THE UNIVERSITY OF TEXAS
MD Anderson Cancer Center

LIS Interfaces: Basics, Implementation, and Pitfalls

Mark Routbort, MD, PhD
UT MD Anderson Cancer Center

Pathology Informatics Review – Pathology Informatics 2014

Grateful acknowledgements to John Sinard, MD, PhD

api PATHOLOGY INFORMATICS

Laboratory System Interfaces

- No laboratory/pathology information system is an island ….
- Information from lab/path systems underlies patient treatment and diagnosis, at multiple levels:
  - Transactional
    - Hematocrit
    - Do I need to transfuse this patient with anemia?
  - Problem list
    - Coombs test positive
    - Hemolytic anemia, will require careful monitoring and possibly therapy
  - Persistent
    - Diagnosis: CLL
    - This patient has chronic lymphocytic leukemia, with subsequent paraneoplastic autoimmuno hemolytic anemia
- Understanding how this information is transmitted (and represented) is core the to practice of clinical informatics for the laboratorian

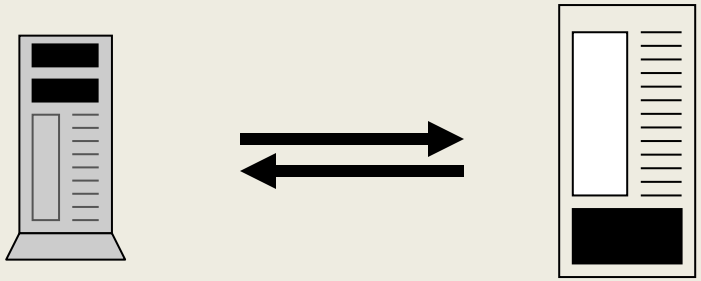
Electronic Interfaces

- Hardware/software used to get data into and out of an information system electronically, usually for communication with another computer system
- Most common input interfaces:
  - ADT (admission/discharge/transfer – patient demographics)
  - Instrument
  - Orders
- Most common output interfaces:
  - Billing
  - Results
  - Orders

Foundational Issues

- Connectivity
  - How does the electronic data get from one computer system to another?
- Comprehension
  - Syntax
    - How is the message structured?
    - How do I know when one message ends and another begins?
    - Where do I look in the message to find a particular data element?
  - Semantics
    - What does the data mean?

