-1 transport protocol [ISO/IEC 18092].

An example of the use of NDEF is when two NFC Forum Devices are in proximity, an NDEF message is exchanged over the NFC Forum LLCP protocol. When an NFC Forum Device is in proximity of an NFC Forum Tag, an NDEF message is retrieved from the NFC Forum Tag by means of the NFC Forum Tag protocols. The data format of the NDEF message is the same in these two cases so that an NFC Forum Device may process the NDEF information independent of the type of device or tag with which it is communicating.

Because of the large number of existing message encapsulation formats, record marking protocols, and multiplexing protocols, it is best to be explicit about the design goals of NDEF and, in particular, about what is outside the scope of NDEF.

1.1.1 Design Goals

The design goal of NDEF is to provide an efficient and simple message format that can accommodate the following:

1. Encapsulating arbitrary documents and entities, including encrypted data, XML documents, XML fragments, image data like GIF and JPEG files, etc.
2. Encapsulating documents and entities initially of unknown size. This capability can be used to encapsulate dynamically generated content or very large entities as a series of chunks.
3. Aggregating multiple documents and entities that are logically associated in some manner into a single message. For example, NDEF can be used to encapsulate an NFC-specific message and a set of attachments of standardized types referenced from that NFC-specific message.
4. Compact encapsulation of small payloads should be accommodated without introducing unnecessary complexity to parsers.

To achieve efficiency and simplicity, the mechanisms provided by this specification have been deliberately limited to serve these purposes. NDEF has not been designed as a general message description or document format such as MIME or XML. Instead, NFC applications can take advantage of such formats by encapsulating them in NDEF messages.

1.1.2 Anti-Goals

The following list identifies items outside the scope of NDEF:

1. NDEF does not make any assumptions about the types of payloads that are carried within NDEF messages or about the message exchange patterns implied by such messages.
2. NDEF does not in any way introduce the notion of a connection or a logical circuit (virtual or otherwise).
3. NDEF does not attempt to deal with head-of-line blocking problems that might occur when using stream-oriented protocols like TCP.

1.2 References

[ISO/IEC 18092]	ISO/IEC 18092, "Information Technology- Telecommunications and information exchange between systems- Near Field Communication - Interface and Protocol (NFCIP-1)".
[NFC RTD]	"NFC Record Type Definition (RTD) Specification", NFC Forum, 2006.
[RFC 1700]	Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, October 1994.
[RFC 1900]	B. Carpenter, Y. Rekhter, "Renumbering Needs Work", RFC 1900, IAB, February 1996.
[RFC 2046]	N. Freed, N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types" RFC 2046, Innosoft, First Virtual, November 1996.

[RFC 2047]	K. Moore, “MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text”, RFC 2047, University of Tennessee, November 1996.
[RFC 2048]	N. Freed, J. Klensin, J. Postel, “Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures”, RFC 2048, Innosoft, MCI, ISI, November 1996.
[RFC 2119]	S. Bradner, “Key words for use in RFCs to Indicate Requirement Levels”, RFC 2119, Harvard University, March 1997. http://www.apps.ietf.org/rfc/rfc2119.html
[RFC 2616]	R. Fielding, J. Gettys, J. C. Mogul, H. F. Nielsen, T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1”, RFC 2616, U.C. Irvine, DEC W3C/MIT, DEC, W3C/MIT, W3C/MIT, January 1997.
[RFC 2717]	R. Petke, I. King, “Registration Procedures for URL Scheme Names”, BCP: 35, RFC 2717, UUNET Technologies, Microsoft Corporation, November 1999.
[RFC 2718]	L. Masinter, H. Alvestrand, D. Zigmond, R. Petke, “Guidelines for new URL Schemes”, RFC 2718, Xerox Corporation, Maxware, Pirsenteret, WebTV Networks, Inc., UUNET Technologies, November 1999.
[RFC 2732]	R. Hinden, B. Carpenter, L. Masinter, “Format for Literal IPv6 Addresses in URL's”, RFC 2732, Nokia, IBM, AT&T, December 1999.
[RFC 3023]	M. Murata, S. St. Laurent, D. Kohn, “XML Media Types” RFC 3023, IBM Tokyo Research Laboratory, simonstl.com, Skymoon Ventures, January 2001.
[RFC 3986]	T. Berners-Lee, R. Fielding, L. Masinter, “Uniform Resource Identifiers (URI): Generic Syntax”, RFC 3986, MIT/LCS, U.C. Irvine, Xerox Corporation, January 2005. http://www.apps.ietf.org/rfc/rfc3986.html
[URI SCHEME]	List of Uniform Resource Identifier (URI) schemes registered by IANA is available at:http://www.iana.org/assignments/uri-schemes

1.3 Administration

The NFC Forum Data Exchange Format Specification is an open specification supported by the Near Field Communication Forum, Inc., located at:

401 Edgewater Place, Suite 600
Wakefield, MA, 01880

Tel.: +1 781-876-8955
Fax: +1 781-224-1239

http://www.nfc-forum.org/

The Devices technical working group maintains this specification.

