NFC Record Type Definition (RTD)

Technical Specification

NFC Forum^{TM}

RTD 1.0

NFCForum-TS-RTD_1.0

2006-07-24

Contents

Figures

Tables

Test Requirements

1 Introduction

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 data format to exchange information between an NFC Forum Device and another NFC Forum Device or an NFC Forum Tag. The information that can be exchanged by means of NDEF may describe which services an NFC Forum Device or NFC Forum Tag offers, it may contain application or service-specific parameters and meta-data, or it may describe capabilities of NFC Forum Devices or NFC Forum Tags.

NDEF supports the use of standardized MIME content types and URIs to describe record content which is specified outside of the NFC Forum. This specification describes two NFC Forum specific types, known as "NFC Forum Well Known Types" and "NFC external types".

1.1 Objectives

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

The NFC Data Exchange Format specification defines the NDEF data structure format as well as rules to construct a valid NDEF packet as a collection of NDEF records. Furthermore, it defines the mechanism for constructing unique NDEF record type names by different parties, including a format for NFC Forum well-known types.

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. Specific record types are defined in separate documents.

The first part of this specification considers the type format of the NFC Forum well-known types—that is, the contents of an NDEF Type field when the "TNF" header field value is 0x01 (NFC Well-known Type).

The second part of this specification considers the third party extension type known as an "NFC external type", which is signified by the TNF header field value of 0x04.

1.2 Purpose

1.2.1 Mission Statement and Goals

It is the mission of the NFC Forum to ensure interoperability of the NFC technology in a broad variety of devices. The RTD specification is intended to support NFC-specific application and service frameworks by providing a means for reservation of well-known record types, and third party extension types.

The RTD specification provides guidelines for the specification of well-known types for inclusion in NDEF messages exchanged between NFC Forum devices and between NFC Forum Devices and NFC Forum Tags.

Actual type registrations are not provided in this specification but are expected to be included in other documents.

1.3 References

[ASCII]	ANSI X3.4-1986, Coded Character Set 7-bit American Standard Code for Information Interchange
[ISO/IEC 18092]	Information Technology- Telecommunications and information exchange between systems- Near Field Communication - Interface and Protocol (NFCIP-1).
[NDEF]	NFC Data Exchange Format, NFC Forum, 2006.
[NFC Best]	Best Practices for NFC Forum Terminology, NFC Forum, Technical Committee Document.
[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 2046]	N. Freed, N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types" RFC 2046, Innosoft, First Virtual, November 1996.
[RFC 2141]	R. Moats, "URN SYNTAX", May 1997.
[RFC 2234]	D. Crocker, P. Overell, "Augmented BNF for Syntax Specifications: ABNF", November 1997.
[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

1.4 Administration

The NFC Record Type Definition (RTD) 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 Reference Applications Framework technical working group maintains this specification.

This specification has been contributed to by Microsoft, Nokia, Panasonic, Philips, and Sony.

1.5 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.

1.6 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.

1.7 Intellectual Property

The NFC Record Type Definition (RTD) 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.

1.8 Glossary

This section defines all relevant terms and acronyms used in this specification.

Table 1. Definitions

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.

NDEF record

> An NDEF record contains a payload described by a type, a length, and an optional identifier.

NDEF payload

The application data carried within an NDEF record.

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*.

Table 2. Acronyms

NDEF	NFC Data Exchange Format. See *NFC DATA EXCHANGE FORMAT (NDEF*, Re-Draft Revision 0.96*)*, NFC Forum Draft, October 2005
NID	Namespace Identifier. Identifies uniquely an URN namespace. Please see [RFC 2141] for a full definition.
NSS	Namespace Specific String. The rest of the URN after the NID. See [RFC 2141] for a full definition.
MIME	Multipurpose Internet Mail Extensions. A standard specifying the format of strongly-typed data transferred over the Internet. Defined in [RFC 2045-2049]
RTD	Record Type Definition. An NFC-specific record type and type name which may be carried in an NDEF record with a TNF field value of 0x01 (NFC Well-Known Type).
URI	Uniform Resource Identifier. A compact sequence of characters that identifies an abstract or physical resource. [RFC 3986] Uniform Resource Names (URNs) and Uniform Resource Locators (URLs) are both forms of URI.
URN	Uniform Resource Name. A particular type of URI that is defined in [RFC 2141].

