Frame Systems

CPSC 433: Artificial Intelligence
Fall 2024

Jonathan Hudson, Ph.D.
Assistant Professor (Teaching)
Department of Computer Science
University of Calgary

August 8, 2024

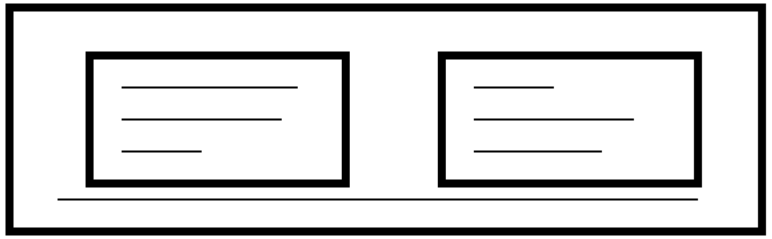
Frame Systems

Frames

- Slot-and-filler mechanism
- Conditions on filling objects for a slot possible
- Filler can be another frame
- Extend record concept with associated functionality (procedural knowledge)

☞ Predecessor/special case/ more general concept of object-oriented programming

☞ Conditions and procedural knowledge define semantics



XML

XML

- eXtensible Markup Language
- Subset of SGML
- Originally a method for putting structured data in a text file
- Allows to define own terms and markup
  ☞ allows to convey knowledge
- One of the key elements of the Semantic Web
  (together with Ontologies)

Basic data structures

- tags enclose text
  `<Address> 2500 University Drive NW </Address>`
- tags can be nested

```
<Address>
        <number> 2500 </number>
        <street> University Drive NW </street>
</Address>
```

- Tags may have attributes
  `<Address type=“North-America”> 2500 University Drive NW </Address>`

Semantics

- DTD to validate XML expressions
  (or XML Schema, Xlink and Xpointer, …)
- Ontologies to describe meaning of tags
  - Based on concensus between parties on human level
  - Provided to computer by procedures that work on tags

DTD - Document Type Definition (I)

- Part of XML file or described in own file
- Describes logical document structure
- <!ELEMENT name (#PCDATA)>
  ☞ defines tags <name> and </name> and content
  between tags has to be parsable character data text
- <!ELEMENT Diet (breakfast,lunch)>
  <!ELEMENT breakfast (#PCDATA)>
  <!ELEMENT lunch (#PCDATA)>
  ☞ Diet consists of entries for breakfast and lunch (in this order)

DTD - Document Type Definition (II)

- <!ELEMENT name (#PCDATA)>
- <!ATTLIST name gender (male|female) #REQUIRED>
- ☞ defines attributes for tags
- plus much more syntax

- (list of options separated by |)
- CDATA for character data
- #REQUIRED -> required
- #IMPLIED -> optional

Ontology

UNIVERSITY OF CALGARY

Ontology

- File or document that defines relations among terms
- Typically: taxonomy + set of inference rules
- Formal description mechanism: a modal logic
- Practical use:
  - Taxonomy = DTD file (or other validation scheme)
  - Inference rules = procedures that use elements to produce other elements

☞Same concept can be expressed by different ontologies

☞Same taxonomy can have different inference rules and therefore different semantics

☞Still lots of research necessary (and coming up with norms)

How to get knowledge into the representation structure

- With ontology:
  state your facts in a file using the provided tags
- Without ontology:
  - Define tags and a DTD for it
  - Provide procedures using tags
  - See above

Discussion

- Uses the internet
- Rather pragmatical
- Meta concept, very general
- Easy to read and understand by humans
- Lots of tools and libraries already available

– Semantics via ontologies dangerous:
  there are many of them for a subject area and Microsoft-like behavior of the humans involved has to be expected
  ☞ semantic standards for subject areas needed!

And what about processing data?

- With ontology:
  run procedures that are provided
  ☞ similar to PROLOG (hopefully less problematic with regard to having to know about control)

- Without ontology or if missing certain functionality:
  write procedure for functionality and run it
  ☞ often involves searching through knowledge base

XML Examples

Examples

Model a knowledge base for the items in a warehouse. An item is either in stock or not, it has a name, a price, a manufacturer and a location. The location consists of a row number and a shelf number and optionally a box.

