Medical Image Data Communication

- Before computer networks were widespread, medical images were archived to 9 track magnetic tape in arcane vendor specific formats.

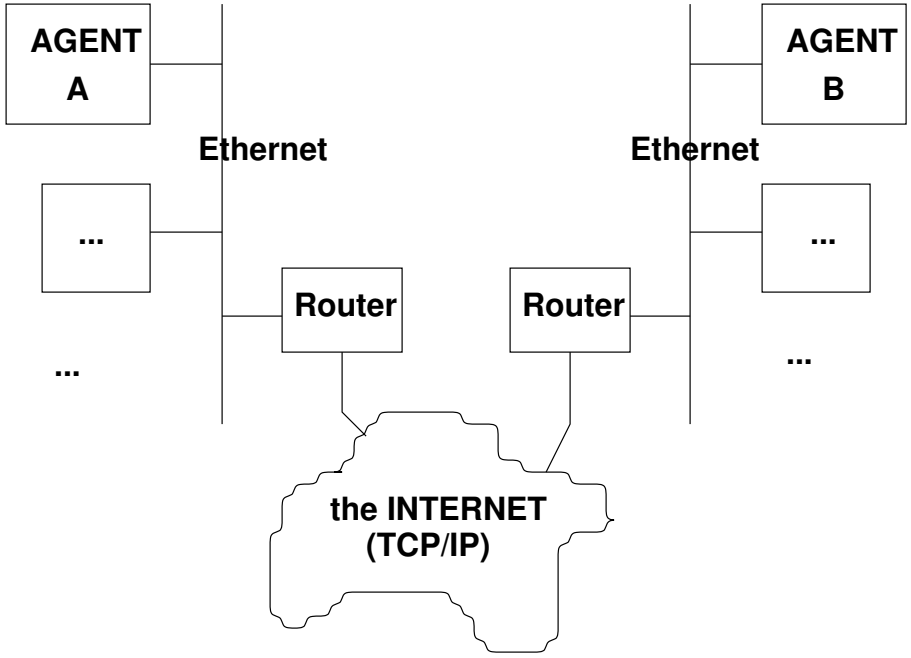
- Applications that needed to import the data included vendor specific code for each tape format.

- The work was often repeated since the output format was *also* arcane and vendor specific.

- Meanwhile, radiology departments were losing films, running out of storage space and silver was getting expensive.

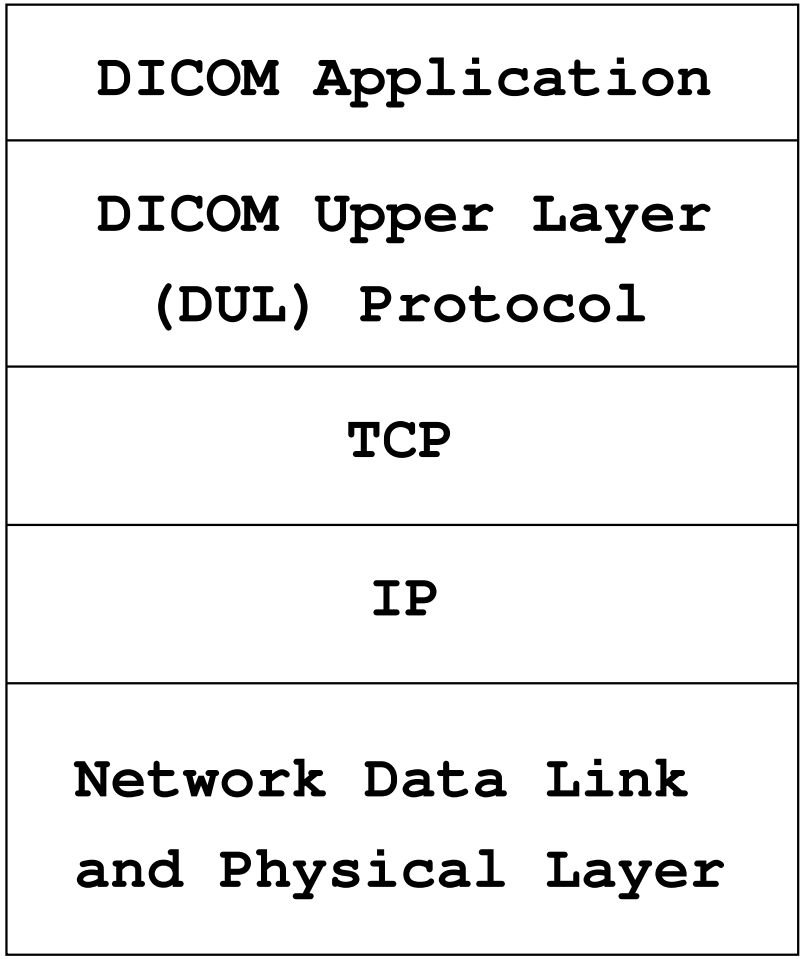
- A standard for image data interchange was needed.