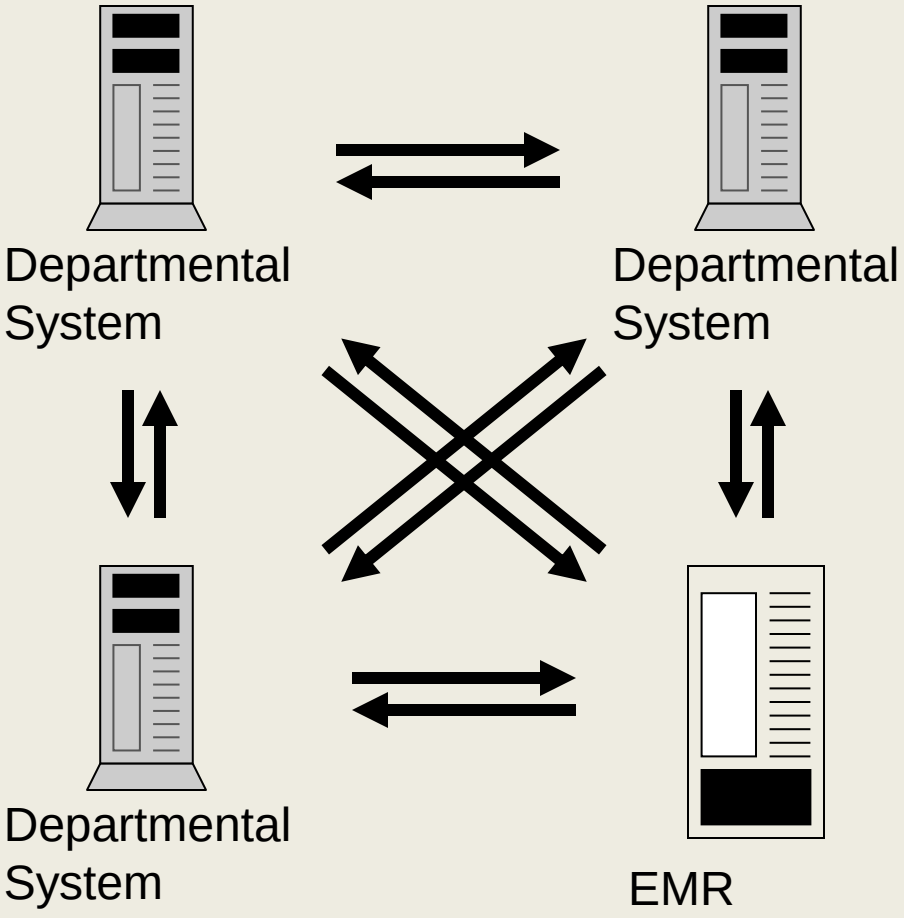
Connectivity



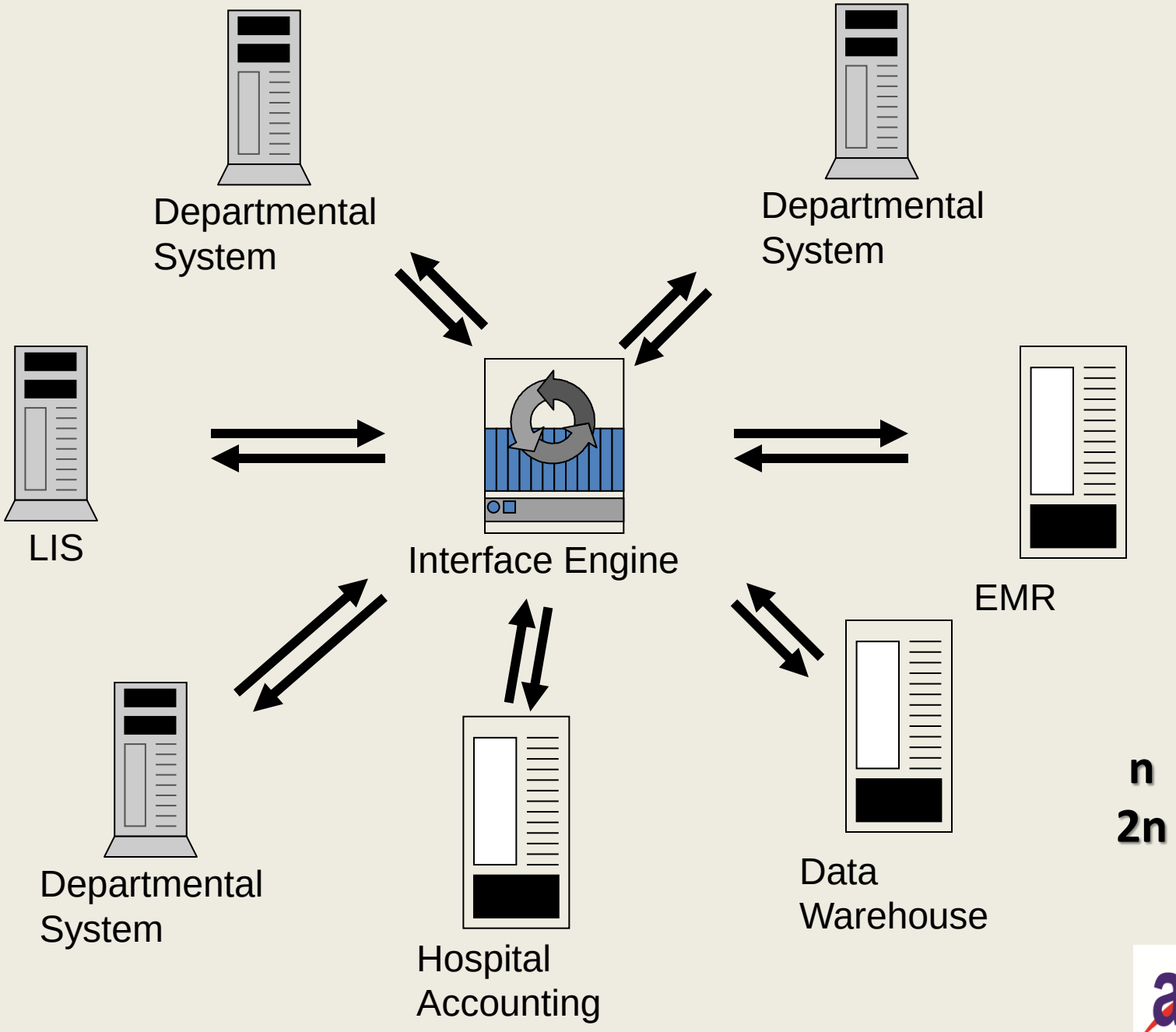
LIS EMR

 # Systems: 2

 # Interfaces: 1



# Systems:	2	→ 3 →	4	**n**
# Interfaces:	2	→ 6 →	12	**n(n-1)**


Departmental System
Departmental System
LIS
Interface Engine
EMR
Departmental System
Hospital Accounting
Data Warehouse
n
2n

Elements of an interface engine
I/O Handler

- Receives data from and outside system and stores it in an input table OR takes data from an output table and sends it to an outside system
- Communication Modes:
  - TCP/IP (direct “real time” communication)
  - Multiple Files (each “message” is put into its own file)
  - Single File (appends new data on to end of file)
  - Serial
  - Custom
- The I/O Handler knows nothing about the content of the message, only how to communicate it (and what format will be used to process it)

Elements of an interface engine
I/O Processor

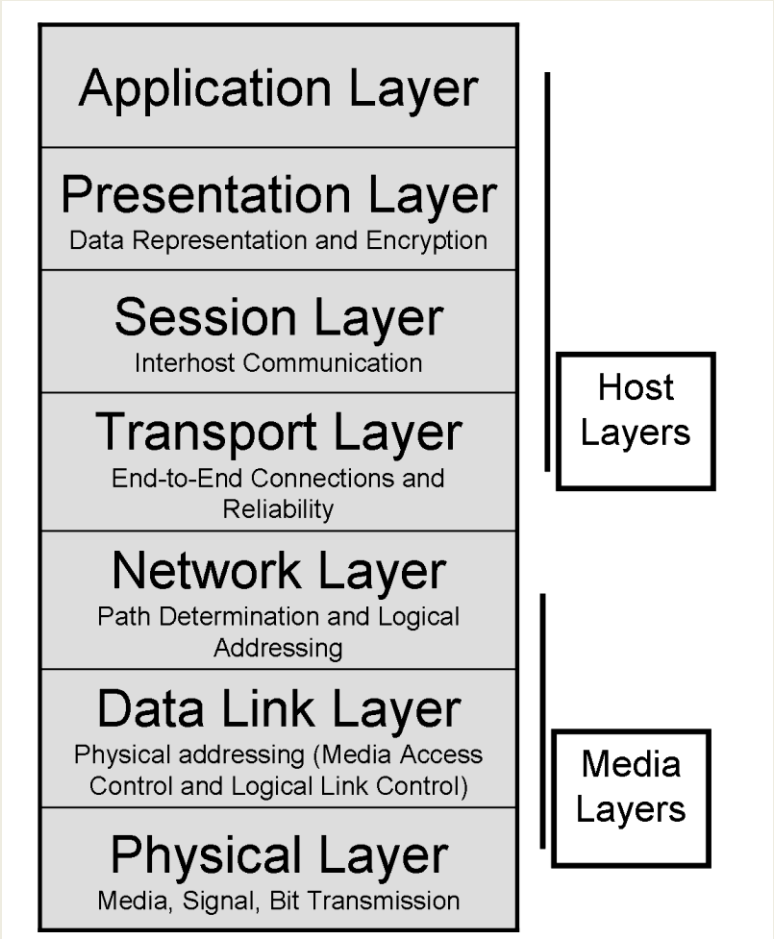
- Breaks down the message into its component data and stores it in the database OR gets data from the database and builds an output message
- Message Segmentation Methods (format types):
  - Delimited
    - Comma or tab separated values
    - In HL7: | ^ & ~
  - Fixed Length Fields (typically used for conversion)
  - More complex segmentation types
    - XML – eXtended Markup Language
    - JSON – Javascript serial object notation
- The I/O Processor is responsible for dealing with the CONTENT of the message, but is not at all involved in communicating it

HL7

- Health Level Seven standard for the electronic exchange of data in healthcare environments, with specific emphasis on inpatient acute care facilities
- Represents a “common syntax” which software from different vendors can use to communicate with each other
- Based on the theory that although each healthcare system stores data in different ways, both with respect to the database structure and data encoding, the data elements stored and the concepts they represent are essentially the same

The “7” in HL7 refers to the Seventh Level of the OSI Model

The “Open Systems Interconnect” (OSI) Model: Layer-based reference model for computer networking developed by the International Standards Organization (ISO)



All client applications. Internet browsers, email, telnet

Character-code translation, data conversion, data compression

Controls when applications/users can send or receive data; synchronization

Monitors the quality of the connection (data integrity)

Fragmentation/re-assembly; routing data among nodes

Provides for the error free transmission of data frames

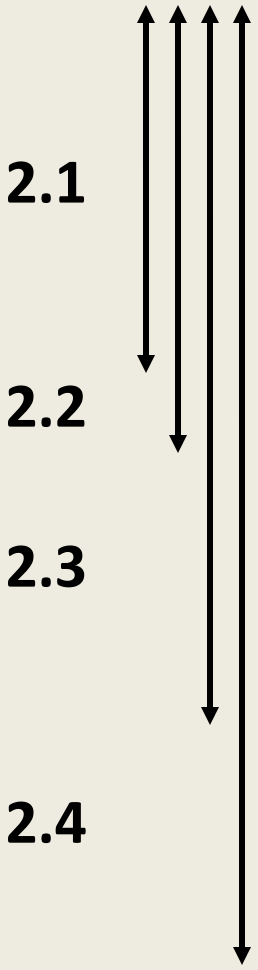
Specifics of voltages, cabling, connectors

api PATHOLOGY INFORMATICS

History of HL7

- Initially developed by 12 volunteers (the “HL7 working group”)
  - Version 1 released in 1987
  - Version 2 released in 1988
- 1994: HL7 working group became an American National Standards Institute (ANSI) accredited standards organization
- Version 2
  - Uses a delimited message format
  - Version 2.5 released in 2003
  - Version 2.7 released in 2009
- Version 3
  - Uses XML (eXtensible Markup Language)
  - Initial release published 2005
  - Very limited penetrance in the US

