Master 2 SSIO 2022/2023

TD/TP 1

NFC Data Exchange Format (NDEF)

The objective of this 1st practical working session is to understand the mechanisms used in NFC Data Exchange Format (NDEF) and to be able at the end writing and parsing an NDEF message.

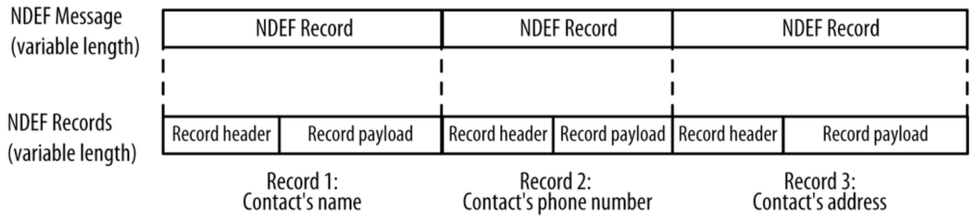
1. Introduction

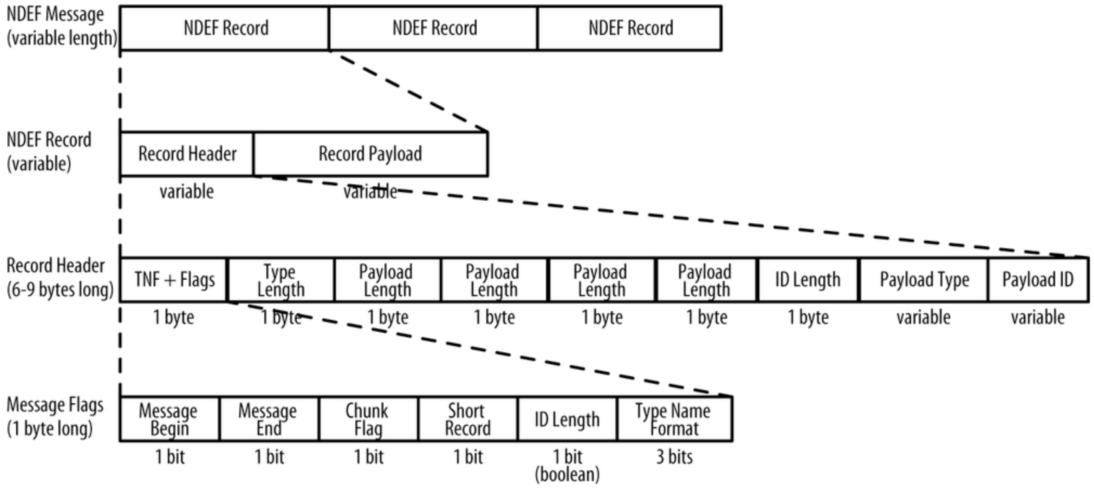
NDEF is a lightweight binary message format designed to encapsulate one or more application-defined payloads into a single message construct. An NDEF message contains one or more NDEF records, each is carrying a payload of arbitrary type and up to 2^{32}-1 octets in size. Records can be chained together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

2. NDEF encapsulation constructs

The first record in a NDEF message is marked with the *MB* (Message Begin) flag set (to 1) and the last record in the message is marked with the *ME* (Message End) flag set. The minimum message length is one record which is achieved by setting both the *MB* and the *ME* flag in the same record. Each record contains a Header Record and a Payload Data Field. The Record Header contains the TNF (Type Name Format) + 5 flags (bits) + the NDEF Payload descriptors. These 5 flags consist on: MB=Message Begin, ME=Message End, Record Chunk Flag, Single Record flag and ID Length flag. The *CF* flag indicates if the current record is either the first record chunk or a middle record chunk of a chunked payload. The *IL* flag indicates, if set, that the ID_LENGTH field (which is a part of NDEF payload descriptors described above) 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. The TNF field indicates as described in the next paragraph the format of the Type field in the record header.

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





2.1. NDEF Payload Descriptors

Payload descriptors consists of Payload Length Fields, Payload Type and Payload Identifier.