DICOM is a network protocol



DICOM is about program to program communication between computers, not a standard file format, and not information retrieval for use by humans.

DICOM in relation to TCP/IP



Components of the DICOM protocol

- The *DICOM Upper Layer protocol* (DUL) is the language used to make connections, compose, send, receive and decode messages.

- *DICOM Services* are the operations DICOM programs can do, e.g., send, store, look up information.

- *DICOM Objects* are the data that programs can send and receive, e.g., patient data, CT, MR and other images, radiation beams, anatomic structure contours.

A *Conformance Statement* specifies which services and objects a particular implementation provides.

DICOM uses UIDs

- 1.2.840.10008 is the UID prefix assigned to NEMA for DICOM entities.
- UID 1.2.840.10008.3.1.1.1 refers to the data item *Application-Context-Name*.
- UID 1.2.840.10008.1.2 refers to the data item *Transfer-Syntax-Name*.
- UID 1.2.840.10008.5.1.4.1.1.2 refers to the CT Image Storage SOP
- 1.2.840.113994 is the UID prefix assigned to the University of Washington.
- 1.2.840.113994.100.10 is assigned to the UW Radiation Oncology Department.
- 1.2.840.113994.100.10.1.1 is the Implementation Class UID for the Prism DICOM Server.

DICOM PDU types

Code	PDU type symbol	Description
01	:A-Associate-RQ	Association request (from client)
02	:A-Associate-AC	Association accept (from server)
03	:A-Associate-RJ	Association reject (from server)
04	:P-Data-TF	Message containing command(s) and/or data
05	:A-Release-RQ	Association release request
06	:A-Release-RSP	Association release response
07	:A-Abort	Association abort notification

DICOM state definitions

- Sta 1 Idle
- Sta 2 Transport connection open (Awaiting A-ASSOCIATE-RQ PDU)
- Sta 3 Awaiting local A-ASSOCIATE response primitive (from local user)
- Sta 4 Awaiting transport connection opening to complete (from local transport service)
- Sta 5 Awaiting A-ASSOCIATE-AC or A-ASSOCIATE-RJ PDU
- Sta 6 Association established and ready for data transfer
- Sta 7 Awaiting A-RELEASE-RP PDU
- . . .