HL7 Supported Message Specifications



- Admit/Discharge/Transfer (ADT)
- Order Entry
- Query
- Finance (Billing)
- Observation Reporting (Results)
- Master Files
- Medical Records/Information Management
- Scheduling
- Patient Referral
- Patient Care
- Laboratory Automation
- Application Management
- Personnel Management

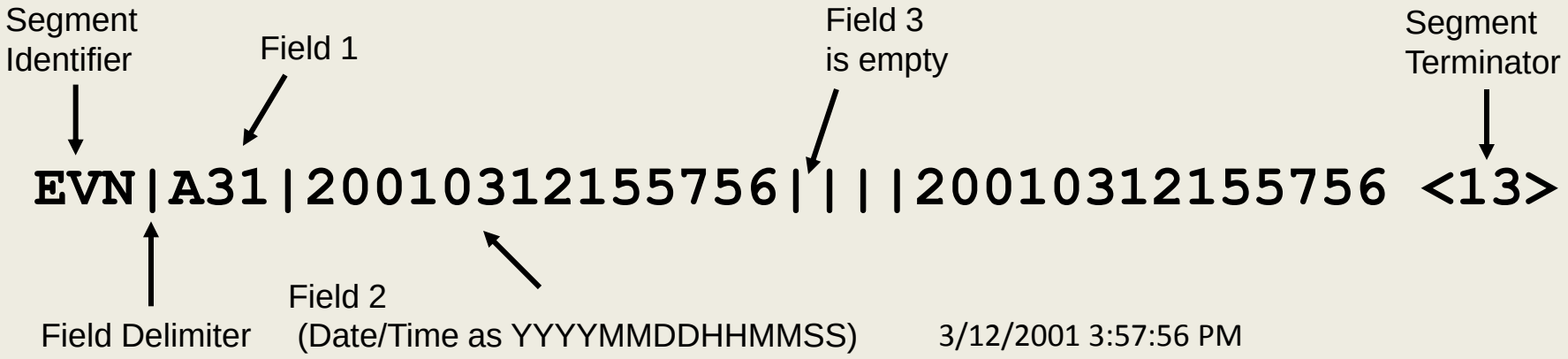
HL7 Message Construction Rules

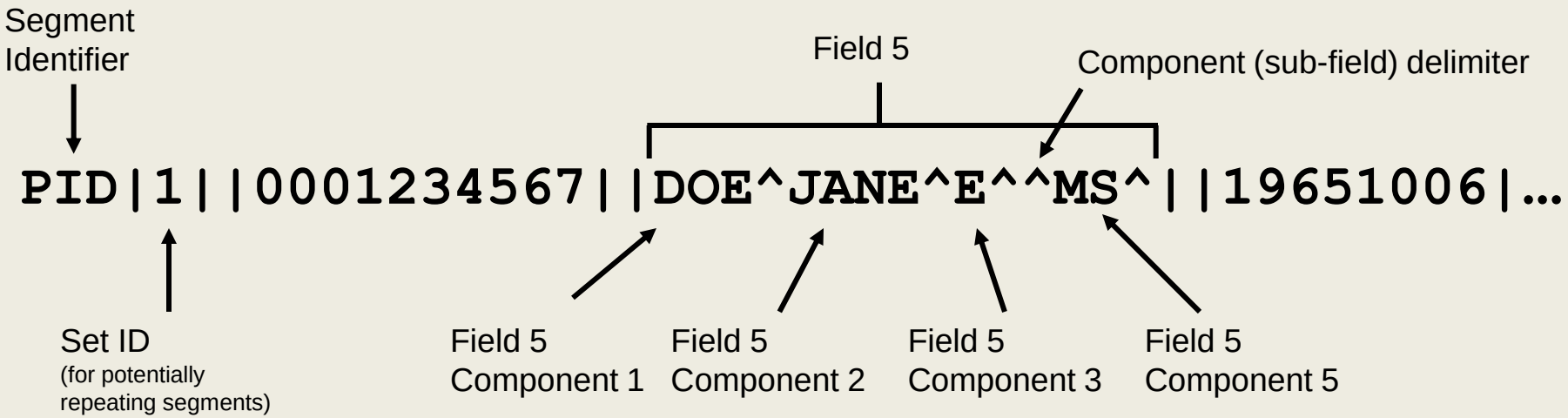
- Messages are composed of varying length segments, each terminated by a carriage return (ASCII 13).
- Segments may be defined as required or optional, and may be allowed to repeat
- Each segment is a series of variable length fields, separated by field separator characters (delimiters)
- Each segment begins with a 3 character identifier
  - Custom (site specific) segment identifiers must start with a "Z"
- Individual data fields are identified solely by their position within the segment

HL7 Message Construction Rules (cont)

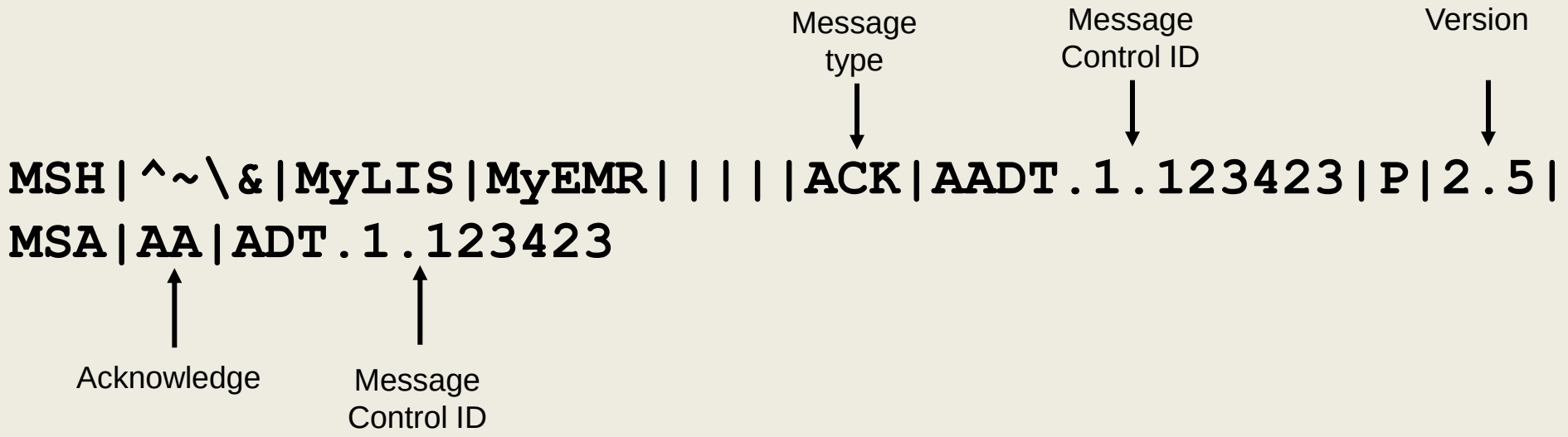
- Each message begins with a message header (MSH)
- Every message has a “type” – which corresponds to its real-world purpose and further specifies what segments it must/can contain
  - Contained in MSH-9 (ninth field of MSH segment)
  - Examples
    - Patient registration
    - Laboratory order
    - Laboratory result