1.4 Special Word Usage

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119.

1.5 Name and Logo Usage

The Near Field Communication Forum’s policy regarding the use of the trademarks *NFC Forum* and the NFC Forum logo is as follows:

- Any company MAY claim compatibility with NFC Forum specifications, whether a member of the NFC Forum or not.
- Permission to use the NFC Forum logos is automatically granted to designated members only as stipulated on the most recent Membership Privileges document, during the period of time for which their membership dues are paid.
- Member’s distributors and sales representatives MAY use the NFC Forum logo in promoting member’s products sold under the name of the member.
- The logo SHALL be printed in black or in color as illustrated on the Logo Page that is available from the NFC Forum at the address above. The aspect ratio of the logo SHALL be maintained, but the size MAY be varied. Nothing MAY be added to or deleted from the logos.
- Since the NFC Forum name is a trademark of the Near Field Communication Forum, the following statement SHALL be included in all published literature and advertising material in which the name or logo appears:

  NFC Forum and the NFC Forum logo are trademarks of the Near Field Communication Forum.

1.6 Intellectual Property

The NFC Data Exchange Format (NDEF) Specification conforms to the Intellectual Property guidelines specified in the NFC Forum's Intellectual Property Right Policy, as approved on November 9, 2004 and outlined in the NFC Forum Rules of Procedures, as approved on December 17, 2004.

1.7 Glossary

NDEF application

The logical, higher-layer application on an NFC Forum Device using NDEF to format information for exchange with other NFC Forum Devices or NFC Forum Tags. Also *user application* or *NDEF user application*.

NDEF message

The basic message construct defined by this specification. An NDEF message contains one or more NDEF records (see section 2.3.1).

NDEF record

An NDEF record contains a payload described by a type, a length, and an optional identifier (see section 2.3.2).

NDEF short record

An NDEF record with the *SR* flag set to 1; the PAYLOAD_LENGTH field in short records is a single octet allowing payloads or chunks of up to 255 bytes to be carried (see section 3.2.4).

NDEF record chunk

An NDEF record that contains a chunk of a payload rather than a full payload (see section 2.3.3). Each record chunk carrying a portion of the chunked payload, except the last record of each chunked payload, has its *CF* flag set to 1.

NDEF payload

The application data carried within an NDEF record.

NDEF chunked payload

Application data that has been partitioned into multiple chunks each carried in a separate NDEF record, where each of these records except the last has the *CF* flag set to 1. This facility can be used to carry dynamically generated content for which the payload size is not known in advance or very large entities that don't fit into a single NDEF record. Chunked payloads are not intended to support multiplexing or streaming of content and such use is deprecated. (See section 2.3.3.)

NDEF payload length

The size of the payload in a single NDEF record indicated as the number of octets (see section 2.4.1).

NDEF payload type

An identifier that indicates the type of the payload. This specification supports URIs [RFC 3986], MIME media type constructs [RFC 2616], as well as an NFC-specific record type as type identifiers (see section 2.4.2).

NDEF payload identifier

An optional URI that can be used to identify a payload (see section 2.4.3).

NDEF generator

An entity or module that encapsulates application-defined payloads within NDEF messages.

NDEF parser

An entity or module that parses NDEF messages and hands off the payloads to an NDEF application.

User Application

See *NDEF Application*.

2 NDEF Mechanisms

This section describes the mechanisms used in NDEF. The specific syntax for these mechanisms is defined in Section 3.

2.1 Introduction

NFC Forum Data Exchange Format is a lightweight binary message format designed to encapsulate one or more application-defined payloads into a single message construct. An NDEF message contains one or more NDEF records, each carrying a payload of arbitrary type and up to 2^{32}-1 octets in size. Records can be chained together to support larger payloads. An NDEF record carries three parameters for describing its payload: the payload length, the payload type, and an optional payload identifier. The purpose of these parameters is as follows:

The payload length

> The payload length indicates the number of octets in the payload (see section 2.4.1). By providing the payload length within the first 8 octets of a record, efficient record boundary detection is possible.

