Comprendre les métadonnées DICOM

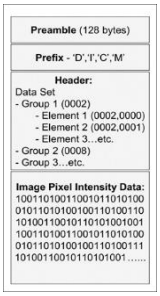
Par Dimitri PIANETA

2016

Sommaire

Chapitre 1 : Comprendre les métadonnées

1. Les images DICOM sont composes de trois parties :
   a. Preamble
   b. Header
   c. Image (or more technically – pixels)



- **Preamble** : Chaque entête commence par un préambule de 128 octets généralement mis à zéro suivi de 4 octets pour y inscrire les caractères 'D', 'I', 'C', 'M'.

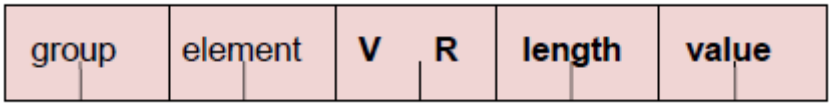
- **DICOM header** : est un entête très complexe selon le type d'image. Dans cette entête les attributs sont appelés étiquette (TAG).

1. Les TAGs sont posées les unes après les autres sans espacement ou le caractère différentiel.
2. Tags sont uniques. Il n'y a pas Tags qui ont les mêmes numéros ou des identificateurs
3. Selon les normes DICOM les TAGs sont innombrables . Donc, les balises sont divisés en familles. Certaines familles sont - 0002 , 0008, 0028,7FE0 .
4. Ces familles ont à nouveau les membres de leur famille respective. Par exemple 0002,0010 est une balise. Où 0002 est le nom de famille ou le numéro de groupe et 0010 est le nom du membre ou le numéro d'élément. Il continue à (0002,0011) , (0002,0012) etc
5. **Tag Structure** se compose de quatre segments. Ces segments commencent par le nom de la balise suivie par la représentation de la valeur suivie par la longueur de la valeur suivie de la valeur
6. La méta-données de DICOM sont faites par Tag (Groupe , élément) :

Group Number + Element Number = Tag Name.

Un nom de tag est constitué de 2 parties. Le numéro de groupe suivi par le numéro d'élément occupant 2 octets chacun. i.e le nom du TAG est totalement occupe 4 octets chacun.

7. Ces numéros d'étiquette sont au format hexadécimal . Lorsque vous les lisez les convertir de binaire en format hexadécimal . En java , il peut être fait en utilisant Integer.toHexString ((int) octets) ;
8. TAG:



- **Value Representation(VR)** est suivi du nom de Tag. Il se compose de 2 ou 3 caractères lettrés décrivant le Tag. Il occupe 2 octets. Quelques exemples de VR sont ' OW ', ' OB ', etc.
Ceci est codé sous forme de caractères. Il suffit de lire le binaire et de faire une chaîne char de lui.
- **Value Length (VL)** est suivi par le VR. Il contient les informations de la taille du champ Valeur qui suit VL. La valeur est enregistrée au format Unsigned Int. Il occupe 2 octets.
Même cela est au format hexadécimal . J'ai converti ce au format décimal pour lire la valeur. Lecture de la valeur peut également être effectuée sans conversion.
- **Value** suivi VL. Ceci est le dernier segment d'une marque. Il se compose d'une valeur qui est nécessaire. La taille qu'il occupe est obtenue à partir VL.
Le format dans lequel la valeur doit être lue est généralement Chaîne de caractères. Sauf pour certaines valeurs qui pourraient contenir des valeurs numériques. Si vous le faites pour la première fois juste enregistrer la valeur comme un octet et passer à la prochaine balise.