Table 9-10 DICOM UPPER LAYER PROTOCOL STATE TRANSITION TABLE

STATES / EVENTS	No assoc^{n}	Association establishment				Data transfer	Association release (normal & collision)						Wait for Tp Close
	Sta1	Sta2	Sta3	Sta4	Sta5	Sta6	Sta7	Sta8	Sta9	Sta10	Sta11	Sta12	Sta13
A-ASSOCIATE Request (local user)	AE-1 Sta4												
Transport Conn. Confirm^{n} (local transport service)				AE-2 Sta5									
A-ASSOCIATE-AC PDU (received on transport connection)		AA-1 Sta13	AA-8 Sta13		AE-3 Sta6	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-6 Sta13
A-ASSOCIATE-RJ PDU (received on transport connection)		AA-1 Sta13	AA-8 Sta13		AE-4 Sta1	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-6 Sta13
Transport Connection Indication (local transport service)	AE-5 Sta2												
A-ASSOCIATE-RQ PDU (received on transport connection)		AE-6 Sta3 or 13	AA-8 Sta13		AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-7 Sta13
A-ASSOCIATE response primitive (accept)			AE-7 Sta6										
A-ASSOCIATE response primitive (reject)			AE-8 Sta13										
P-DATA request primitive						DT-1 Sta6		AR-7 Sta8					
P-DATA-TF PDU		AA-1 Sta13	AA-8 Sta13		AA-8 Sta13	DT-2 Sta6	AR-6 Sta7	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-6 Sta13
A-RELEASE Request primitive						AR-1 Sta7							
A-RELEASE-RQ PDU (received on open transport connection)		AA-1 Sta13	AA-8 Sta13		AA-8 Sta13	AR-2 Sta8	AR-8 Sta9 or 10	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-6 Sta13
A-RELEASE-RP PDU (received on transport connection)		AA-1 Sta13	AA-8 Sta13		AA-8 Sta13	AA-8 Sta13	AR-3 Sta1	AA-8 Sta13	AA-8 Sta13	AR-10 Sta12	AR-3 Sta1	AA-8 Sta13	AA-6 Sta13
A-RELEASE Response primitive								AR-4 Sta13	AR-9 Sta11			AR-4 Sta13	
A-ABORT Request primitive			AA-1 Sta13	AA-2 Sta1	AA-1 Sta13	AA-1 Sta13	AA-1 Sta13	AA-1 Sta13	AA-1 Sta13	AA-1 Sta13	AA-1 Sta13	AA-1 Sta13	
A-ABORT PDU (received on open transport connection)		AA-2 Sta1	AA-3 Sta1		AA-3 Sta1	AA-3 Sta1	AA-3 Sta1	AA-3 Sta1	AA-3 Sta1	AA-3 Sta1	AA-3 Sta1	AA-3 Sta1	AA-2 Sta1
Transport connection closed indication (local transport service),		AA-5 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AA-4 Sta1	AR-5 Sta1
ARTIM timer expired (Association reject/release timer),		AA-2 Sta1											AA-2 Sta1
Unrecognized or invalid PDU received		AA-1 Sta13	AA-8 Sta13		AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-8 Sta13	AA-7 Sta13

9.3 DICOM UPPER LAYER PROTOCOL FOR TCP/IP DATA UNITS STRUCTURE

9.3.1 GENERAL

The Protocol Data Units (PDUs) are the message formats exchanged between peer entities within a layer. A PDU shall consist of protocol control information and user data. PDUs are constructed by

