Module 2: XML Representation

- Concepts
- Parsing and Validation
- Schemas
What is Metadata?

Literally, data about data

- Description of data that captures some useful property regarding its
  - Structure and meaning
  - Provenance: origins
  - Treatment as permitted or allowed: storage, representation, processing, presentation, or sharing

- Markup is metadata pertaining to media artifacts (documents, images), generally specified for suitable parsable units
MONOLITHIC

UI

REASONING

DATA

UI

R

D

D

Y

R

Y

R

Y

R

Payroll:

Benefits:
Motivations for Metadata

Mediating information structure (surrogate for meaning) over time and space

- Storage: extend life of information
- Interoperation for business
- Interoperation (and storage) for regulatory reasons

General themes
- Make meaning of information explicit
- Enable reuse across applications: repurposing (compare to screen-scraping)
- Enable better tools to improve productivity

Reduce need for detailed prior agreements
How much prior agreement do you need?

- No markup: significant prior agreement
- Comma Separated Values (CSV): no nesting
- Ad hoc tags: \[ st = \text{vivek}, gr = \text{A}, \ldots \] 
- SGML (Standard Generalized Markup Language): complex, few reliable tools; used for document management
- HTML (HyperText Markup Language): simplistic, fixed, unprincipled vocabulary that mixes structure and display
- XML (eXtensible Markup Language): simple, yet extensible subset of SGML to capture custom vocabularies
  - Machine processible
  - Comprehensible to people: easier debugging
Uses of XML

Supporting arms-length relationships

- Exchanging information across software components, even within an administrative domain
- Storing information in nonproprietary format
- Representing semistructured descriptions:
  - Products, services, catalogs
  - Contracts
  - Queries, requests, invocations, responses (as in SOAP): basis for Web services

configured
Example XML Document

```
1  <?xml version="1.0"?>  <!-- processing instruction -->  
   <topelem attr0="foo.">  <!-- exactly one root -->  
   <subelem attr1="v1" attr2="v2">  
     Optional text (PCDATA)  <!-- parsed character data -->  
     <subsubelem attr1="v1" attr2="v2"/>  
   </subelem>  
2   </topelem>  
   <null_elem/>  
3   <short_elem attr3="v3"/>  
4   </topelem>  
```

© Munindar P. Singh. CSC 513, Spring 2010
1. **STRUCTURE**
2. **WAY TO STANDARDIZE**

Each, a custom vocabulary

### Table Structure

<table>
<thead>
<tr>
<th>Address</th>
<th></th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>unit price</td>
<td>qty</td>
</tr>
</tbody>
</table>

### Additional Details
- discount
- fax
- shipping

### Groups
- type
- advance
- due
(Reminder)

(from)

to

(amount due)

since

penalty

(name)

(address)

currency

amt

num st

tom

zip

USD

100

30

f

amt

5

5.00

(<Reminder>)

(<From>)

(From)

(To)

(name)

(name)

(name)

(address)

(address)

(address)

(1707)

(<Reminder>)
Reminder

/ \  
To  address . . . zip

( Reminder
  )
( To )
( address )
( zip )
) Reminder
Exercise

Produce an example XML document corresponding to

- An invoice from Locke Brothers for 100 units of door locks at $19.95, each ordered on 15 January and delivered to Custom Home Builders
- Factor in certified delivery via UPS for $200.00 on 18 January
- Factor in addresses and contact info for each party
- Factor in late payments
invoice

From To Item Order

Unit Price Date

Invoice

Item From To OrderDate

Qty Price Name

Invoice

From Unit Item Date To

Number Price Qty Price

Date should be under item

Qty "not"
NESTING: ONLY STRUCTURING PRIMITIVE

PATTERNS: PARENT-CHILD SIBLINGS