Validating

Verifying whether a document matches a given grammar (assumes well-formedness)

- Applications have an explicit or implicit syntax (i.e., grammar) for their particular elements and attributes
  - Explicit is better have definitions
  - Best to refer to definitions in separate documents

- When docs are produced by external software components or by human intervention, they should be validated
NOTION OF VALIDITY

MESSAGES IN X.M

INDEPENDENT OF IMPLEMENTATION

OTS TOOLS TO VALIDATE

(VERIFY VALIDITY)
How to conceptualize the information:
- No sublem text in student
- Nulls?
- Variations

Requirements for the language in which to specify the grammar of valid docs:
- Legal attribute values
- May, min

Criteria by which to judge specific docs:
- Root element
  - Elem: Course with attributes (such as CID)
  - Subelem: Student
- One copy of course with many copies of students
- Student may have zero or more subelems
- Uniqueness of students

Intended type of doc:
- Course
- Student
- Course
Specifying Document Grammars

Verifying whether a document matches a given grammar

- Implicitly in the application (in a program)
  - Worst possible solution, because it is difficult to develop and maintain

- Explicit in a formal document; languages include
  - Document Type Definition (DTD): in essence obsolete
  - XML Schema: good and prevalent
  - Relax NG: (supposedly) better but not as prevalent
**Desired Form**

```xml
<course sid='1'>
  <student sid='1'>
    ...
  </student>
</course>
```

**Desired Tree**

```
course
  /
  student sid
    sid
      ... 
```

```
<student sid='1'>
  <course cid='1'>
    ...
  </course>
</student>
```

Well formed

But invalid w.r.t. above intention
XML Schema

- Same syntax as regular XML documents
- Local scoping of subelement names
- Incorporates namespaces
- (Data) Types
  - Primitive (built-in): string, integer, float, date, ID (key), IDREF (foreign key), ...
  - simpleType constructors: list, union
  - Restrictions: intervals, lengths, enumerations, regex patterns,
  - Flexible ordering of elements
  - Key and referential integrity constraints