Sample HL7 Segments





The simplest message ACK (nowledgment)



Acknowledge Status	Meaning
AA	Positive acknowledgment: the message was successfully processed.
AE	Application error
AR	Application reject

Cracks in the ACK …

- Does an acknowledgement mean:
  - Message received?
  - Message received and processed?
  - Message received and processed without error?
- Relatively few systems/interfaces use the AE/AR rejection/error codes
  - What does a sending system **DO** with a rejection?
- There are marked differences between what the HL7 message interfaces allows and what vendors have implemented
- Much of the limitations of HL7 are associated with the “lowest common denominator” phenomenon

Some important message types

ADT^A04	Register a patient
ADT^A08	Update patient information
ADT^A01	Admit/visit notification
ORM^O01	Order message
ORU^R01	Observation message (results)

Sample ORU message

MSH|^~\`|674|GHC|SISRL|PAML|20060922162830|L674-
20060922162831022O|ORU^R01|ORU000016168|P|2.3|||AL |AL
PID|1||1478895^4^M10^PA||XTEST^PATIENT^||19591123| F||||||||||||||||||||
ORC|RE|F4334|51013174200601|||||^|||||||||||||||
OBR|1|F4334|51013174200601|80048^BASIC METABOLIC
PANEL|||20060922152300|||||||||023901^PACLAB|||||||CH|F||20060922162659^^GHA
|||||||||||||||||
OBX|1|NM|84295^SODIUM^GH|1|145|mmol/L|||||F|||20060922152300|GH
OBX|2|NM|84132^POTASSIUM^GH|2|5.2|mmol/L|||||F|||20060922152300|GH
OBX|3|NM|82435^CHLORIDE^GH|3|108|mmol/L|||||F|||20060922152300|GH
OBX|4|NM|82374^CARBON DIOXIDE^GH|4|31|mmol/L|||||F|||20060922152300|GH
OBX|5|NM|82947^GLUCOSE^GH|5|76|MG/DL|||||F|||20060922152300|GH
OBX|6|NM|84520^BUN^GH|6|22|MG/DL|||||F|||20060922152300|GH

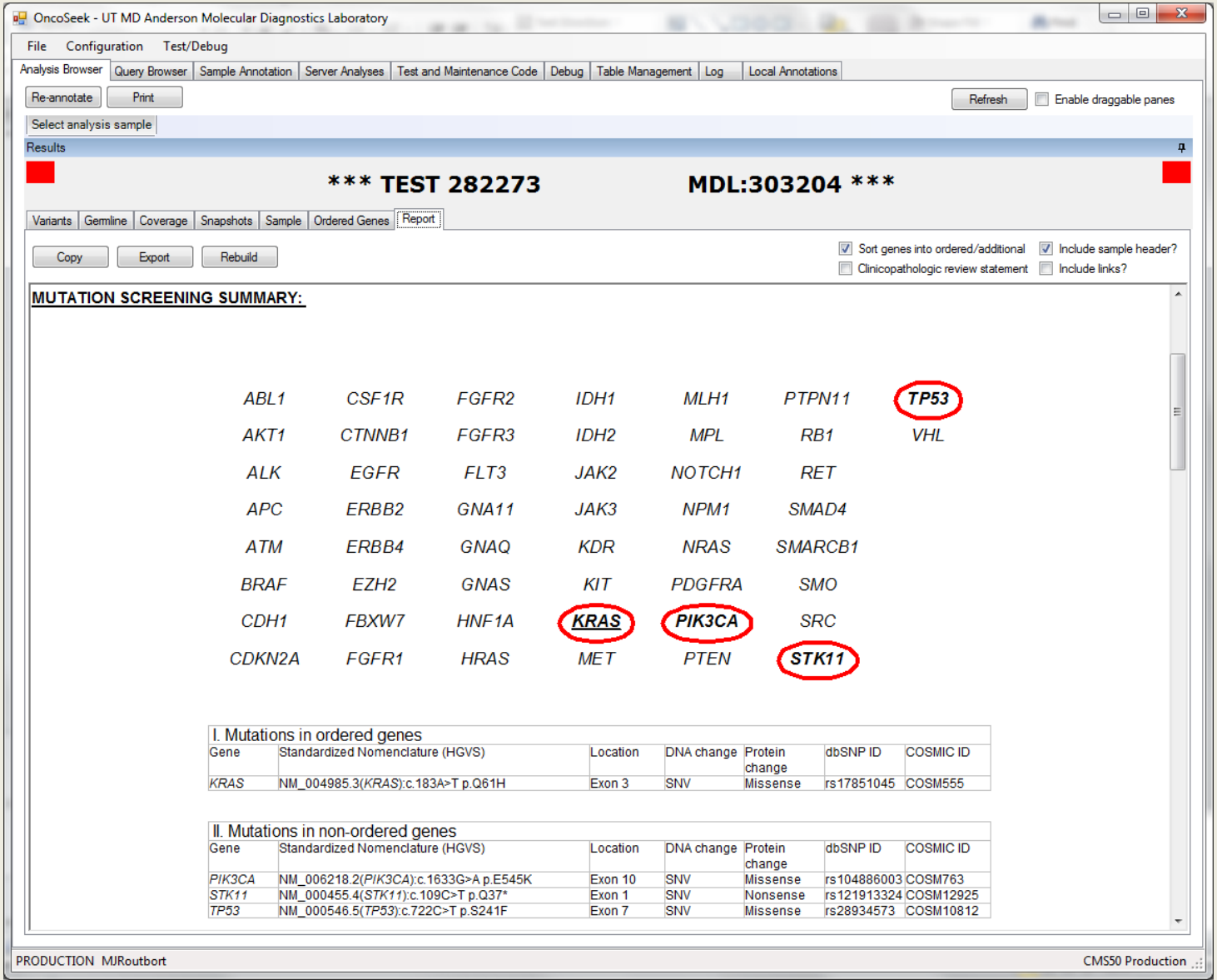
OBX – observation segment

- OBX-1: Set ID - OBX (SI) **optional**
- OBX-2: Value Type (ID)
- OBX-3: Observation Identifier (CE)
- OBX-4: Observation Sub-ID (ST) **optional**
- OBX-5: Observation Value (Varies) **optional repeating**
- OBX-6: Units (CE) **optional**
- OBX-7: References Range (ST) **optional**
- OBX-8: Abnormal Flags (ID) **optional repeating**
- OBX-9: Probability (NM) **optional**
- OBX-10: Nature of Abnormal Test (ID) **optional**
- OBX-11: Observ Result Status (ID)
- OBX-12: Date Last Obs Normal Values (TS) **optional**
- OBX-13: User Defined Access Checks (ST) **optional**
- OBX-14: Date/Time of the Observation (TS) **optional**
- OBX-15: Producer's ID (CE) **optional**
- OBX-16: Responsible Observer (XCN) **optional**
- OBX-17: Observation Method (CE) **optional repeating**

LIS/OBX model limitations

- Observables needs to be tightly defined and adhere to the basic lab test model:
  - Test
  - Result value
  - Result units
  - Result flags
  - Reference range
- More complex data can primary be represented by either
  - Going “free text” – embedding a bunch of sequential OBX segments containing “sentences” to transmit a free text report
  - Embedding or linking to a PDF or other binary data form