2 Record Types

The record type string field of an NDEF record contains the name of the record type (called "record type name"). Record type names are used by NDEF applications to identify the semantics and structure of the record content.

Record type names may be specified in several formats, called Type Name Formats, as signified by the TNF field of the NDEF record header. Record type names may be MIME media types, absolute URIs, NFC Forum external type names, or may be well-known NFC type names (RTD's, the subject of this specification).

Each record type definition is identified by its record type name.

Record type names can be defined by the NFC Forum and by third parties. In the following sections, the rules governing the RTD type name space are defined.

2.1 NFC Forum Well-known Type

The NFC Forum Well-known Type is a dense format designed for tags and creating primitives for certain common types. It is meant to be used in case there is no equivalent URI or MIME type available, or when message size limitations require a very short name.

An NFC Forum Well Known Type is identified inside an NDEF message by setting the TNF field of a record to the value of 0x01, as defined in the NDEF specification.

An NFC Forum Well-Known Type is a URN as defined by [RFC 2141], with the namespace identifier (NID) "nfc".

The Namespace Specific String of the NFC Well Known Type URN is prefixed with "wkt:". However, when encoded in an NDEF message, the Well Known Type MUST be written as a relative-URI construct [RFC 3986], omitting the NID and the "wkt:" –prefix.

For example, the Well Known Type "urn:nfc:wkt:a" would be encoded as "a". The Well Known Type "urn:nfc:wkt:Very-complicated-type" would be encoded as "Very-complicated-type".

There are two kinds of NFC Forum Well Known Types detailed in the sections below. For brevity, we exclude the URN NID and the NSS prefix from the examples.

For a definition of the character ranges used in the Well Known Types, please see chapter 3.

2.1.1 NFC Forum Global Type

The NFC Forum is responsible for defining and managing NFC Forum Global Types. Other parties MUST NOT define or redefine these.

An NFC Forum Global Type SHALL start with an upper-case letter (character range <upper>).

Examples of NFC Forum Global Types: "U", "Cfq", "Trip-to-Texas".

2.1.2 NFC Forum Local Type

NFC Forum Local Types SHALL start with a character in sets <lower> or <number>.

NFC Forum Local Types are available for use within the context of another record. A processing application MUST NOT process these types when application context is not available. Local types are used whenever the burden of using a long, domain name–based external type is too much, and there is no need to define its meaning outside of the local context.

An RTD or an application defines the context for the interpretation for a Local Type. A Local Type MAY be reused by another application in a different context and with different content.

Examples of NFC Forum Local Types: "0", "foo", "u".

2.2 NFC Forum External Type

The External Type Name is meant for organizations that wish to self-allocate a name space to be used for their own purposes.

An External Type is identified in an NDEF record by setting the TNF field value to 0x04, as defined in the NDEF specification [NDEF].

The External Type is, much like a Well Known Type, an URN, with the NID of "nfc". However, the NSS specific part is put into another namespace named "ext". A canonical version of the External Type Name would look like:

"urn:nfc:ext:example.com:f"

The External Type Name MUST be formed by taking the domain name of the issuing organization, adding a colon, and then adding the type name as managed by the organization.

As with Well Known Types, the binary encoding of External Type Name inside NDEF messages MUST omit the NID and the NSS prefix of "ext".

2.3 Record Types Generic Requirements

Test Requirements 1. Record Types Generic Requirements

NFC Forum standardized types defined as RTD records SHALL use NFC Forum Well-Known type names.
When packaged into NDEF records, NFC Forum standardized types defined as RTD records SHALL be signified in the NDEF record header by the Type Name Format (TNF) field value of 0x01 (NFC Forum Well-Known Type).
An NFC Forum Well Known Type SHALL be a URN with the "urn:nfc:wkt:" prefix.
An NFC Forum Global Type MUST NOT be defined or redefined by other parties than NFC Forum.
An NFC Forum Global Type SHALL start with a character in the range <upper> as defined in Chapter 3.
An NFC Forum Local Type SHALL start with a character in the range <lower> or <number> as defined in Chapter 3.
A processing application MUST NOT process a NFC Forum Local Type if an application context is not available.
An NFC Forum Local Type MAY be reused by another application in a different context and with different content.
An NFC Forum External Type SHALL be identified with the TNF field value of 0x04.
An NFC Forum External Type SHALL be a URN with the prefix of "urn:nfc:ext:".
In the NDEF binary format, the URN prefix MUST NOT be used.
The External Type MUST be formed by taking the domain name of the issuing organization, adding a colon, and then adding a type name. An External Type MUST include a colon and a non-zero length type name.