A-ASSOCIATE-RQ PDU/A-ASSOCIATE-AC PDU

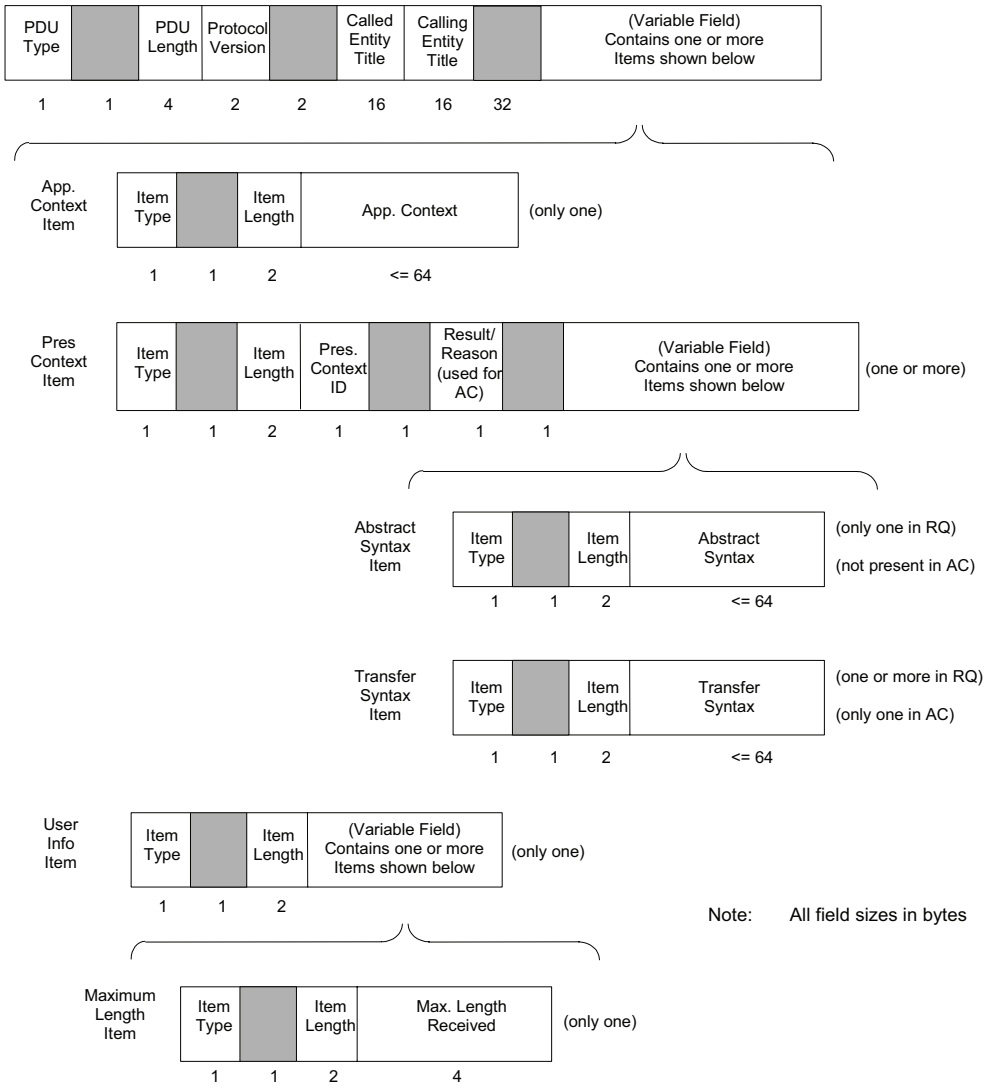


Figure 9-1
PROTOCOL DATA UNITS STRUCTURE AND ENCODING

A trace of an A-Associate-RQ message

```
SEND-PDU: Receiving A-Associate-RQ PDU, 17-Jul-2002 18:13:02.
Dumping Incoming PDU (all 183 bytes):

02 00 00 00 00 b1 00 01 00 00   50 72 69 73 6d 5f 49 6d 61 67
.........Prism_Imag
65 5f 53 72 76 72 50 41 53 53   50 4f 52 54 5f 52 51 20 20 20
e_SrvrPASSPORT_RQ
20 20 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00 00 00
  ..................
00 00 00 00 00 00 00 00 00 00   00 00 00 00 10 00 00 16 31 2e
..................1.
32 2e 38 34 30 2e 31 30 30 30   38 2e 33 2e 31 2e 31 2e 31 00
2.840.10008.3.1.1.1.
21 00 00 1a 01 00 00 00 40 00   00 12 31 2e 32 2e 38 34 30 2e
!.......@...1.2.840.
31 30 30 30 38 2e 31 2e 32 00   50 00 00 31 51 00 00 04 00 01
10008.1.2.P..1Q.....
00 00 52 00 00 1a 31 2e 32 2e   38 34 30 2e 31 31 33 39 34 34
..R...1.2.840.113944
2e 31 30 30 2e 31 30 2e 31 2e   31 00 55 00 00 07 50 44 53 5f
.100.10.1.1.U...PDS_
31 2e 30
1.0
```

A procedural implementation (from CTN)