The payload type

> The NDEF payload type identifier indicates the type of the payload. NDEF supports URIs [RFC 3986], MIME media type constructs [RFC 2046], and an NFC-specific type format as type identifiers (see section 2.4.2). By indicating the type of a payload, it is possible to dispatch the payload to the appropriate user application.

The payload identifier

> A payload may be given an optional identifier in the form of an absolute or relative URI (see section 2.4.3). The use of an identifier enables payloads that support URI linking technologies to cross-reference other payloads.

2.2 Intended Usage

The intended usage of NDEF is as follows: A user application wants to encapsulate one or more related documents into a single NDEF message. For example, this can be an application-specific message along with a set of attachments, each of standardized type. The NDEF generator encapsulates each document in NDEF records as payload or chunked payload, indicating the type and length of the payload along with an optional identifier. The NDEF records are then put together to form a single NDEF message. The NDEF message is transmitted across an NFC link to another NFC Forum Device where they are received and parsed, or as an intermediate step, the message is written to an NFC Forum Tag. An NFC Forum Device brought close to this NFC Forum Tag will read the NDEF message from this tag and hand it over to the NDEF parser. The NDEF parser deconstructs the NDEF message and hands the payloads to a (potentially different) user application. Each NDEF message MUST be sent or received in its entirety.

NDEF records can encapsulate documents of any type. It is possible to carry MIME messages in NDEF records by using a media type such as “message/rfc822”. An NDEF message can be encapsulated in an NDEF record by using an NFC-specific predefined type (see [NFC RTD]).

It is important to note that although MIME entities are supported, there are no assumptions in NDEF that a record payload is MIME; NDEF makes no assumption concerning the types of the payloads carried in an NDEF message. Said differently, an NDEF parser need not inspect the NDEF record type nor peer inside an NDEF record in order to parse the NDEF message.

NDEF provides no support for error handling. It is up to the NDEF parser to determine the implications of receiving a malformed NDEF message or an NDEF message containing a field length beyond its processing capabilities. It is the responsibility of the user applications involved to provide any additional functionality such as QoS that they may need as part of the overall system in which they participate.

2.3 NDEF Encapsulation Constructs

2.3.1 Message

An NDEF message is composed of one or more NDEF records. The first record in a message is marked with the *MB* (Message Begin) flag set and the last record in the message is marked with the *ME* (Message End) flag set (see sections 3.2.1 and 3.2.2). The minimum message length is one record which is achieved by setting both the *MB* and the *ME* flag in the same record. Note that at least two record chunks are required in order to encode a chunked payload (see section 2.3.3). The maximum number of NDEF records that can be carried in an NDEF message is unbounded.

NDEF messages MUST NOT overlap; that is, the *MB* and the *ME* flags MUST NOT be used to nest NDEF messages. NDEF messages MAY be nested by carrying a full NDEF message as a payload within an NDEF record.

NDEF Message						
R_1 MB=1	…	R_r	…	R_s	…	R_t ME=1

Figure 1. Example of an NDEF Message with a Set of Records

The message head is to the left and the tail to the right, with the logical record indices t > s > r > 1. The *MB* (Message Begin) flag is set in the first record (index 1) and the *ME* (Message End) flag is set in the last record (index t).

Actual NDEF records do not carry an index number; the ordering is implicitly given by the order in which the records are serialized. For example, if records are repackaged by an intermediate application, then that application is responsible for ensuring that the order of records is preserved.

2.3.2 Record

A record is the unit for carrying a payload within an NDEF message. Each payload is described by its own set of parameters (see section 2.4).

2.3.3 Record Chunks

A record chunk carries a chunk of a payload. Chunked payloads can be used to partition dynamically generated content or very large entities into multiple subsequent record chunks serialized within the same NDEF message.

Chunking is not a mechanism for introducing multiplexing or data streaming into NDEF and it MUST NOT be used for those purposes. It is a mechanism to reduce the need for outbound buffering on the generating side. This is similar to the message chunking mechanism defined in HTTP/1.1 [RFC 2616].