10. reading Tags :

- Le 1er tag commence toujours avec le Groupe 0002. Dans chaque fichier, il y a un certain nombre de groupe 0002 balises de la famille pour décrire quel type est le fichier. Ex : si elle est petite type Endian ou Big Endian type, implicite type VR ou explicite de type VR, et également si elle est un fichier CTN. (Ceux-ci seront expliqués plus loin).
- Le dernier Tag est le (FE0, 0010) Tag. Il se compose de pixels. Cette balise ultime. Pour trouver ce tag nous devons lire toutes les balises précédentes dans le fichier image.
- Une exception dans ce cas est si le VR est égal à ' OB ', ' OW ' , 'OF' , ' SQ ', ' UI »ou« ONU », le VR est d'avoir un supplément de 2 octets de fuite à elle. Ces 2 octets de fin de VR sont vides et ne sont pas décodés.
- Lorsque VR est d'avoir ces 2 octets vides supplémentaires VL occupera 4 octets au lieu de 2 octets.
    - La balise 0002,0010 a la syntaxe de transfert qui détermine si le fichier est explicite, implicite, Little endian, ou Big endian.
- Si le transfert de syntaxe est 1.2.840.10008.1.2.1 le fichier est Little Endian. ex : chaque bit suivant, nous lisons est préfixé au réseau (dans lequel nous lisons).
- Il existe différents types de fichiers DICOM. Main 2 sont explicites VR et VR implicite. Il suffit de parler du type VR Explicit dispose d'un VR où le type implicite n'a pas de VR.

9. Explicit VR type:

Structure du TAG se compose de quatre segments. Ces segments commencent par le nom de la balise suivie par la représentation de la valeur suivie par la longueur de la valeur suivie de la valeur.

Group Number + Element Number = Tag Name.

Un nom de tag est constitué de 2 parties. Le numéro de groupe suivi par le numéro d'élément occupant 2 octets chacun. Ex : le nom de balise est totalement occupe 4 octets chacun.

Ces numéros d'étiquette sont au format hexadécimal. Lorsque vous les lisez les convertir de binaire en format hexadécimal. En java, il peut être fait en utilisant Integer.toHexString ((int) octets).

Value Representation(VR) est suivi du nom de Tag.
Il se compose de 2 ou 3 caractères lettrées décrivant le Tag. Il occupe 2 octets. Quelques exemples de VR sont ' OW ', ' OB ', etc.
Ceci est codé sous forme de caractères. Il suffit de lire le binaire et de faire une chaîne char de lui.

Value Length (VL) est suivi par le VR . Il contient les informations de la taille du champ Valeur qui suit VL. La valeur est enregistrée au format Unsigned Int. Il occupe 2 octets.

Même cela est au format hexadécimal. J'ai converti ce au format décimal pour lire la valeur. Lecture de la valeur peut également être effectuée sans conversion.

Value suit le VL. Ceci est le dernier segment d' une marque . Il se compose d'une valeur qui est nécessaire. La taille qu'il occupe est obtenue à partir VL.

Le format dans lequel la valeur doit être lue est généralement Chaîne de caractères. Sauf pour certaines valeurs qui pourraient contenir des valeurs numériques. Si vous le faites pour la première fois juste enregistrer la valeur comme un octet et passer au prochain Tag.
Ainsi, lorsque nous lisons un en-tête DICOM nous devons lire les points 8 thru 11 en continu jusqu'à ce que le dernier TAG

10. Implicit VR type:

Tag structure de implicit VR de type commence par une balise 4 octets et suivi de 4 octets VL suivie par la valeur qui est la longueur est définie dans VL.