Examples

Model a knowledge base for the **items** in a **warehouse**. An **item** is either in stock or not, it has a name, a price, a manufacturer and a **location**. The **location** consists of a row number and a shelf number and optionally a box.

Things -> **warehouse**, **item**, **location**

Examples

Model a knowledge base for the items in a warehouse. An item is either in stock or not, it has a name, a price, a manufacturer and a location. The location consists of a row number and a shelf number and optionally a box.

Things -> **warehouse**, **item**, **location**

```
<Warehouse>
  <Item>
        <Location></Location>
  </Item>
</Warehouse>
```

Examples

Model a knowledge base for the items in a warehouse. An item is either in stock or not, it has a **name**, a **price**, a **manufacturer** and a **location**. The location consists of a row number and a shelf number and optionally a box.

Things -> warehouse, item, location,box

<Warehouse>

  <Item **instock**="true", **name**="item_name", **price**="5", **manufacturer**="ucalgary">

      <**Location**/>

  </Item>

</Warehouse>

Examples

Model a knowledge base for the items in a warehouse. An item is either in stock or not, it has a name, a price, a manufacturer and a location. The **location** consists of a **row** number and a **shelf** number and optionally a **box**.

Things -> warehouse, item, location,box

```
<Warehouse>
  <Item instock=“true”, name=“item_name”, price=“5”, manufacturer=“ucalgary”>
        <Location row=“1”, shelf=“2” box=“3”/>
  </Item>
</Warehouse>
```

Examples

Model a knowledge base for the items in a warehouse. An item is either in stock or not, it has a name, a price, a manufacturer and a location. The location consists of a row number and a shelf number and optionally a box.

Things -> warehouse, item, location,box

```
<?xml version=“1.0” ?>
<Warehouse>
   <Item instock=“true”, name=“item_name”, price=“5”, manufacturer=“ucalgary”>
            <Location row=“1”, shelf=“2” box=“3”/>
   </Item>
</Warehouse>
```

Examples

Model a knowledge base for the items in a warehouse. An item is **either in stock or not**, it has a name, a price, a manufacturer and a location. The location consists of a row number and a shelf number and **optionally** a box.

```
<?xml version=”1.0” ?>
<!DOCTYPE Warehouse[
        <!ELEMENT Warehouse (Item)>
        <!ELEMENT Item (Location)>
        <!ELEMENT Location EMPTY>
        <!ATTLIST Item instock (true|false) #REQUIRED>
        <!ATTLIST Item name CDATA #REQUIRED>
        <!ATTLIST Item price CDATA #REQUIRED>
        <!ATTLIST Item manufacturer CDATA #REQUIRED>
        <!ATTLIST Location row CDATA #REQUIRED>
        <!ATTLIST Location shelf CDATA #REQUIRED>
        <!ATTLIST Location box CDATA #IMPLIED>
]>
```

```
<?xml version=“1.0” ?>
<Warehouse>
<Item instock=“T/F”, name=“”, price=“”, manufacturer=“”>
<Location row=“”, shelf=“” box=“opt”/>
</Item>
</Warehouse>
```

Examples

Model a knowledge base for the items in a warehouse. An item is **either in stock or not**, it has a name, a price, a manufacturer and a location. The location consists of a row number and a shelf number and **optionally** a box.

```
<?xml version=”1.0” ?>

<!DOCTYPE Warehouse[
        <!ELEMENT Warehouse (Item)>
        <!ELEMENT Item (Location)>
        <!ELEMENT Location EMPTY>
        <!ATTLIST Item instock (true|fals
        <!ATTLIST Item name CDATA #REQ
        <!ATTLIST Item price CDATA #REQ
        <!ATTLIST Item manufacturer CDATA #REQUIRED>
        <!ATTLIST Location row CDATA #REQUIRED>
        <!ATTLIST Location shelf CDATA #REQUIRED>
        <!ATTLIST Location box CDATA #IMPLIED>
]>
```

NOT THE ONLY ANSWER!!!

```
<?xml version=“1.0” ?>
<Warehouse>
<Item instock=“T/F”, name=“”, price=“”, manufacturer=“”>
      w=“”, shelf=“” box=“opt”/>

      e>
```

Onward to …
AI Summary

Jonathan Hudson, Ph.D.
jwhudson@ucalgary.ca
https://cspages.ucalgary.ca/~jwhudson/

UNIVERSITY OF CALGARY