Stated Topic and Goal

*Determine the feasibility of using binary DICOM for building browser based medical imaging applications*

Method:

* Design and implement a DICOM editor that reads and writes binary DICOM and displays it using HTML5, CSS3 and the Dart programming language.
* Test performance by reading, displaying and writing DICOM studies in binary format.
* Goal: Read and display imaging studies in less than 3 seconds.
Relevance/Importance

• DICOM is the standard by which medical imaging information is transmitted, stored, and displayed for use by healthcare providers and their healthcare organizations.

• Nearly all healthcare systems use the DICOM standard in their medical equipment and information systems.

• The DICOM Standard does not specify any application implementation details.
Zero-footprint Client

• It is crucial that information systems be created that allow end users to access and edit DICOM data in a fast, secure, and intuitive manner.

• Zero Clients have many advantages over downloaded software applications:
  – HIPAA covered patient data is not stored on the end point device.
    • Data cache can be encrypted.
  – Applications can be maintained and upgraded without the need for access control on the end point device.
  – Improved security
  – Lower cost (management, implementation, security).
Dart

• Class based and object oriented programming language for creating web applications.
• Dart programs compile to JavaScript (dart2js) or native in Chromium.
• Native Dart code is much faster than JavaScript and dart2js during benchmarking.¹
• Development overhead can be simpler
  — Dart has classes, first class procedures, and optional static typing.

¹ DeltaBlue, FluidMotion, Richards, Tracer Benchmarks (https://www.dartlang.org/performance/)
Technical Summary and Approach

• Learn the DICOM standard to understand how medical imaging data is stored.
• Design a data structure for DICOM studies.
• Create a project plan including packages and classes for our viewer/editor.
• Learn the Dart syntax and translate our plan into code.
  – Develop unit tests for each class
• Iteratively optimize code.
Data Structures: Top Level

Information Model
- PatientStudies
- Patient
- Study
- Series
- Instance
- Frame

Data Types
- Dataset
- Sequence
- Attribute
- aType (attribute type)
- VR (Value Representation)
- ValueType
- VM (Value Multiplicity)
Data Structure: Top Level

DICOM (structured study)

• Patient
• Study
  — Series
    — Instance
    — Frame
Data Structure: DICOM Internals

**Dataset**

<table>
<thead>
<tr>
<th>Tag</th>
<th>VR</th>
<th>Length</th>
<th>Value Field</th>
</tr>
</thead>
</table>

- **Number (hex)**
- **Name**
- **Keyword**
- **VR**
- **VM**
- **Retired**

Data Types

Tag can have a value of sequence.
Value field is an array of items and each item contains a data set.
Packages and Classes

• Study
  – Atype
  – Bulkdata
  – DataSet
  – Instance
  – PatientName
  – QC-mode
  – Sequence Series
  – Studies Study
  – TagDictionary
  – UT
  – ValueField
  – VM
  – VR

• I/O
  – BulkdataInputStream
  – BulkdataOutputStream
  – MetadataInputStream
  – MetadataOutputStream

• Test
  – ReadBulkdata
  – ReadMetadata
  – WriteBulkdata
  – WriteMetadata

• Test
  – ReadThenWriteMetadata
  – VM-Test

• Display
  – DisplayImage
  – Study
  – WorkList

• Util
  – ByteUtils
  – DateRange
  – DateUtils
  – StringUtils
  – TagUtils
  – UIDUtils
  – UUIDUtils
Deliverables

• Minimum deliverables
  – Read and display DICOM in a browser and then write it
  – Build a test program that compares input and output to validate correctness
  – Create unit tests for each class

• Expected deliverables
  – Display a work list of studies of n patients
  – Display patient as collapse/expand tree for study information model

• Maximum deliverables
  – Edit metadata
  – Display images and add overlay information
  – Encrypt and decrypt studies using AES (GCM) using an encryption framework created at Hopkins Security Institute.
Dependencies

• Access to our mentor
• Computer to write code
• Bitbucket to share code
• Dart & DICOM Reference Information
• Access to DICOM Test Data
Management Plan

• Meet twice a week to work on code as a team.
  • Mondays & Fridays at 10:00AM

• Weekly meetings with Dr. Philbin
  • Wednesdays at 9:30AM
  • Code reviews as classes and modules develop

• More meetings can be scheduled as required.
Key Dates

• **February 20:** Have project proposal finished and all of the programming planned and reviewed by Dr. Philbin
• **March 6:** Read input (parse)
• **March 20:** Write and validate output
• **April 3:** HTML5/CSS3 display metadata
• **April 17:** Edit data
• **May 1:** Display/Edit images
• **May 9:** Final Poster Presentation
Reading List

- MINT Toolkit source code and documentation (provided by Dr. Philbin)
- DICOM Standard
- Dart documentation