- Even in Implicit VR type files, the 0002 group tags or technically – the Metadata, is Explicit. i.e an implicit VR type file starts with Explicit VR only and the metadata decides if the file is Implicit. And if implicit the implicit reading should start after the end of Metadata.
- So recording the size of metadata is important. This info is in the first tag – 0002,0000. Caution – the length of Metadata is the length after the value field of the tag 0002,0000 till the end of metadata.
- If the Transfer syntax is 1.2.840.10008.1.2 it is of Implicit VR Type with Little Endian. Implicit VR is always only Little Endian. No Big Endian in Implicit VR.
- There is absolutely no need of a Data Dictionary (as said by Lead Tools) in the first instance even for implicit VR. No trailing empty bits (as in explicit VR) apply to Implicit VR.
- Another main part of the DICOM files is the Squences.
- The tags which have a VR of SQ are identified as Sequence tags (meaning doesn't apply to Implicit VR type files).
- A ***Sequence Tag*** has more tags encapsulated in its Value field.
- A Sequence Tag can again have a Sequence Tag in its encapsulated tags.
- So nesting is widely possible.
- The sequence tag consists of Items in it.
- Each Item starts with FFFE,E000.
- The ***Item structure*** starts with 2bytes of item no. 4 bytes of item length, and item value (value consists of normal tags as defined in point 8).
- If the Item Length is having a value of "-1" then the value field is known to be having an undefined value length.
- In an undefined Value length, the End of Item is defined by the tag FFFE,E00D.
- The End of Sequence is defined by FFFE,E0DD.
- Both the fffe,e00d and fffe,e0dd have a 2 bytes tag no. and 4 bytes Value length. But do not have a value field.
- Another possible Dicom File is the CTN file. This does not have the preamble. There are no empty bytes in the beginning of the file. There is no DICM either.
- Surprisingly this file doesn't start with a Metadata tags. i.e no 0002 group tags at all.
- It starts with 0008 group tags.
- It is Implicit VR with Little Endian.
- The rest of it is same as Impicit VR.

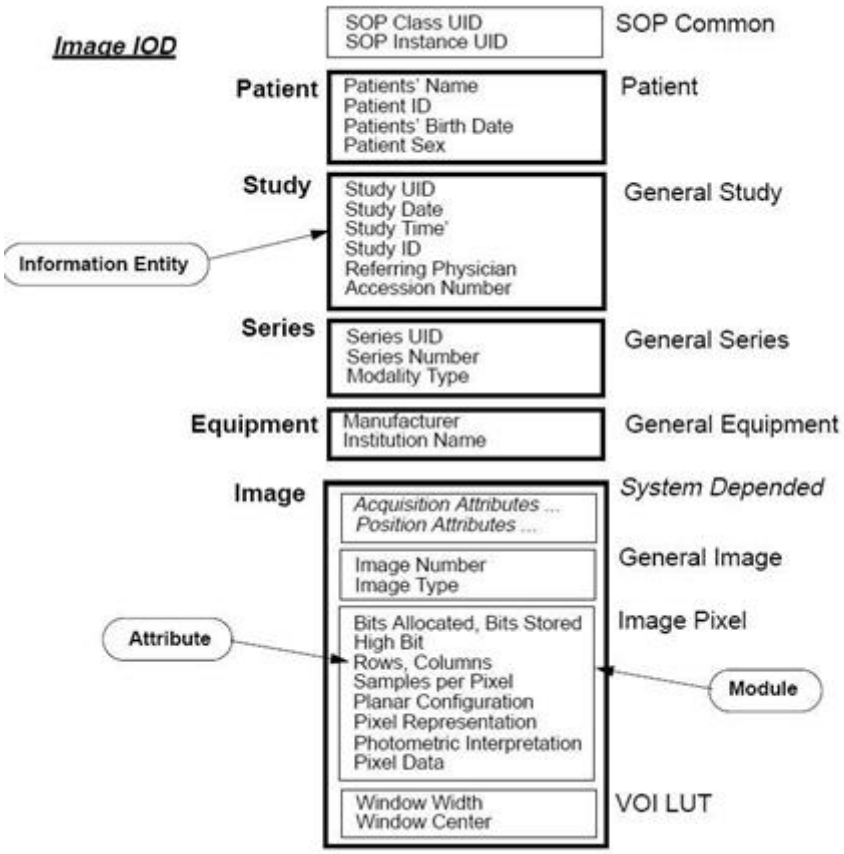
Multiframe images: Un fichier dicom peut contenir nombre de plusieurs cadres. Le nombre de cadres est défini dans la balise 0028,0008.