An NDEF message can contain zero or more chunked payloads. Each chunked payload is encoded as an *initial* record chunk followed by zero or more *middle* record chunks and finally by a *terminating* record chunk. Each record chunk is encoded as an NDEF record using the following encoding rules:

- The *initial* record chunk is an NDEF record with the *CF* (Chunk Flag) flag set (see section 3.2.3). The type of the entire chunked payload MUST be indicated in the TYPE field regardless of whether the PAYLOAD_LENGTH field value is zero or not. The ID field MAY be used to carry an identifier of the entire chunked payload. The PAYLOAD_LENGTH field of this initial record indicates the size of the data carried in the PAYLOAD field of the initial record only, not the entire payload size (see section 2.4.1).
- Each *middle* record chunk is an NDEF record with the *CF* flag set indicating that this record chunk contains the next chunk of data of the same type and with the same identifier as the initial record chunk. The value of the TYPE_LENGTH and the *IL* fields MUST be zero and the TNF (Type Name Format) field value MUST be 0x06 *(Unchanged)* (see section 3.2.6). The PAYLOAD_LENGTH field indicates the size of the data carried in the PAYLOAD field of this single middle record only (see section 2.4.1).
- The *terminating* record chunk is an NDEF record with the *CF* flag cleared, indicating that this record chunk contains the last chunk of data of the same type and with the same identifier as the initial record chunk. As with the middle record chunks, the value of the TYPE_LENGTH and the *IL* fields MUST be zero and the TNF (Type Name Format) field value MUST be 0x06 *(Unchanged)* (see section 3.2.6). The PAYLOAD_LENGTH field indicates the size of the data carried in the PAYLOAD field of this terminating record chunk (see section 2.4.1).

A chunked payload MUST be entirely encapsulated within a single NDEF message. That is, a chunked payload MUST NOT span multiple NDEF messages. As a consequence, neither an initial nor a middle record chunk can have the *ME* (Message End) flag set.

2.4 NDEF Payload Description

Each record contains information about the payload carried within it. This section introduces the mechanisms by which these payloads are described.

2.4.1 Payload Length

Regardless of the relationship of a record to other records, the payload length always indicates the length of the payload encapsulated in *this* record. The length of the payload is indicated in the PAYLOAD_LENGTH field. The PAYLOAD_LENGTH field is one octet for *short records* and four octets for normal records. Short records are indicated by setting the *SR* bit flag to a value of 1 (see section 3.2.4). Zero is a valid payload length.

2.4.2 Payload Type

The payload type of a record indicates the kind of data being carried in the payload of that record. This may be used to guide the processing of the payload at the discretion of the user application. The type of the first record, by convention, SHOULD provide the processing context not only for the first record but for the whole NDEF message. Additional context for processing the message MAY be provided, for example, by the link layer service access point (LSAP) or transport service port (e.g. TCP, UDP, etc) at which the message was received and by other communication parameters.

It is important to emphasize that NDEF mandates no specific processing model for NDEF messages. The usage of the payload types is entirely at the discretion of the user application. The comments regarding usage above should be taken as guidelines for building processing conventions, including mappings of higher level application semantics onto NDEF.

The format of the TYPE field value is indicated using the *TNF* (Type Name Format) field (see section 3.2.6). This specification supports TYPE field values in the form of NFC Forum well-known types, NFC Forum external types, absolute URIs [RFC 3986], and MIME media-type constructs. The first allows for NFC Forum specified payload types supporting NFC Forum reference applications [NFC RTD]; URIs provide for decentralized control of the value space; media types allow NDEF to take advantage of the media type value space maintained by IANA [RFC 1700].

The media type registration process is outlined in RFC 2048 [RFC 2048]. Use of non-registered media types is discouraged. The URI scheme registration process is described in RFC 2717 [RFC 2717]. It is RECOMMENDED that only well-known URI schemes registered by IANA be used (see [URI SCHEME] for a current list).

URIs can be used for message types that are defined by URIs. Records that carry a payload with an XML-based message type MAY use the XML namespace identifier of the root element as the TYPE field value. A SOAP/1.1 message, for example, may be represented by the URI

    http://schemas.xmlsoap.org/soap/envelope/

NOTE: Encoding of URI characters which fall outside the US-ASCII range is left to the NDEF application. Therefore, an NDEF parser must not assume any particular encoding for this field. See [RFC 3986] and the specifications of particular protocol schemes (e.g. HTTP, URN, etc.) for more information on parsing of URIs and character encoding requirements for non-ASCII characters.

Records that carry a payload with an existing, registered media type SHOULD carry a TYPE field value of that media type. The TYPE field indicates the type of the payload; it does NOT refer to a MIME message that contains an entity of the given type. For example, the media type

    image/jpeg

indicates that the payload is an image in JPEG format using JFIF encoding as defined by RFC 2046 [RFC 2046]. Similarly, the media type

    message/http

indicates that the payload is an HTTP message as defined by RFC 2616 [RFC 2616]. The value

    application/xml; charset="utf-16"

indicates that the payload is an XML document as defined by RFC 3023 [RFC3023].