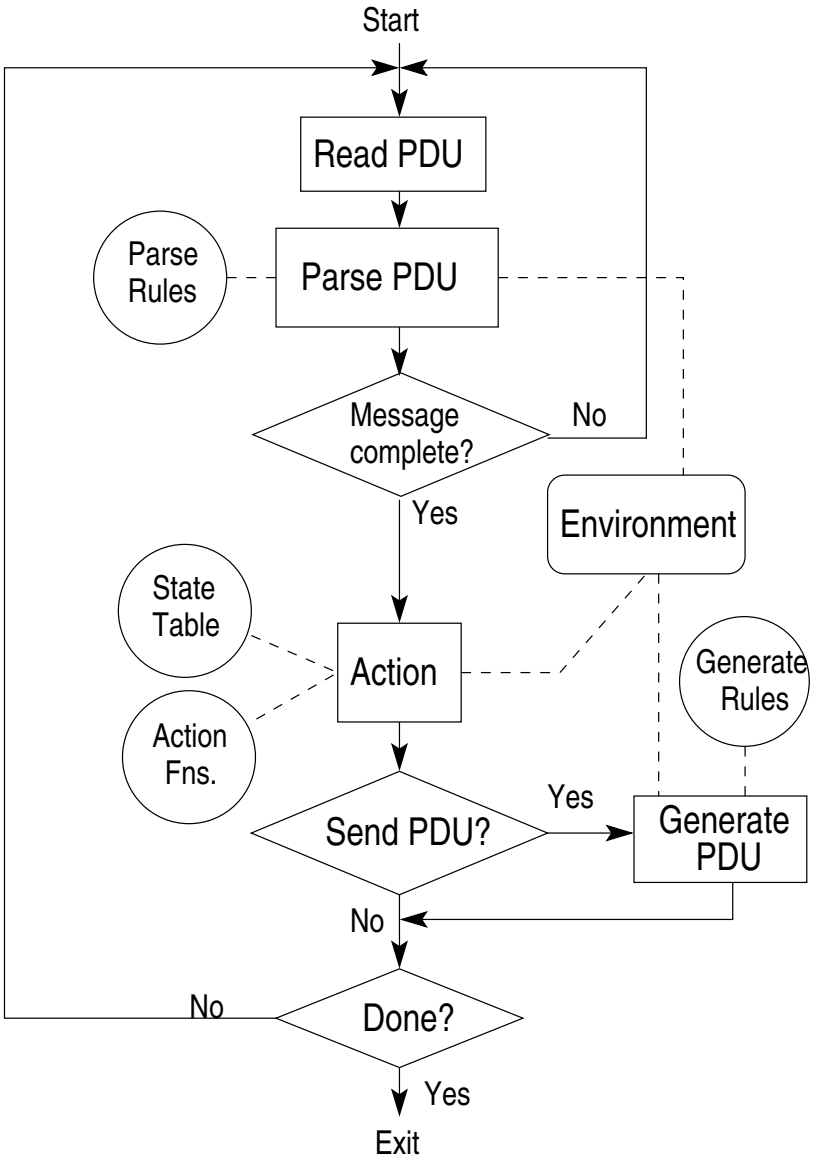
```
CONDITION
parseAssociate(unsigned char *buf, unsigned long pduLength,
       PRV_ASSOCIATEPDU * assoc)
{
    CONDITION cond;
    unsigned char type;
    unsigned long itemLength;
    PRV_PRESENTATIONCONTEXTITEM * context;
...
    (void) strncpy(assoc->calledAPTitle, (char *) buf, 16);
...
    (void) strncpy(assoc->callingAPTitle, (char *) buf, 16);
...
    cond = DUL_NORMAL;
    while ((cond == DUL_NORMAL) && (pduLength > 0)) {
        type = *buf;
...
        switch (type) {
        case DUL_TYPEAPPLICATIONCONTEXT:
            cond = parseSubItem(&assoc->applicationContext,
                                buf, &itemLength);
...
        case DUL_TYPEPRESENTATIONCONTEXTRQ:
        case DUL_TYPEPRESENTATIONCONTEXTAC:
            context = CTN_MALLOC(sizeof(*context));
...
            cond = parsePresentationContext(type, context,
                                            buf, &itemLength);
...
        case DUL_TYPEUSERINFO:
            cond = parseUserInfo(&assoc->userInfo,
                                 buf, &itemLength);
...
```

DICOM is a language

Sentence	$\Leftrightarrow$	DICOM message
Grammar	$\Leftrightarrow$	PDU structure and state table
Vocabulary	$\Leftrightarrow$	DICOM objects and services

We implemented this language with *rules* and a *generic* recursive descent parser, and embedded it in a state machine.

DICOM parser and state machine



DICOM state machine implementation

```
(defvar *event* nil)

(defun dicom-state-machine (socket)
  (let ((tcp-stream nil)
        (state 'state-01)
        (env nil)
        (trans-data nil)
        (action-fn nil))
    (loop
      (case state
        (state-01 (setq tcp-stream (accept-connection socket
                                                      :wait t)
                        *event* 'event-05))
        ((state-02 state-05 state-07 state-10 state-11 state-13)
         (setq env (read-pdu tcp-stream env))))
      (setq trans-data (get-transition-data state *event*)
            action-fn (first trans-data)
            env (funcall action-fn env tcp-stream)
            state (second trans-data)))))
```