- Tips in reading pixles:

    a. Suppose pixles[] is the array into which you will read the pixles, its length should be equal to (rows x columns).
    b. For multiframe images after every end of (rows x columns) size the next image starts . This continues until ((rows x columns) x no.of frames).
    c. Another tip is if you are given undefined length for length of pixels tag, just (rows x columns) for 8 – bit image, (rows x columns) x 2 for 16 bit image and so on.

Chapitre 2 : rangement des tags

1. Tous les Tags des métadonnées sont ranges dans cette ordre



2. Table of tag

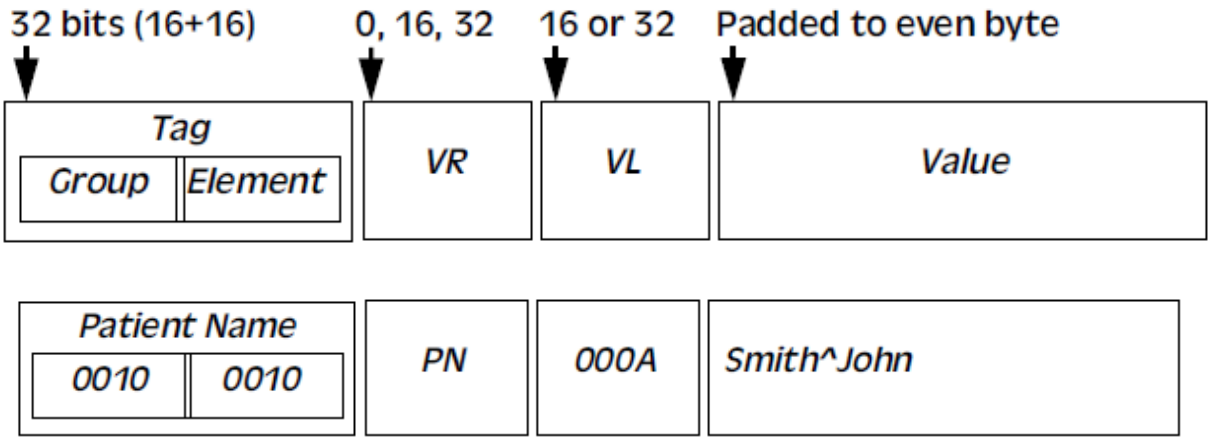
Tag Number	Tag Name
(0000,xxxx)	Command elements and uses for send files dicom
(0002,xxxx)	These tags of meta group et isn't send of network.
(0004,xxxx)	DicomDir
(0008,xxxx)	Identifying tags
(0010,xxxx)	Patient info tags
(0012,xxxx)	Clinical Trial
(0018,xxxx)	Acquisition of owners of tags or part of exam body in the study.
(0020,xxxx)	Position and information relatif at acquisition
(0022,xxxx)	
(0028,xxxx)	Presentation of image (dimension, grayscale, lut)
(0032,xxxx)	Requesting ou Study information
(0038,xxxx)	Scheduled admission Information

(003a,xxxx)	Waveform properties tags
(0040,xxxx)	Protedure tags (texte)
(0042,xxxx)	Document
(0044,xxxx)	Product
(0046,xxxx)	Visual Acuity
(0050,xxxx)	Container Component
(0054,xxxx)	Energy Window
(0060,xxxx)	Histogram
(0062,xxxx)	Segmentation
(0064,xxxx)	Grid, Vector Grid, Source Frame ID
(0066,xxxx)	Surface
(0070,xxxx)	Annotation
(0072,xxxx)	Hanging Protocol
(0074,xxxx)	Unified Procetured, Beam, Contact.
(0088,xxx)	Storage, Topic
(0088,09xx)	Topic Information
(0100,xxxx)	Sop instance Status, SOP autorization
(0400,xxxx)	MAC, Digital, certificate, encrypted
(2000,xxxx)	
(2010,xxxx)	Characteristic of films and print
(2020,xxxx)	Image Box, Polarity, Requested, Basic, refereneced Image
(2030,xxxx)	Annotation position and text string
(2050,xxxx)	Presentation LUT
(2100,xxxx)	Execution, Creation
(2110,xxxx)	Printer
(2200,xxxx)	Label, Barcode, include, requested media
(3002,xxxx)	,RT image, Xray, radiation machine, fluence data
(3004,0001)	DVH
(3006,xxxx)	Structure Set, Contour, ROI, frame of reference
(3008,xxxx)	Measured Dose, Treatment Control, Treatment termination
(300A,xxxx)	RT, dose reference, Table Top
(300C,xxxx)	Rereferenced
(300E,xxxx)	Approval, Review
(5200,xxxx)	Shated Functional Groupe Sequence
(5400,xxxx)	Waveform data tags
(60xx,xxxx)	Overlay, type of compression.
(7FE0,0010)	Pixel Data, Pixels de l'image
(FFFA,FFFA)	Digital Signatures Sequence
(FFFC,FFFC)	Data Set Trailing Padding
(FFFE,xxxx)	Items