2.4.3 Payload Identification

The optional payload identifier allows user applications to identify the payload carried within an NDEF record. By providing a payload identifier, it becomes possible for other payloads supporting URI-based linking technologies to refer to that payload. NDEF does not mandate any particular linking mechanism or format but leaves this to the user application to define in the language it prefers.

It is important that payload identifiers are maintained so that references to those payloads are not broken. If records are repackaged, for example, by an intermediate application, then that application is responsible for ensuring that the linked relationship between identified payloads is preserved.

2.5 NDEF Mechanisms Test Requirements

This section identifies the testable requirements of the NDEF mechanisms defined in chapter 2. The purpose of this section and the table below is to guide the development of conformance tests and does not supersede the normative requirements presented in the other sections of this chapter.

Test Requirements 1. NDEF Mechanisms Test Requirements

Message requirements
Each NDEF message MUST be exchanged in its entirety.
The first record in a message is marked with the *MB* (Message Begin) flag set.
The last record in the message is marked with the *ME* (Message End) flag set.
NDEF messages MUST NOT overlap; that is, the *MB* and the *ME* flags MUST NOT be used to nest NDEF messages.
Record chunk requirements
Each chunked payload is encoded as an initial record chunk followed by 0 or more middle record chunks and finally by a terminating record chunk.
The initial record chunk is an NDEF record with the *CF* (Chunk Flag) flag set.
The type of the entire chunked payload MUST be indicated in the TYPE field of the initial record chunk.
The PAYLOAD_LENGTH field of the initial record indicates the size of the data carried in the PAYLOAD field of the initial record only, not the entire payload size.
Each middle record chunk is an NDEF record with the *CF* flag set.
For each middle record chunk the value of the TYPE_LENGTH and the *IL* fields MUST be 0.
For each middle record chunk the TNF (Type Name Format) field value MUST be 0x06 (*Unchanged*).
For each middle record chunk, the PAYLOAD_LENGTH field indicates the size of the data carried in the PAYLOAD field of this single record only.
The terminating record chunk is an NDEF record with the *CF* flag cleared.
For the terminating record chunk, the value of the TYPE_LENGTH and the *IL* fields MUST be 0.
For the terminating record chunk, the TNF (Type Name Format) field value MUST be 0x06 (*Unchanged*).
For the terminating record chunk, the PAYLOAD_LENGTH field indicates the size of the data carried in the PAYLOAD field of this record only.
A chunked payload MUST be entirely encapsulated within a single NDEF message.
An initial record chunk MUST NOT have the *ME* (Message End) flag set.
A middle record chunk MUST NOT have the *ME* (Message End) flag set.

NDEF payload requirements
The PAYLOAD_LENGTH field is four octets for normal records.
The PAYLOAD_LENGTH field is one octet for records with an *SR* (Short Record) bit flag value of 1.
The PAYLOAD_LENGTH field of a short record MUST have a value between 0 and 255.
The PAYLOAD_LENGTH field of a normal record MUST have a value between 0 and 2^{32}-1.

3 The NDEF Specification

3.1 Data Transmission Order

The order of transmission of the NDEF record described in this document is resolved to the octet level. For diagrams showing a group of octets, the order of transmission of those octets is first left to right and then top to bottom, as they are read in English. For example, in the diagram in Figure 2, the octets are transmitted in the order they are numbered.

```
+--+--+--+--+--+--+--+--+
|        Octet 1        |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|        Octet 2        |        Octet 3        |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|        Octet 4        |
+--+--+--+--+--+--+--+--+
|        Octet 5        |
+--+--+--+--+--+--+--+--+
```

Figure 2. NDEF Octet Ordering

Whenever an octet represents a numeric quantity, the leftmost bit in the diagram is the high order or most significant bit. For each multi-octet field representing a numeric quantity defined by NDEF, the leftmost bit of the whole field is the most significant bit. Such quantities are transmitted in a big-endian manner with the most significant octet transmitted first.

3.2 Record Layout

NDEF records are variable length records with a common format illustrated in the figure below. In the following sections, the individual record fields are described in more detail.