Increasing amounts of laboratory data do not fit the rigid observable/OBX model



LIS/OBX model limitations

- More complex data can primary be represented by either
  - Going “free text” – embedding a bunch of sequential OBX segments containing “sentences” to transmit a free text report
  - Embedding or linking to a PDF or other binary data form
- Either of these approaches obviate much of the benefit of HL7 and reduce it to a message transport syntax
- A hybrid model is possible and probably the best option
  - PDF or well-defined representational format for complex, uncommon reports
  - HL7 OBX model for numeric values that need to be trendable or drive clinical decision support

Billing Batch Interfaces

- Usually operate in “Batch” mode
  - Don’t send a bill as each case is signed out
  - On a daily basis, compile a list of charges, verify the batch for errors
  - “Send” the batch: create an HL7 message with the patient information and charging information
  - The message can either be transmitted electronically or saved in a text file and then transferred to the billing system
- Usually have additional “housekeeping” segments (batch header and trailer segments) to verify that the entire batch is complete
- Details of segment contents varies a lot with different billing systems

Summary of HL7

- Nearly ubiquitous standard
- But there is no standard implementation
- Two vendor systems which say they "support" HL7 may not necessarily connect with each other
  - More important is the domain of information interchanged
  - And the mapping of the underlying data models
- One way to ask how efficient/scalable HL7 is is to ask the question:
  - For integrated EMR systems, e.g. Epic, CERNER – how many use HL7 as their internal communication framework between Lab and Clinician?
  - Answer: None
  - These systems either use direct database access or a services archictecture
  - This is frequently advanced as a reason for the different functionality which may be available between Lab and Clinician when using an "interfaced" versus "integrated" solution

Other segmentation/data representation models of importance in pathology informatics

- Simple delimited
- ODBC
- XML
- JSON

Simple delimited

- Delimiter separated values
  - CSV – comma separated values
  - TSV – tab separated values
- Usually lines separated by carriage returns
- Commonly, the first line (header) contains column names
- Generic mechanism for sharing 2-dimensional data (Excel-like)
- Ubiquitous “lingua franca” for device/result interfacing outside the LIS intrument interface pathway
  - List of samples to run
  - Output data for those samples

Caveats for simple delimited data

- Handling delimiter characters within the data

```
Name,MRN,Sample,Concentration
Smith,1234,M-14-123,2.45
Doe, John,2345,M-14-124,1.94
Frank,3456,M-14-125,1.74
```

Name	MRN	Sample	Concentration
Smith	1234	M-14-123	2.45
Doe, John	2345	M-14-124	1.94
Frank	3456	M-14-125	1.74

Name	MRN	Sample	Concentration	
Smith	1234	M-14-123	2.45	
Doe, John	2345	M-14-124	1.94	1.94
Frank	3456	M-14-125	1.74	

Caveats for simple delimited data

- Content is unstructured
  - Any validation is left to the specific implementation
  - Subject to disruptive change
    - On software versioning
    - At boundary states
- Data is untyped
  - No differentiation between numeric, string, date data types

ODBC

- Open Database Connectivity
- Standard middleware API (application programing interface) for accessing database management systems
- Database and operating system independent
- Independence is accomplished by using ODBC drivers as a translation layer between applications and the DBMS
- ODBC-compliant applications can access any DBMS for which a driver is installed
- Drivers exist for
  - all major DBMS
  - other data sources (Excel)
  - Text/flat delimited files

ODBC use cases

- Reports
  - Many LISs use a third party report generation tool like Crystal Reports
  - These tools uniformly use ODBC to access the data
- Data extraction
  - Nearly all LISs have a DBMS back-end
  - Many support direct ODBC read access to the underlying data for extraction for more complex off-line analysis or reporting

XML

- “Extensible Markup Language (XML)”
- A markup language that defines a set of rules for encoding documents and data in a format that is both human-readable and machine-readable
- Defined in specification produced by the W3C as a set of open standards
- Widely used for the representation of arbitrary data structures
- Many application programming interfaces (APIs) have been developed to aid software developers with processing XML data, and several schema systems exist to aid in the definition of XML-based languages.

XML

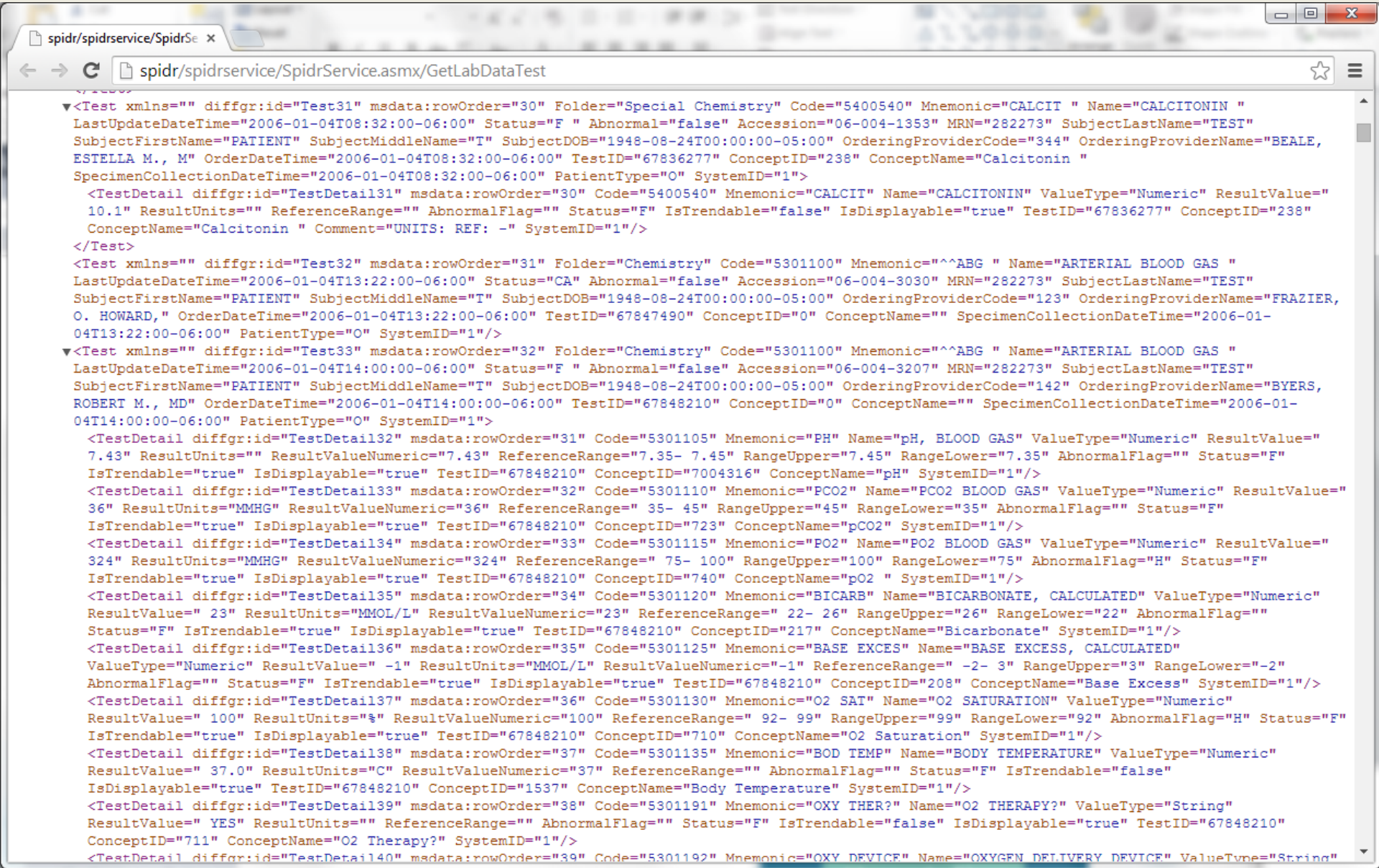
```
<?xml version="1.0"?>
<quiz>
 <qanda seq="1">
  <question>
   Who was the forty-second
   president of the U.S.A.?
  </question>
  <answer>
   William Jefferson Clinton
  </answer>
 </qanda>
 <!-- Note: We need to add
  more questions later.-->
</quiz>
```