Chapitre 3 : lire les métadonnées DICOM

Les métadonnées sont des tags de mots de 16 bits donc deux mots de 2 octets de (16 bits). Ce mot est en hexadécimal.

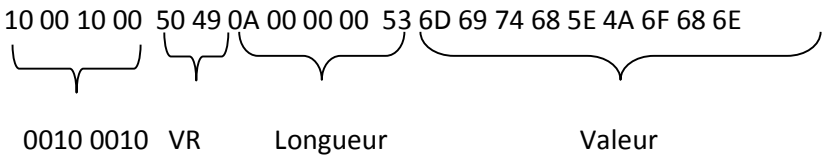
Nous allons voir un exemple du Tag du Nom du Patient:



PN is Person Name, 000A a pour base hexadécimale (16 bits) qui converti en base de 10 qui signifie ici 10 caractères.

VR signifie Valeur et que cette valeur est en format ASCII de base de 16.

Réponse de l'exemple en hexadécimal:



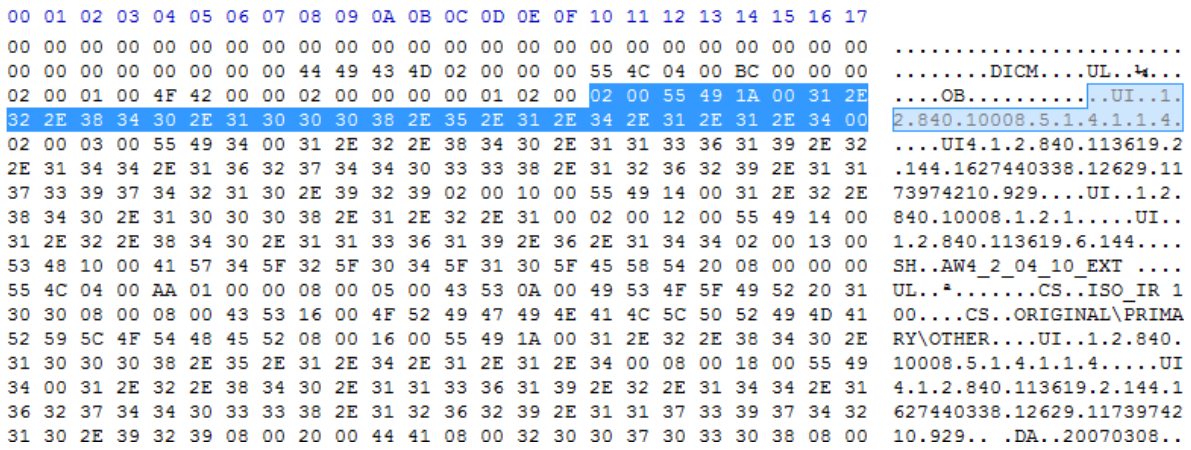
1er exemple :