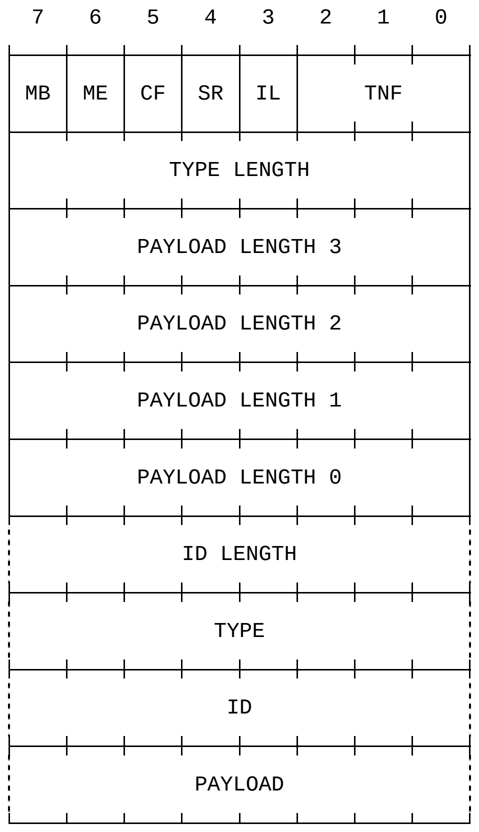

Figure 3. NDEF Record Layout

3.2.1 MB (Message Begin)

The *MB* flag is a 1-bit field that when set indicates the start of an NDEF message (see section 2.3.1).

3.2.2 ME (Message End)

The *ME* flag is a 1-bit field that when set indicates the end of an NDEF message (see section 2.3.1). Note, that in case of a chunked payload, the *ME* flag is set only in the terminating record chunk of that chunked payload (see section 2.3.3).

3.2.3 CF (Chunk Flag)

The *CF* flag is a 1-bit field indicating that this is either the first record chunk or a middle record chunk of a chunked payload (see section 2.3.3 for a description of how to encode a chunked payload).

3.2.4 SR (Short Record)

The *SR* flag is a 1-bit field indicating, if set, that the PAYLOAD_LENGTH field is a single octet. This *short record* layout is intended for compact encapsulation of small payloads which will fit within PAYLOAD fields of size ranging between 0 to 255 octets.

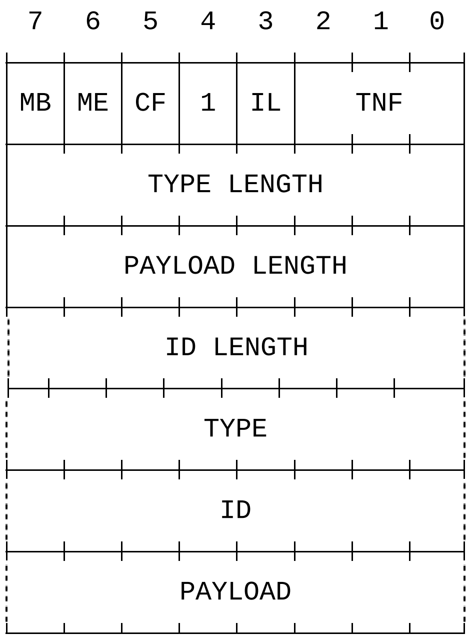

Figure 4. NDEF Short-Record Layout (SR=1)

While it is tempting for implementers to choose one or the other record layout exclusively for a given application, NDEF parsers MUST accept both normal and *short record* layouts. NDEF generators MAY generate either record layout as they deem appropriate. A single NDEF message MAY contain both normal and short records.

3.2.5 IL (ID_LENGTH field is present)

The *IL* flag is a 1-bit field indicating, if set, that the ID_LENGTH field is present in the header as a single octet. If the *IL* flag is zero, the ID_LENGTH field is omitted from the record header and the ID field is also omitted from the record.

3.2.6 TNF (Type Name Format)

The *TNF* field value indicates the structure of the value of the TYPE field (see section 2.4.2 for a description of the TYPE field and section 4 for a description of internationalization issues related to the TYPE field). The TNF field is a 3-bit field with values defined in the table below:

Table 1. TNF Field Values

Type Name Format	Value
Empty	0x00
NFC Forum well-known type [NFC RTD]	0x01
Media-type as defined in RFC 2046 [RFC 2046]	0x02
Absolute URI as defined in RFC 3986 [RFC 3986]	0x03
NFC Forum external type [NFC RTD]	0x04
Unknown	0x05
Unchanged (see section 2.3.3)	0x06
Reserved	0x07

The value 0x00 (*Empty*) indicates that there is no type or payload associated with this record. When used, the TYPE_LENGTH, ID_LENGTH, and PAYLOAD_LENGTH fields MUST be zero and the TYPE, ID, and PAYLOAD fields are thus omitted from the record. This TNF value can be used whenever an empty record is needed; for example, to terminate an NDEF message in cases where there is no payload defined by the user application.

The value 0x01 (*NFC Forum well-known type*) indicates that the TYPE field contains a value that follows the *RTD* type name format defined in the NFC Forum RTD specification [NFC RTD].