http://en.wikipedia.org/wiki/File:XML.svg

XML details

- Tag
  - A markup construct that begins with < and ends with >.
    - *start-tags*; for example: <section>
    - *end-tags*; for example: </section>
    - *empty-element tags*; for example: <empty_section />
- Element
  - A logical document component which either begins with a start-tag and ends with a matching end-tag or consists only of an empty-element tag.
  - The characters between the start- and end-tags, if any, are the element's *content*, and may contain markup, including other elements, which are called *child elements*.
  - <Greeting>Hello, world.</Greeting>
- Attribute
  - A markup construct consisting of a name/value pair that exists within a start-tag or empty-element tag.
    - **<img** src="XML.jpg" alt="Example of XML structure" **/>**
    - Attributes are "src" and "alt"

XML is fully relational and capable of supporting arbitrarily complex data structures



Power of XML for data interchange comes from two sets of extensions

- XML Schema
  - Brings an enforcible “data contract” to XML
  - Defines the expected/required/optional elements of a particular class of XML documents
  - XML can be automatically verified for compliance by many frameworks/tools prior to application processing
- Web services using WSDL and SOAP

XML Schema

- A description of a type of XML document
  - Provides a data “contract” for implementation
  - Rules governing
    - Order
    - Data types
    - Required versus optional
    - Cardinality
    - Uniqueness
    - Referential integrity

XML Schema

Schema (prototype)

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Document (instance)

```
<?xml version="1.0"?>
<note
xmlns="http://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xsi:schemaLocation="http://www.w3schools.com
note.xsd">
  <to>Homer</to>
  <from>Marge</from>
  <heading>Reminder</heading>
  <body>Please remember to
 take out the garbage</body>
</note>
```

JSON

- Javascript serial object notation
- Open standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs
- Used primarily to transmit data between a server and web application, as an alternative to XML
- More parsimonious than XML (no closing tags)

```
{
    "firstName": "John",
    "lastName": "Smith",
    "isAlive": true,
    "age": 25,
    "height_cm": 167.64,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": "10021-3100"
    },
    "phoneNumbers": [
        { "type": "home", "number": "212 555-1234" },
        { "type": "fax",  "number": "646 555-4567" }
    ]
}
```

api PATHOLOGY INFORMATICS

Web services/WSDL

- Software implementations designed to support interoperable machine-to-machine interaction over a network
- Interfaces described in a machine-processable format (specifically WSDL)
- Permit use of complex functionality by downstream applications, with little development effort

Summary of additional data formats

- None of these solve the problem of complex data being complex, or needing to map between different applications which may have different relational views of the world
- Provide a toolset to approach problems with
- Successful solutions come from integrations of toolsets with domain expertise/feedback

Implementation

- Resources
  - http://HL7.org
    - HL7 organization
    - Standards available for review/download
  - Mirth Connect
    - An open source, highly capable messaging framework with deep HL7 support
    - http://www.mirthcorp.com/products/mirth-connect
  - Interfaceware
    - http://www.interfaceware.com/
    - Tutorials & information
    - Extensive (non-free) toolset for HL7

Top Pitfalls for Interfacing

- Semantic corruption
  - A major risk with conversions for anatomic pathology reports

HL7-based delivery of pathology reports converted from Microsoft Word to ASCII

Pathologist

Self, transcriptionist, resident entry

"Native" pathology report

Format conversion to ASCII text

DIAGNOSIS

Metastatic adenocarcinoma.