A. Payload Length

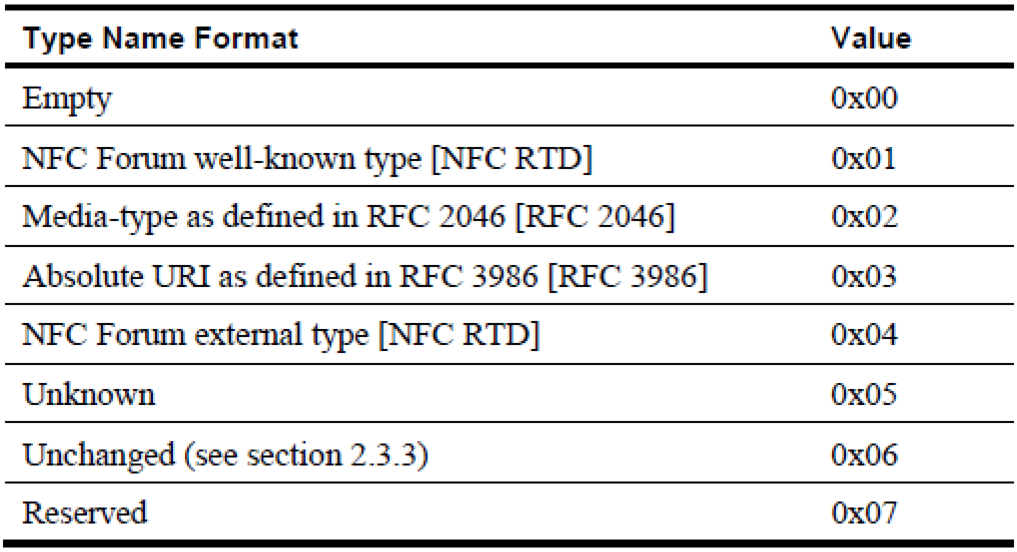
This PAYLOAD_LENGTH field is one octet for short records and four octets for normal records. Short records are indicated by setting the flag SR (Single Record) bit flag to a value of 1.

B. Payload Type

The NDEF payload type identifier indicates the type of data being carried in the payload of that record which can be URIs [Uniform Resource Identifier- RFC 3986], Multipurpose Internet Mail Extensions (MIME) media type constructs [RFC 2046] or a NFC-specific type format as type identifiers. By indicating the type of a payload, it is possible for the NDEF parser to dispatch the payload to the appropriate user application.

The type of the first record provides the processing context not only for the first record but for the whole NDEF message.

The format of the TYPE field value is indicated by the ***TNF* (Type Name Format)** field. The NFC Forum specification supports as shown in the table below four different 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.



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 NFC Well-Known type allows for NFC Forum specified payload types supporting NFC Forum reference applications [NFC RTD : Record Type Definition]; URIs provide for decentralized control of the value space; media types allow NDEF to take advantage of the media type value space maintained RFC 1700.

The TNF field is a 3-bit field. The value 0x00 (Empty) of this field indicates that there is no type or payload associated with this record.

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* construct defined by RFC 2046.

The value 0x03 (*absolute-URI*) indicates that the TYPE field contains a value that follows the *absolute-URI* defined by 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.

C. Payload Identification

The optional Payload Identifier allows user applications to identify the payload carried within an NDEF record.

2.2. 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. NDEF Record Types Technical Specification

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**. Thus, for an example of writing a Smart Poster within a NFC tag, the developer has to refer to the **NFC Smart Poster RTD Technical Specification.** And if this developer wants to write a text (ex. A biography of an author or piece of an art work) he then has to refer to **NFC Text RTD Technical Specification.** You will see that in some examples this developer has to refer à the same time to different NFC Forum technical specifications to write information within a tag (ex. Smart Poster that display at the same time an internet URL with a text information or any other information or content).

Some NDEF writing examples:

A. Sample URI on a Tag

Offset	Content	Length (Byte)	Explanation
0	0xD1	1	0xD1 is the NDEF Header =[1101 0001]: MB = 1, ME = 1, CF = 0, SR = 1, IL = 0 and TNF = 0 x 01 (Well Known Type)
1	0x02	1	Record name length (2 bytes)
2	0x12	1	Length of the Smart Poster data (18 bytes)
3	“Sp”	2	The record name. Its hexadecimal corresponding value “0 x 53 0 x 70” is provided by the ASCII character Chart table in Annex A.
5	0xD1	1	NDEF header. TNF = 0x01, SR=1, MB=1, ME=1
6	0x01	1	Record name length (1 byte)
7	0x0E	1	The length of the URI payload (14 bytes)
8	“U”	1	Record type: “U” = 0x54 (see ASCII table)
9	0x01	1	Code abbreviation of http://www given in the URI RTD Tech. Spec (see Annex B).
10	“nfc-forum.org”	13	The URI itself.