The value 0x02 (*media-type*) indicates that the TYPE field contains a value that follows the *media-type* BNF construct defined by RFC 2046 [RFC 2046].

The value 0x03 (*absolute-URI*) indicates that the TYPE field contains a value that follows the *absolute-URI* BNF construct defined by RFC 3986 [RFC 3986].

The value 0x04 (*NFC Forum external type*) indicates that the TYPE field contains a value that follows the type name format defined in [NFC RTD] for external type names.

The value 0x05 (*Unknown*) SHOULD be used to indicate that the type of the payload is unknown. This is similar to the “application/octet-stream” media type defined by MIME [RFC 2046]. When used, the TYPE_LENGTH field MUST be zero and thus the TYPE field is omitted from the NDEF record. Regarding implementation, it is RECOMMENDED that an NDEF parser receiving an NDEF record of this type, without further context to its use, provides a mechanism for storing but not processing the payload (see section 4.2).

The value 0x06 (*Unchanged*) MUST be used in all middle record chunks and the terminating record chunk used in chunked payloads (see section 2.3.3). It MUST NOT be used in any other record. When used, the TYPE_LENGTH field MUST be zero and thus the TYPE field is omitted from the NDEF record.

There is no default value for the TNF field. Reserved (or unassigned) TNF field values are for future use and MUST NOT be used. An NDEF parser that receives an NDEF record with an unknown or unsupported TNF field value SHOULD treat it as 0x05 (*Unknown*).

3.2.7 TYPE_LENGTH

The *TYPE_LENGTH* field is an unsigned 8-bit integer that specifies the length in octets of the TYPE field. The TYPE_LENGTH field is always zero for certain values of the TNF field (see section 3.2.6).

3.2.8 ID_LENGTH

The *ID_LENGTH* field is an unsigned 8-bit integer that specifies the length in octets of the ID field. This field is present only if the *IL* flag is set to 1 in the record header. An ID_LENGTH of zero octets is allowed and, in such cases, the ID field is omitted from the NDEF record.

3.2.9 PAYLOAD_LENGTH

The *PAYLOAD_LENGTH* field is an unsigned integer that specifies the length in octets of the PAYLOAD field (the application payload). The size of the PAYLOAD_LENGTH field is determined by the value of the *SR* flag (see section 3.2.4).

If the *SR* flag is set, the PAYLOAD_LENGTH field is a single octet representing an 8-bit unsigned integer.

If the *SR* flag is clear, the PAYLOAD_LENGTH field is four octets representing a 32-bit unsigned integer. Transmission order of the octets is MSB-first (see section 3.1).

A payload length of 0 is allowed in which case the PAYLOAD field is omitted from the NDEF record. Application payloads larger than 2^{32}-1 octets can be accommodated by using chunked payloads (see section 2.3.3).

3.2.10 TYPE

The value of the *TYPE* field is an identifier describing the type of the payload (see section 2.4.2). The value of the TYPE field MUST follow the structure, encoding, and format implied by the value of the TNF field (see section 3.2.6).

An NDEF parser receiving an NDEF record with a TNF field value that it supports but an unknown TYPE field value SHOULD interpret the type identifier of that record as if the TNF field value were 0x05 (*Unknown*).

It is STRONGLY RECOMMENDED that the type identifier be globally unique and maintained with stable and well-defined semantics over time.

3.2.11 ID

The value of the ID field is an identifier in the form of a URI reference [RFC 3986] (see sections 2.4.3 and 4.4). The required uniqueness of the message identifier is guaranteed by the generator. The URI reference can be either relative or absolute; NDEF does not define a base URI which means that user applications using relative URIs MUST provide an actual or a virtual base URI (see [RFC 3986]).

Middle and *terminating* record chunks (that is, records containing other than the *initial* chunk of a chunked payload; see section 2.3.3) MUST NOT have an ID field. All other records MAY have an ID field.

3.2.12 PAYLOAD

The PAYLOAD field carries the payload intended for the NDEF user application. Any internal structure of the data carried within the PAYLOAD field is opaque to NDEF.

3.3 THE NDEF Specification Test Requirements

This section identifies the testable requirements of the NDEF mechanisms defined in chapter 3. The purpose of this section is to guide the development of conformance tests and does not supersede the normative requirements presented in the other sections of this chapter.