State table entry for Data Transfer state

```
(state-06

  "Association established, ready for data transfer"

  ((event-09) dt-01 state-06)
  ((event-10) dt-02 state-06)
  ((event-20) dt-03 state-06)
  ((event-21) dt-04 state-06)
  ((event-11) ar-01 state-07)
  ((event-12A event-12B) ar-02 state-08)
  ((event-16) aa-03 nil)
  ((event-17) aa-04 nil)
  ((event-15) aa-01 state-13)
  ((event-03 event-04 event-06 event-13 event-19)
              aa-08 state-13))
```

Parse rule for A-Associate Request PDU

```
(:A-Associate-RQ
  #x01                ; A-Associate-RQ PDU Type tag
  =ignored-byte       ; Reserved field [1 byte]
  (=ignored-bytes 4)   ; PDU Length [4 bytes]
  (>decode-var Protocol-Version fixnum 2 :Big-Endian)
  (=ignored-bytes 2) ; Reserved field -- not used
  (>decode-var Called-AE-Title string 16 :Space-Pad)
  (>decode-var Calling-AE-Title string 16 :Space-Pad)
  (=ignored-bytes 32)  ; Reserved field -- not used
  :Application-Context-Item
  (:Repeat (1 :No-Limit) :Presentation-Context-Item-RQ)
  :User-Information-Item)
```

Parse rule for Abstract-Syntax-Item

```
(:Abstract-Syntax-Item-RQ

  #x30                ; Abstract Syntax Item type tag

  =ignored-byte             ; Reserved field [1 byte]

  ;; Abstract Syntax Name field length [2 bytes]

  (>decode-var ASN-Len fixnum 2 :Big-Endian)

  ;; Abstract Syntax Name string - variable-length

  (>decode-var ASN-Str string (<lookup-var ASN-Len) :No-Pad))
```

An excerpt from the DICOM data dictionary table

```
(defparameter *group/elemname-alist*
  ;;-----------------------------------------
  ;; Group 0000: CMD "Command"
  '(((#x0000 . #x0000) UL "Group Length")
    ((#x0000 . #x0002) UI "Affected SOP Class UID")
    ((#x0000 . #x0003) UI "Requested SOP Class UID")
    ((#x0000 . #x0100) US "Command Field")
    ((#x0000 . #x0110) US "Message ID")
    ((#x0000 . #x0120) US "Message ID Responded To")
    ((#x0000 . #x0600) AE "Move Destination")
    ...
    ;;-----------------------------------------
    ;; Group 0010: PAT "Patient Information"
    ((#x0010 . #x0000) UL "Group Length")
    ((#x0010 . #x0010) PN "Patient Name")
    ((#x0010 . #x0020) LO "Patient ID")
    ((#x0010 . #x0021) LO "Issuer of Patient ID")
    ((#x0010 . #x0030) DA "Patient's Birthdate")
    ((#x0010 . #x0032) TM "Patient's Birth Time")
    ...
    ;;-----------------------------------------
    ;; Group 0028: IMG "Image"
    ((#x0028 . #x0000) UL "Group Length")
    ((#x0028 . #x0002) US "Samples Per Pixel")
    ((#x0028 . #x0005) RET "Image Dimensions (RET)")
    ((#x0028 . #x0006) US "Planar Configuration")
    ((#x0028 . #x0010) US "Rows")
    ((#x0028 . #x0011) US "Columns")
    ((#x0028 . #x0012) US "Planes")
    ...
    ))
```

An excerpt from the value representation table

```
(defparameter *datatype-alist*
  '((AE "Application Entity" (string 0 16) :Space-Pad)
    (AS "Age String" (string 4) :No-Pad)
    (AT "Attribute Tag" (fixnum 4))
    (CS "Code String" (string 0 16) :Space-Pad)
    (DA "Date" (string 8) :No-Pad)
    (DS "Decimal String" (string 0 16) :Space-Pad)
    (FD "Floating-Point Double" (double-float 8))
    (FL "Floating-Point Single" (single-float 4))
    (IT "Item in Sequence")
    (ITDL "Item Delimiter")
    (OB "Other Byte" ((unsigned-byte 8) 0 *) :Null-Pad)
    (OW "Other Word" ((unsigned-byte 16) 0 *) :No-Pad)
    (SL "Signed Long" (fixnum 4))
    (SQ "Sequence of Items")
    (SQDL "Sequence Delimiter")
    (SS "Signed Short" (fixnum 2))
    (ST "Short Text" (string 0 1024) :Space-Pad)
    (TM "Time String" (string 0 16) :Space-Pad)
    (UI "Unique Identifier" (string 0 64) :Null-Pad)
    ...
    ))
```