HL7

Interface engine

HL7

HIS Viewer

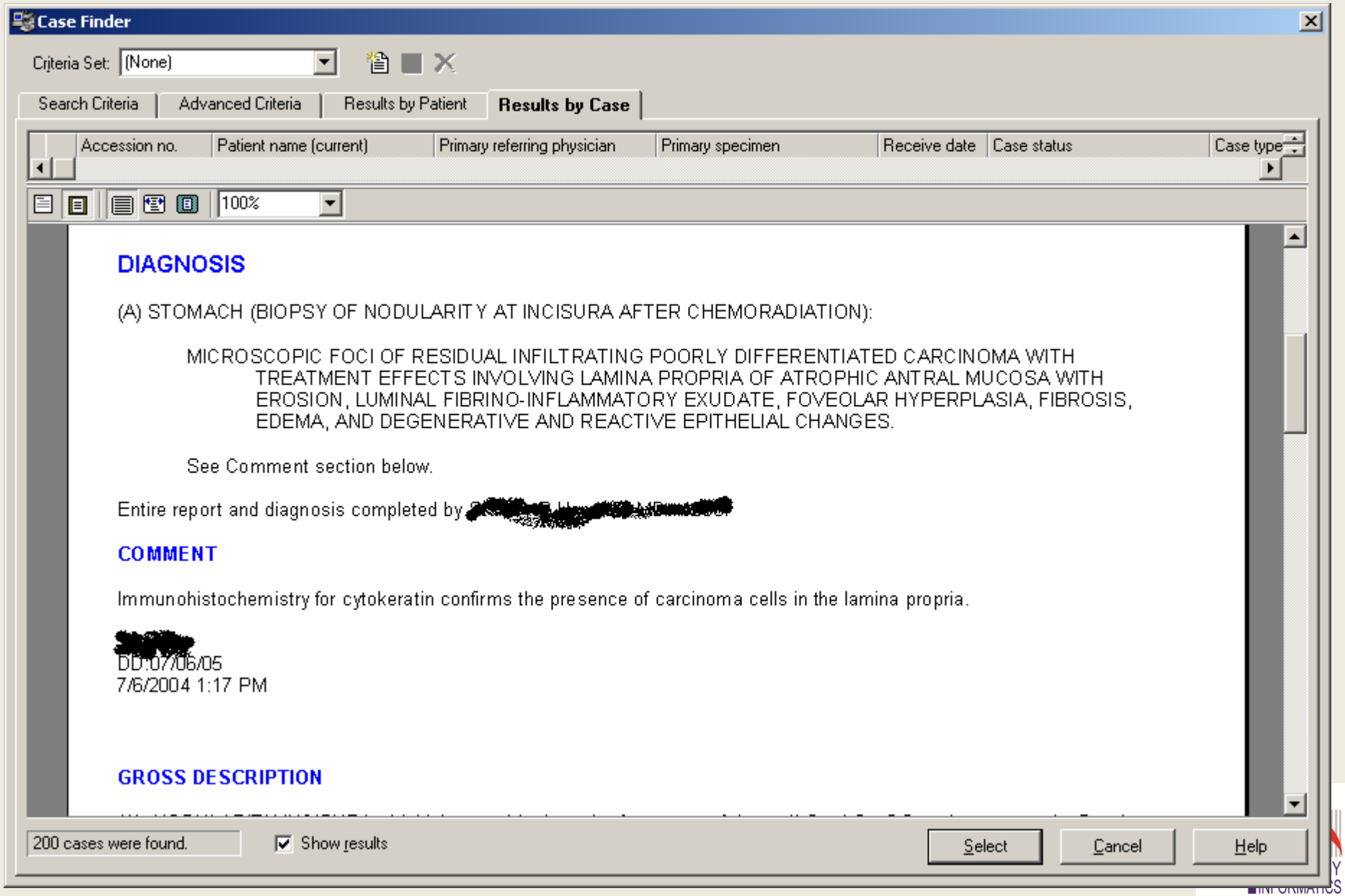
Clinician

Custom display logic

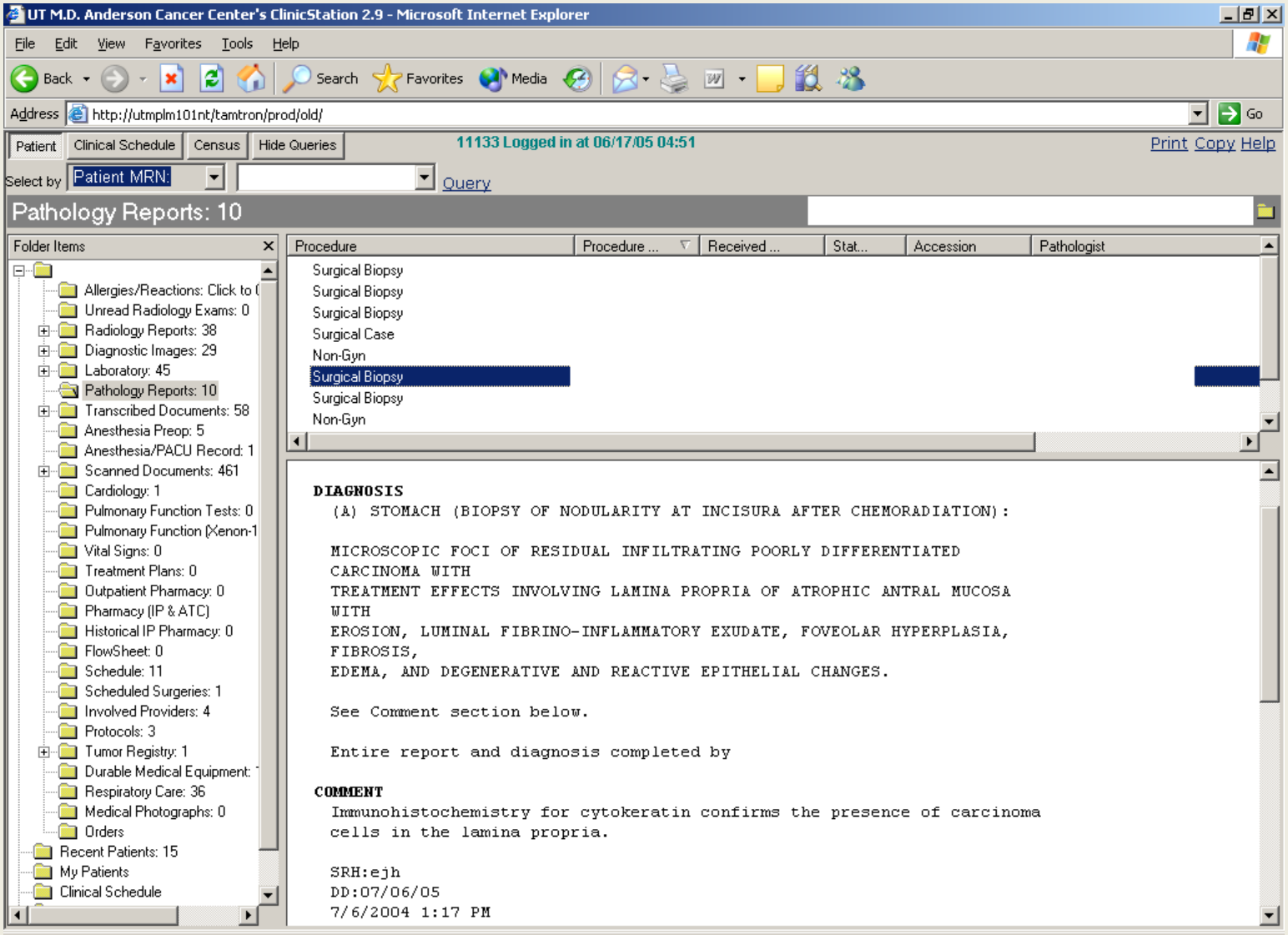
Some issues with this approach

- No “what you see is what you get”
- Advantages of HL7 standard (“standardization”) balanced with inflexibility
  - Every interface is custom
- Pathologists not directly involved in HIS viewer display logic
- One standard for quality reporting (of many!): Concordance between pathologist and clinician view of the report