3 RTD Type Names

This section defines the normative requirements for the NFC Forum Well-Known Type Names (below: RTD-URI). The language used is the ABNF format as defined in RFC 2234 [RFC 2234].

```
RTD-URI             = "urn:nfc:" nfc-nss

nfc-nss             = wkt-nss / external-nss

wkt-nss             = wkt-id ":" WKT-type

external-nss        = external-id ":" external-type

wkt-id              = "wkt"

external-id         = "ext"

WKT-type            = local / global

local               = ( lower / number ) *WKT-char

global              = upper *WKT-char

external-type       = dns-part ":" name-part

dns-part            = 1*DNS-char

name-part           = 1*WKT-char

WKT-char            = upper / lower / number / other

DNS-char            = upper / lower / number / "." / "-"


upper               = "A" / "B" / "C" / "D" / "E" / "F" / "G" / "H" /
                      "I" / "J" / "K" / "L" / "M" / "N" / "O" / "P" /
                      "Q" / "R" / "S" / "T" / "U" / "V" / "W" / "X" /
                      "Y" / "Z"

lower               = "a" / "b" / "c" / "d" / "e" / "f" / "g" / "h" /
                      "i" / "j" / "k" / "l" / "m" / "n" / "o" / "p" /
                      "q" / "r" / "s" / "t" / "u" / "v" / "w" / "x" /
                      "y" / "z"

number              = "0" / "1" / "2" / "3" / "4" / "5" / "6" / "7" /
                      "8" / "9"

other               = "(" / ")" / "+" / "," / "-" /
                      ":" / "=" / "@" / ";" / "$" /
                      "_" / "!" / "*" / "'" / "."

reserved            = "%" / "/" / "?" / "#"
```

3.1 Binary Encoding

The binary encoding of Well Known Types and External Type Names for NDEF MUST be done according to the ASCII chart in Appendix A.

The URN NID and the NFC NSS prefixes MUST NOT be included in the binary NDEF format. (However, if RTDs are used in other formats, such as XML, the URNs SHOULD be given in the absolute URN format.)

NOTE: *This specification does not define legal characters for any particular record content. Record content is specified in other documents, specific to those record types.*

3.2 Percent Encoding in NFC Forum Types

To help define equivalence rules for NFC Forum Well Known Types, NFC Forum will not issue a Global Type Name using percent-encoding as defined in [RFC 2141]. Any Local Type Name used by third parties MUST NOT use the percent encoding.

External Types SHOULD NOT use the percent encoding. However, an application using such an external type MUST first encode the string in UTF-8 before converting it to the percent encoding.

3.3 Equivalence of Record Type Names

The comparison of record type names is done on a character-by-character basis.

Two Well Known Type names MUST be compared in a case-sensitive manner. Because of the fact that the encoding is fixed to US-ASCII, it also implies that two Well Known Types MUST be considered equivalent if and only if their binary representations are identical.

Example:

```
"Foobar"
"fooBar"
"fOoBaR"
"foobar"
```

The four examples above are all different Well Known Type names.

Two External Type Names MUST be compared in a case-insensitive manner. Example:

```
"example.com:foobar"
"Example.com:foobar"
"Example.COM:Foobar"
"eXaMpLe.CoM:fOoBaR"
```

The four examples above represent all the same External Type Name.

3.4 RTD Type Names Requirements

Test Requirements 2. RTD Type Names Requirements

The binary encoding of Well Known Types (including Global and Local Names) and External Type names MUST be done according to the ASCII chart in Appendix A.

Well Known Types (including Global and Local Names) MUST NOT use the percent-encoding as defined by RFC 2141.

External types SHOULD NOT use the percent encoding as defined by RFC 2141.