Test Requirements 2. The NDEF Specification Test Requirements

Data transmission order requirements
Quantities are transmitted in a big-endian manner with the most significant octet transmitted first.
Record layout requirements
NDEF parsers MUST accept both normal and short record layouts.
NDEF parsers MUST accept single NDEF messages composed of both normal and short records.
If the IL flag is 1, the ID_LENGTH field MUST be present.
If the IL flag is 0, the ID_LENGTH field MUST NOT be present.
If the IL flag is 0, the ID field MUST NOT be present.
The TNF field MUST have a value between 0x00 and 0x06.
If the TNF value is 0x00, the TYPE_LENGTH, ID_LENGTH, and PAYLOAD_LENGTH fields MUST be zero and the TYPE, ID, and PAYLOAD fields MUST be omitted from the record.
If the TNF value is 0x05 (Unknown), the TYPE_LENGTH field MUST be 0 and the TYPE field MUST be omitted from the NDEF record.
If the TNF value is 0x06 (Unchanged), the TYPE_LENGTH field MUST be 0 and the TYPE field MUST be omitted from the NDEF record.
The TNF value MUST NOT be 0x07.
If the ID_LENGTH field has a value 0, the ID field MUST NOT be present.
If the SR flag is 0, the PAYLOAD_LENGTH field is four octets, representing a 32-bit unsigned integer, and the transmission order of the octets is MSB-first.
If the SR flag is 1, the PAYLOAD_LENGTH field is a single octet representing an 8-bit unsigned integer.
If the PAYLOAD_LENGTH field value is 0, the PAYLOAD field MUST NOT be present.
The value of the TYPE field MUST follow the structure, encoding, and format implied by the value of the TNF field.
Middle and terminating record chunks MUST NOT have an ID field.

4 Special Considerations

4.1 Internationalization

Identifiers used in NDEF such as URIs and MIME media-type constructs may provide different levels of support for internationalization. Implementers are referred to RFC 2718 [RFC 2718] for internationalization considerations of URIs, RFC 2046 [RFC 2046] for internationalization considerations of MIME media types and RFC 2047 [RFC 2047] for internationalization of message headers (MIME).

4.2 Security

Implementers should pay special attention to the security implications of any record types that can cause the remote execution of any actions in the recipient's environment. Before accepting records of any type, an application should be aware of the particular security implications associated with that type.

Security considerations for media types in general are discussed in RFC 2048 [RFC 2048] and in the context of the "application" media types in RFC 2046 [RFC 2046].

4.3 Maximum Field Sizes

The size of the PAYLOAD field and the values used in the ID field and the TYPE field are limited by the maximum size of these fields. The maximum size of the PAYLOAD field is 2^{32}-1 octets in the normal NDEF record layout and 255 octets in the *short record* layout (see section 3.2.4). The maximum size of values in the ID and TYPE fields is 255 octets in both record layouts.

While messages formed to these maximal record and field sizes are considered well-formed, not all user applications will have the ability or the need to handle payload content, payload IDs, or types identifiers of these maximal sizes. NDEF parsers that are resource-constrained MAY choose to reject messages that are not sized to fit their specific needs.

However, NDEF parsers MUST NOT reject an NDEF message based solely on the value of the *SR* flag.

4.4 Use of URIs in NDEF

NDEF uses URIs [RFC 3986] for some identifiers. To NDEF, a URI is simply a formatted string that identifies—via name, location, or any other characteristic—a resource.

The use of IP addresses in URIs SHOULD be avoided whenever possible (see RFC 1900 [RFC 1900]). However, when used, the literal format for IPv6 addresses in URIs as described by RFC 2732 [RFC 2732] SHOULD be supported.

NDEF does not define any equivalence rules for URIs in general as these are defined by the individual URI schemes and by RFC 3986 [RFC 3986]. However, because of inconsistencies with respect to some URI equivalence rules in many current URI parsers, it is RECOMMENDED that generators of NDEF messages rely only on the most rudimentary equivalence rules defined by RFC 3986.

4.5 Special Consideration Test Requirements

This section identifies the testable requirements of the NDEF mechanisms defined in chapter 4. The purpose of this section and the table below is to guide the development of conformance tests and does not supersede the normative requirements presented in the other sections of this chapter.

Test Requirements 3. Special Consideration Test Requirements

An NDEF parser MUST NOT reject an NDEF message based solely on the value of the SR flag.
An NDEF parser MAY reject messages that include records with TYPE, ID, or PAYLOAD fields larger than its design limits.

A. Revision History

The following table outlines the revision history of the NDEF Technical Specification.

Table 2. Revision History

Document Name	**Revision and Release Date**	**Status**	**Change notice**	**Supersedes**
NFCForum-TS-NDEF_1.0	1.0, July 2006	Final	none	