Pathologist view



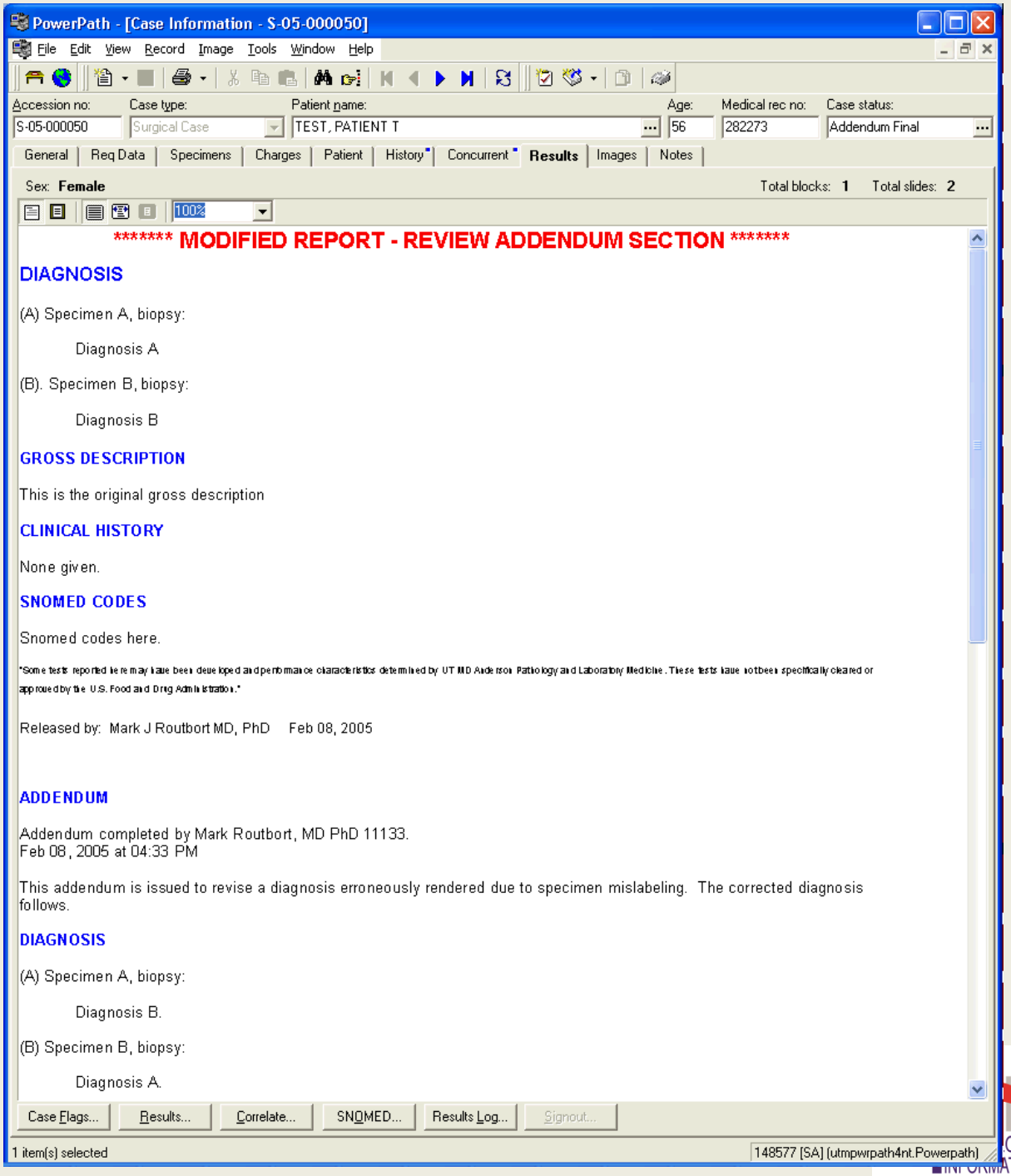
Clinician view in EMR



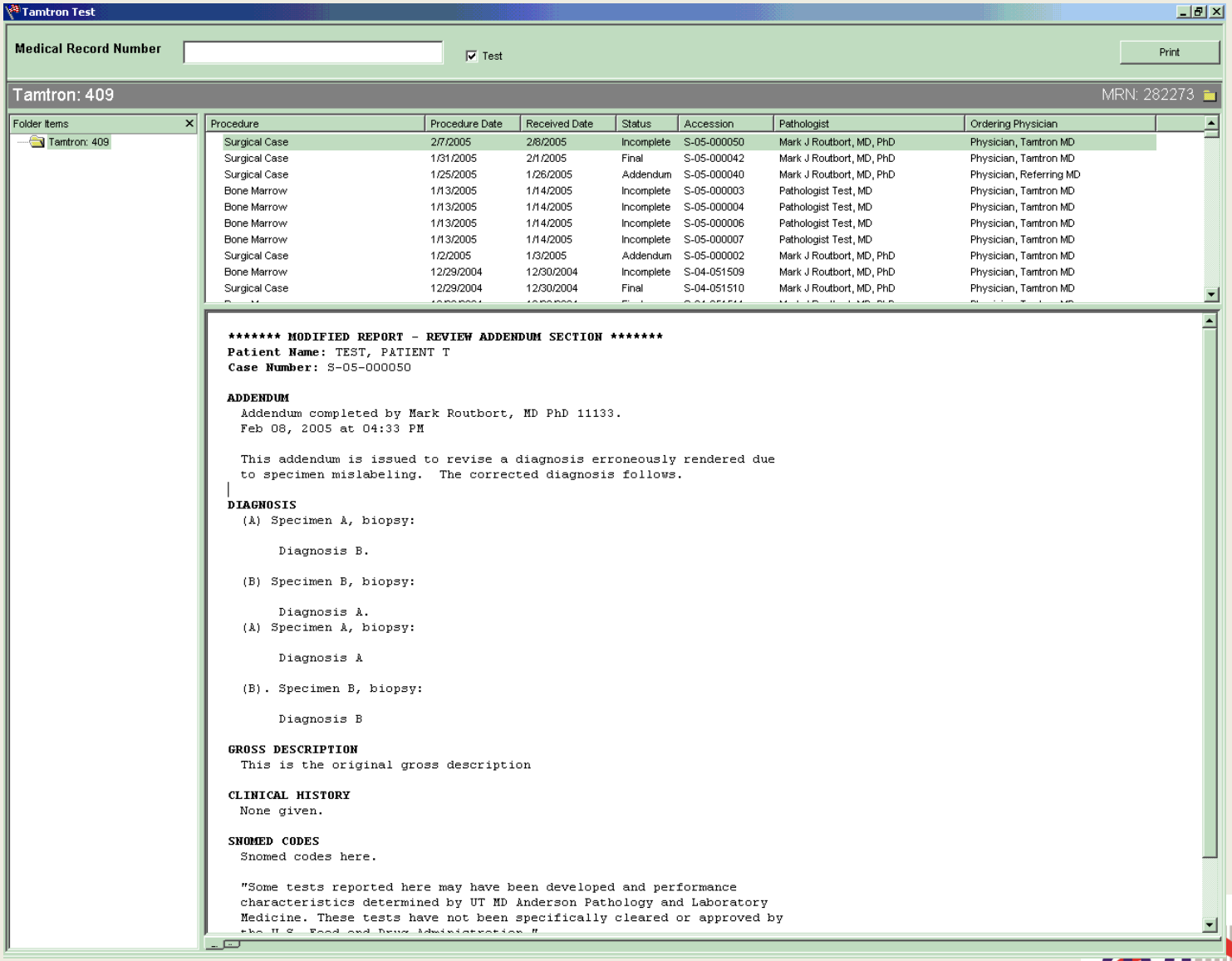
Issues noted in the ASCII conversion of formatted pathology reports included a necessary loss of font & style information, as well as variable line breaking which in some cases led to confusing reports

Integrity of semantic content is at stake in *any* transformation

In rare cases we saw the potential for semantic corruption:



Can you spot the problem?



“Direct” electronic delivery of pathology reports

Pathologist

Self, transcriptionist, resident entry

“Native” pathology report stored in PowerPath database

Web service based direct query for report Rich Text Format (RTF)

HIS Viewer

Custom path report viewing control

Clinician

Top Pitfalls for Interfacing

- Allowing multiple views of reality
  - For instance, enabling an ADT feed for patient demographics but also also manual patient entry for outreach
  - Multiple systems will always get out of synch – particularly if some allow manual data entry
  - Patient merges tend to be a band-aid solution for systemic problems

Top Pitfalls for Interfacing

- Vendor promises
  - "Sure we can do that"
  - "We have an interface/flat file transfer system/API that can solve that issue"
  - Any promises should be
    - Taken with a grain of salt
    - Detailed in the contract (pre-nup)
    - Verified with references where possible
    - Verified with real world demonstration where possible

Top Pitfalls for Interfacing

- Square pegs and round holes
  - Poor mapping between the relational database design or workflow requirements between two different systems can't be solved independently with interfacing technologies
    - Example. System A uses an 8 character mnemonic/code for lab tests. System B is not constrained.
  - Mapping issues require good old fashioned common sense, workflow/impact analysis, and harmonization between systems

Summary

- There are many tools for LIS interfacing
- If you have a hammer, everything looks like a nail
- The specific tool is less important than
  - having a general understanding of the toolkit
  - asking questions and learning
  - seeking to merge domain expertise with technical implementation work