Two Well Known Types (including Global and Local Names) MUST be compared on a case-sensitive, character-by-character basis. In other words, two Well Known Types MUST be considered equal if and only if their binary representations are identical.

Two External Types MUST be compared on a case insensitive, character-by-character basis.

---

4 Error Handling

4.1 Illegal characters

A record with a type name containing characters outside of the valid range of characters defined in Chapter 3 MUST be ignored.

4.2 Unknown Record Types

Applications MUST ignore records which have a Well Known Type or an External Type that they do not recognize.

4.3 Error Handling Requirements

Test Requirements 3. Error Handling Requirements

Any character not defined as a valid character in Chapter 3 SHALL be considered an illegal character in a record type name.
Records containing illegal characters in the record type name MUST be ignored.
An application that does not recognize a record type name MUST ignore the entire record.

A. Character Set for Record Types

Record type names SHALL be formed of characters from of the US ASCII [ASCII] character set. Characters in the range [0-31] and 127 decimal, as shown in the following table, SHALL NOT be used in record type names.

Table 3. ASCII Character Chart

Binary	Dec	Hex	Graph.	Binary	Dec	Hex	Graph.	Binary	Dec	Hex	Graph.
0010 0000	32	20	(blank)	0100 0000	64	40	@	0110 0000	96	60	`
0010 0001	33	21	!	0100 0001	65	41	A	0110 0001	97	61	a
0010 0010	34	22	"	0100 0010	66	42	B	0110 0010	98	62	b
0010 0011	35	23	#	0100 0011	67	43	C	0110 0011	99	63	c
0010 0100	36	24	$	0100 0100	68	44	D	0110 0100	100	64	d
0010 0101	37	25	%	0100 0101	69	45	E	0110 0101	101	65	e
0010 0110	38	26	&	0100 0110	70	46	F	0110 0110	102	66	f
0010 0111	39	27	'	0100 0111	71	47	G	0110 0111	103	67	g
0010 1000	40	28	(	0100 1000	72	48	H	0110 1000	104	68	h
0010 1001	41	29	)	0100 1001	73	49	I	0110 1001	105	69	i
0010 1010	42	2A	*	0100 1010	74	4A	J	0110 1010	106	6A	j
0010 1011	43	2B	+	0100 1011	75	4B	K	0110 1011	107	6B	k
0010 1100	44	2C	'	0100 1100	76	4C	L	0110 1100	108	6C	l
0010 1101	45	2D	-	0100 1101	77	4D	M	0110 1101	109	6D	m
0010 1110	46	2E	.	0100 1110	78	4E	N	0110 1110	110	6E	n
0010 1111	47	2F	/	0100 1111	79	4F	O	0110 1111	111	6F	o
0011 0000	48	30	0	0101 0000	80	50	P	0111 0000	112	70	p
0011 0001	49	31	1	0101 0001	81	51	Q	0111 0001	113	71	q
0011 0010	50	32	2	0101 0010	82	52	R	0111 0010	114	72	r
0011 0011	51	33	3	0101 0011	83	53	S	0111 0011	115	73	s
0011 0100	52	34	4	0101 0100	84	54	T	0111 0100	116	74	t
0011 0101	53	35	5	0101 0101	85	55	U	0111 0101	117	75	u
0011 0110	54	36	6	0101 0110	86	56	V	0111 0110	118	76	v
0011 0111	55	37	7	0101 0111	87	57	W	0111 0111	119	77	w
0011 1000	56	38	8	0101 1000	88	58	X	0111 1000	120	78	x
0011 1001	57	39	9	0101 1001	89	59	Y	0111 1001	121	79	y
0011 1010	58	3A	:	0101 1010	90	5A	Z	0111 1010	122	7A	z
0011 1011	59	3B	;	0101 1011	91	5B	[	0111 1011	123	7B	{
0011 1100	60	3C	<	0101 1100	92	5C	\	0111 1100	124	7C		
0011 1101	61	3D	=	0101 1101	93	5D	]	0111 1101	125	7D	}
0011 1110	62	3E	>	0101 1110	94	5E	^	0111 1110	126	7E	~
0011 1111	63	3F	?	0101 1111	95	5F	_				

B. Record Type Name Examples

The contents of this appendix are informative and describe examples for encoding and comparing record type names into their binary representation.