The Prism Image file format

```
PRISM::IMAGE-2D
  PRISM::ID  1
  PRISM::DESCRIPTION  "Orig: ca pancreas"
  PRISM::ACQ-DATE  "24-May-1994"
  PRISM::ACQ-TIME  "16:40:23"
  PRISM::IMG-TYPE  "X-ray CT"
  PRISM::ORIGIN  #(-17.25 17.25 0.0)
  PRISM::SCANNER-TYPE  "GE 9800"
  PRISM::HOSP-NAME  "University Hospital, Seattle"
  PRISM::RANGE  4095
  PRISM::UNITS  "H - 1024"
  PRISM::SIZE  (34.504 34.504)
  PRISM::PIX-PER-CM  14.8388
  PRISM::THICKNESS 0.500
  PRISM::X-ORIENT #(1.000 0.000 0.000)
  PRISM::Y-ORIENT #(0.000 -1.000 0.000)
  PRISM::PIXELS  ("pat-1.image-1-1" 512 512)
:END

PRISM::IMAGE-2D
  PRISM::ID  2
  PRISM::DESCRIPTION  "Orig: ca pancreas"
  PRISM::ACQ-DATE  "24-May-1994"
  PRISM::ACQ-TIME  "16:40:23"
  PRISM::IMG-TYPE  "X-ray CT"
  PRISM::ORIGIN  #(-17.25 17.25 1.0)
  ...
```

PDS code (February 2003)

PDS is coded entirely in Common Lisp

- Code: 5991 lines (2700 common, 1930 server, 1361 Prism)
- Data dictionary: 1514 lines
- Rules: 1762 lines
- State table: 227 lines
- Total: 9494 lines, including about 2500 lines of comments

The subset of CTN corresponding to the core above (about 8,000 lines) is about 31,500 lines of C.

Initial experience with PDS

- Successfully tested with:
  - CTN test programs
  - Elscint CT at Soroka Medical Center, Beersheva
  - GE CT and MR at the University of Washington
  - Philips CT at Washington State University
  - Picker CT at the Seattle VA Medical Center
- Clinical implementation in December 1999
- Five errors discovered since initial implementation
- Runs as a server daemon on our Linux cluster

Ingredients for DICOM-RT

For the DICOM-RT client (part of Prism):

- DICOM Upper Layer Protocol support (added 823 lines),
- DICOM-RT object definitions (already in data dictionary),
- C-STORE SOP with RT objects,
- Translation from Prism objects to RT objects,
- User interface (last three added 2947 lines)

The Prism DICOM-RT facility was the first DICOM-RT implementation to successfully communicate with an Elekta radiation therapy machine.

Why has DICOM been a big success?

Factors that helped DICOM become widely adopted:

- agreement among radiologists and medical physicists on the semantics of the data,
- moving from a proprietary point-to-point design to one that works with TCP/IP,
- availability of the CTN code,
- urgent need and precedent (digital data was *already* being moved from computer to computer, by tape and floppy disk),
- insistence from customers.

Future work

- Extend implementation to support other functions ("query and retrieve") and additional data objects,
- Reimplement the Prism data storage as a DICOM-accessible repository,
- Further improve modularity, to produce a DICOM software kit,
- Experiment with the core code, to implement other protocols, e.g., HL7.

Thanks

- Steve Moore, Washington University, St. Louis, author of CTN,
- Mark Phillips, Sharon Hummel and Homayon Parsai, University of Washington Medical Center,
- Marv Kleven, David Haynor and Steve Sutlief, Seattle VA Hospital,
- Pat Gavin and Hege Kippenes, Washington State University Veterinary Hospital,
- Andrew Long and Wayne Avren, Elekta Oncology Systems,
- Ludwig Avraham, Sergio Faermann, Yanai Krutman, Yaron Leibovich, Svetlana Zalmanov, and Ilan Zelingher, Soroka University Medical Center,
- Eran Lachs, Ben-Gurion University Computer Center,
- Yoram Cohen, Chairman of the Oncology Department of Soroka University Medical Center,
- Giora Yahel, Zvi Bar, and Hen Lederman from Marconi/Philips Israel,
- Mark Pepelea and John Harkey from Marconi USA.