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000078	00	00	00	00	00	00	00	00	44	49	43	4D	02	00	00	00	55	4C	04	00	BC	00	00	00	.....DICM....UL..¼...
00000090	02	00	01	00	4F	42	00	00	02	00	00	00	00	00	01	02	00	02	00	55	49	1A	00	31	2E	....OB.....UI..1.
000000A8	32	2E	38	34	30	2E	31	30	30	30	38	2E	35	2E	31	2E	34	2E	31	2E	31	2E	34	00		2.840.10008.5.1.4.1.1.4.
000000C0	02	00	03	00	55	49	34	00	31	2E	32	2E	38	34	30	2E	31	31	33	36	31	39	2E	32		....UI4.1.2.840.113619.2
000000D8	2E	31	34	34	2E	31	36	32	37	34	34	30	33	33	38	2E	31	32	36	32	39	2E	31	31		.144.1627440338.12629.11
000000F0	37	33	39	37	34	32	31	30	2E	39	32	39	02	00	10	00	55	49	14	00	31	2E	32	2E		73974210.929....UI..1.2.
00000108	38	34	30	2E	31	30	30	30	38	2E	31	2E	32	2E	31	00	02	00	12	00	55	49	14	00		840.10008.1.2.1.....UI..
00000120	31	2E	32	2E	38	34	30	2E	31	31	33	36	31	39	2E	36	2E	31	34	34	02	00	13	00		1.2.840.113619.6.144....
00000138	53	48	10	00	41	57	34	5F	32	5F	30	34	5F	31	30	5F	45	58	54	20	08	00	00	00		SH..AW4_2_04_10_EXT ....
00000150	55	4C	04	00	AA	01	00	00	08	00	05	00	43	53	0A	00	49	53	4F	5F	49	52	20	31		UL..ª.......CS..ISO_IR 1
00000168	30	30	08	00	08	00	43	53	16	00	4F	52	49	47	49	4E	41	4C	5C	50	52	49	4D	41		00....CS..ORIGINAL\PRIMA
00000180	52	59	5C	4F	54	48	45	52	08	00	16	00	55	49	1A	00	31	2E	32	2E	38	34	30	2E		RY\OTHER....UI..1.2.840.
00000198	31	30	30	30	38	2E	35	2E	31	2E	34	2E	31	2E	31	2E	34	00	08	00	18	00	55	49		10008.5.1.4.1.1.4.....UI
000001B0	34	00	31	2E	32	2E	38	34	30	2E	31	31	33	36	31	39	2E	32	2E	31	34	34	2E	31		4.1.2.840.113619.2.144.1
000001C8	36	32	37	34	34	30	33	33	38	2E	31	32	36	32	39	2E	31	31	37	33	39	37	34	32		627440338.12629.11739742
000001E0	31	30	2E	39	32	39	08	00	20	00	44	41	08	00	32	30	30	37	30	33	30	38	08	00		10.929.. .DA..20070308..
000001F8	21	00	44	41	08	00	32	30	30	37	30	33	30	38	08	00	22	00	44	41	08	00	32	30		!.DA..20070308..".DA..20
00000210	30	37	30	33	30	38	08	00	23	00	44	41	08	00	32	30	30	37	30	33	30	38	08	00		070308..#.DA..20070308..
00000228	30	00	54	4D	06	00	31	35	35	30	34	30	08	00	31	00	54	4D	06	00	31	35	35	30		0.TM..155040..1.TM..1550
00000240	34	30	08	00	32	00	54	4D	06	00	31	35	35	30	34	30	08	00	33	00	54	4D	06	00		40..2.TM..155040..3.TM..

- 02 00 → (0002,)
- 00 00 → (0002,0000)
- 55 4C → UL

Ici est DICM en hexadécimal (44 49 43 4D).

Second exemple,



- 02 00 02 00 → (0002,0002)
- 55 49 → UI
- 1A 00 VL = 0
- 31 2E 32 2E 38 34 30 2E 31 30 30 30 38 2E 35 2E 31 2E 34 2E 31 2E 31 2E 34 00 → 1.2.840.10008.5.1.4.1.1.4.