B. Complex URI example

Offset	**Content**	**Length**	**Explanation**
0	0xD1	1	NDEF header. TNF = 0x01 (Well Known Type). SR=1, MB=1, ME=1
1	0x02	1	Record name length (2 bytes)
2	0x49	1	Length of the Smart Poster data (73 bytes)
3	“Sp”	2	The record name
5	0x81	1	NDEF header. TNF = 0x01, SR=0, MB=1, ME=0
6	0x01	1	Record name length (1 byte)
7	0x00, 0x00, 0x00, 0x0E	4	The length of the URI payload (14 bytes) (long format)
11	“U”	1	Record type: “U”
12	0x01	1	Abbreviation: “http://www.”
13	“nfc-forum.org”	13	The URI itself.
26	0x11	1	NDEF record header (SR=1, TNF=0x01)
27	0x03	1	The length of the record name
28	0x01	1	The length of the “act” payload.
29	“act”	3	Record type: “act”

32	0x00	1	Action = Launch browser
33	0x11	1	NDEF record header (SR=1, TNF=0x01)
34	0x01	1	Length of the record name
35	0x12	1	Length of the record payload (18 bytes)
36	“T”	1	Record type: “T” (=Text)
37	0x05	1	Status byte for the Text (UTF-8, five-byte code)
38	“en-US”	5	ISO Language code: US-English
43	“Hello, world”	12	The text: “Hello world”, encoded in UTF-8.
55	0x51	1	NDEF record header (SR=1, TNF= 0x01, ME=1)
56	0x01	1	Record name length
57	0x13	1	Length of the Text payload (19 bytes)
58	“T”	1	The name of the Text record (“T”)
59	0x02	1	Status byte: UTF-8, two-byte language code
60	“fi”	2	ISO two-character language code: Finnish
62	“Morjens, maailma”	16	The text “Morjens, maailma” encoded in UTF-8

Exercise 1:

Based on those two provided examples, parse the following NDEF messages.

# NDEF message 1:

```
D1 01 4B 55 02 63 6F 6E 74 65 64 2E 6F 78 2E 61
63 2E 75 6B 2F 61 62 6F 75 74 2F 61 69 2D 72 6F
62 6F 74 69 63 73 2D 64 61 74 61 2D 73 63 69 65
6E 63 65 2D 69 6F 74 2D 69 6E 66 6F 72 6D 61 74
69 6F 6E 2D 65 6E 67 69 6E 65 65 72 69 6E 67
```

# NDEF message 2:

```
D1 02 76 53 70 91 01 57 55 00 68 74 74 70 73 3A
2F 2F 77 77 77 2E 63 6F 6E 74 65 64 2E 6F 78 2E
61 63 2E 75 6B 2F 61 62 6F 75 74 2F 61 69 2D 72
6F 62 6F 74 69 63 73 2D 64 61 74 61 2D 73 63 69
65 6E 63 65 2D 69 6F 74 2D 69 6E 66 6F 72 6D 61
74 69 6F 6E 2D 65 6E 67 69 6E 65 65 72 69 6E 67
51 01 17 54 02 66 72 45 78 65 72 63 69 73 65 20
31 20 4D 65 73 73 61 67 65 20 32
```

# NDEF message 3:

```
D1 01 52 54 02 66 72 50 72 C3 A9 73 65 6E 74 61
74 69 6F 6E 20 64 75 20 6D 61 73 74 65 72 20 53
79 73 74 C3 A8 6D 65 73 20 65 74 20 73 65 72 76
69 63 65 73 20 70 6F 75 72 20 6C 27 69 6E 74 65
72 6E 65 74 20 64 65 73 20 6F 62 6A 65 74 73 20
28 53 53 49 4F 29
```

Exercise 2
Write a NDEF message of a Smart Poster displaying the following URL “https:// boutique.orange.fr” And titled in French language by “Bienvenue chez Orange, que cherchez-vous ?”.

Exercise 3
Use an NFC capable device capable of writing on NFC tags.

Install the following Android applications “NXP TagInfo”, “NXP TagWritter” and “NFC Tools”.

Write NFC tag with the following Record Types by using either “NXP TagWritter” or “NFC Tools”.

1- Configuring a WiFi connection to a known hotspot. This configuration depends on the nature of the targeted WiFi hotspot (ie : open or secured). It’s up to you to choice which hotspot you want to target.
2- Pairing of two Bluetooth devices.
3- Launch an Emergency call
4- Write SMS

Then use the TagInfo app and provide the obtained NDEF message for each requested example.

Based on the attached NFC Forum Specifications provide explanations about the obtained Record messages for each example.

Exercise 4
Comment in few lines or a short paragraph the main rules or functions of the code provided here in this link.

https://cs.android.com/android/platform/superproject/+/master:packages/apps/Tag/src/com/android/apps/tag/record/UriRecord.java

ANNEX A.

Table 1. 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	_				

Annex B.

Table 2. RTD URI abbreviation Table

Decimal	Hex	Protocol
0	0x00	N/A. No prepending is done, and the URI field contains the unabridged URI.
1	0x01	http://www.
2	0x02	https://www.
3	0x03	http://
4	0x04	https://
5	0x05	tel:
6	0x06	mailto:
7	0x07	ftp://anonymous:anonymous@
8	0x08	ftp://ftp.
9	0x09	ftps://