An example of translating a record type name into binary representation:

Table 4. Translating Record Type Names into Binary Representation

String Representation	Binary Representation (as hexadecimal)
Sms	53 6D 73
sms	73 6D 73

To encode the binary representation of the type names, each character from the string representation is replaced by its binary value from Appendix A.

In this example, the two record type names are considered non-equivalent since their binary representations are not identical. The case-sense of letters in the string, white space, and other language comparison rules are not considered when comparing type strings for equivalence. Only the binary representations are considered.

C. Discussion on Associating Records

The contents of this appendix are informative.

There are two basic ways to associate NDEF records to each other. The first one is called "association by reference", which is amounts to a flat hierarchy or a list of objects.

When associating records by reference, the context is typically given by the first record in the message. This is the same association model that is used by MIME. For example, if you wish to represent an email message with two PNG attachments as an NDEF message, you first send the email message in one record (typed message/rfc822), then the first PNG image as a separate record (image/png), and the second PNG image (again, image/png). To illustrate:

NDEF MESSAGE		
Email (message/rfc822)	Pic1.png (image/png)	Pic2.png (image/png)

Figure 1. NDEF Messages (Multiple)

This method allows an application to lift the PNG images off the message, even if it does not understand the email message. In general, when designing your own record types, you should choose association by reference if your message parts would be valuable even on their own, i.e., even if the context is not understood. Association by reference is also a good model if you are moving a large amount of data because it allows you to take advantage of the chunking feature of NDEF. In addition, it also allows the processing to start at the receiver end before the message is finished (this is one of the reasons why it is good to declare the context at the beginning of the message).

The second way is called "association by containment". This is a hierarchical model (not entirely unlike XML or HTML), where the content portion of an NDEF records contains an NDEF message. This is very useful in the case where you wish to imply a stronger relationship between records, or need to serialize information that is already in a hierarchical format. Also, if you are going to send multiple objects of the same type within the message, you probably wish to use an containment model, and then string them together in a list(so yes, it is possible and very sensible to mix these models).

For example, the Smart Poster record defines a URI plus some added metadata about that URI. The added metadata is not useful to an application without the URI itself, and in fact, it would be relatively meaningless. To illustrate:

NDEF Message				
Sp (Smart Poster)				application/vcard
URI	Text	Action	Configuration	vCard data

Figure 2. NDEF Message with Metadata

In this case, there are two records in the NDEF message. The first one is a Smart Poster containing a URI, a Text record for a title, an action, and a configuration record; whereas the other one is just a normal vCard (using the vCard standard). (In this case, there is no particular context defined for the vCard, so an application may either ignore it or use it for some purpose; this is an implementation detail. In general, putting records describing different things and assuming some particular context or processing model will probably result in interoperability trouble.)

Anyway, since the Text, Action, and Configuration are so tightly coupled with the URI (the URI might not even be fetchable without the proper configuration, if the config defines a local access point), they work better using a containment model than a reference model.

Neither of these examples displayed any use of the ID field, which can be used in both models with equal efficiency. In association by reference, the first record typically lists the IDs that it uses and defines the context that way; in association by containment, the IDs would typically be used to signify the role of a record (e.g., "A record with an ID of 'config' shall be used for defining an access point.")

Of course, an application is free to mix-and-match these association types. There is no hard-and-fast rule to say which one is better in a given situation, and as designed, this allows maximum flexibility to the application developer.

A third, but deprecated, practice would be using ordering (i.e., record #1 would always signify something, record #2 something else, record #3 again something else), but this, in general, is not a good idea, since you cannot rely on any particular behavior of a NDEF processor. It could be that by the time your application receives the NDEF message, records may have been inserted or removed. Do not rely on any implementation-specific behavior. This seems obvious to any seasoned developer, but it is easy to forget in the rush of a deadline.

The advice in this discussion is offered because it is likely that developers at some point face the need to associate NDEF records with each other, and it is good that some of the best practices and conventions are laid out for all to see. Reading a new specification can be difficult, and hopefully discussion such as this will ease the work of the developer.

D. Revision History

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

Table 5. Revision History

Document Name	Revision and Release Date	Status	Change notice	Supersedes
NFCForum-TS-RTD_1.0	1.0, July 2006	Final	none	