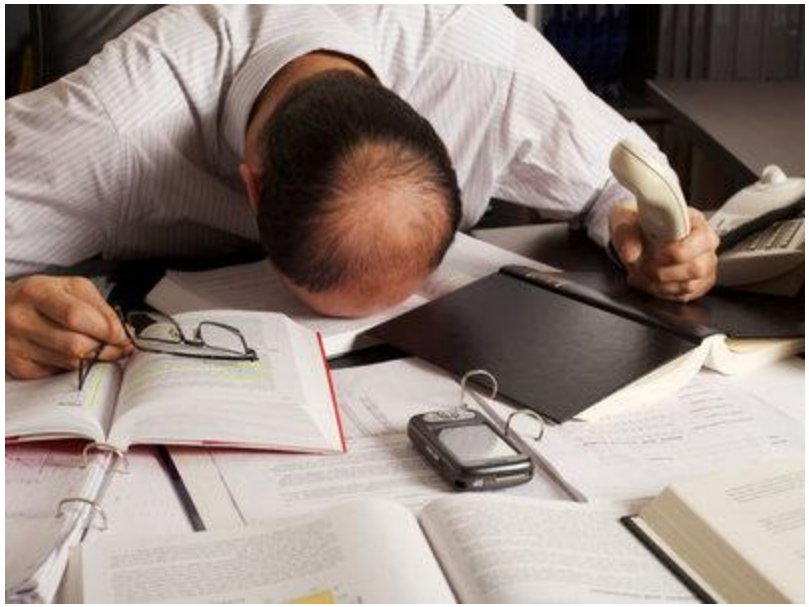
Digital Overload—and What to Do about It

Jim Larson
Co-program Chair
SpeechTEK Conference
May 23-25, 2016, Washington, DC

Digital Overload

- We consume 3x more information as in 1960.
  - University of California, San Diego
- The average computer user checks 40 websites a day
  - New York Times



The Big Picture

Query
via speaking
or typing

Data
management

Browse
results

How We Access Data

- Search
  - Query
    - User knows what information he/she needs
    - Formulates a request
  - Browse
    - User will recognize desired information when he/she sees it
    - Selects an area likely to contain results and examines content

May be spoken or typed

Paging, scrolling, or zooming using a screen

Digital Overload—and What to Do about It

- Look at disciplines that solve the digital overload problem
  - Database management systems
  - Library systems
  - Recipe collections
  - Sheet music collections
  - Etc.

How We Access Data

Organize

Query

Evaluate

Browse

How We Access Data

Organize

Query

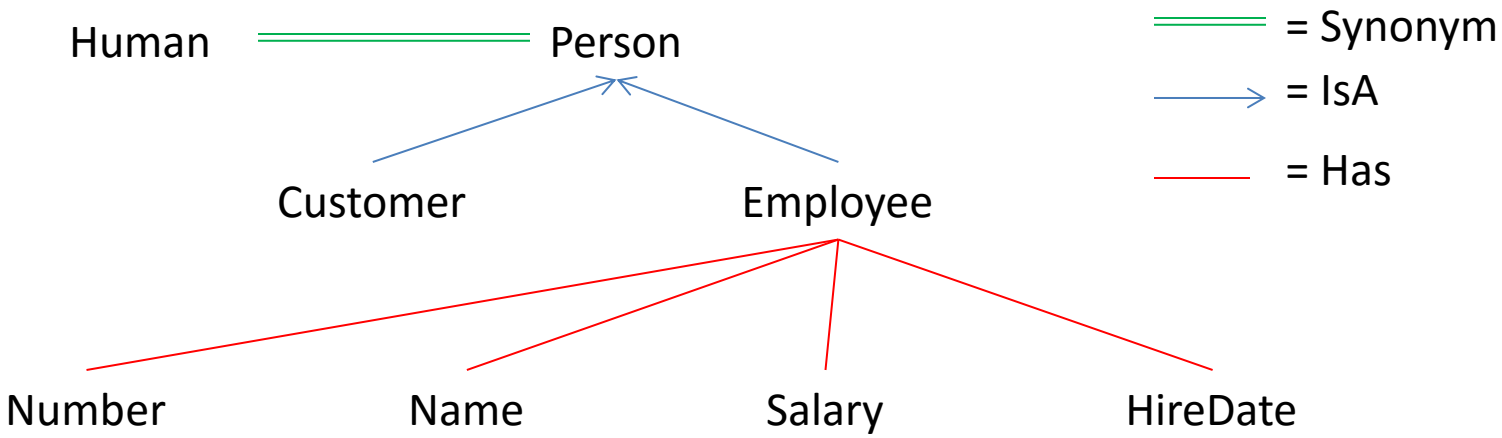
Evaluate

Browse

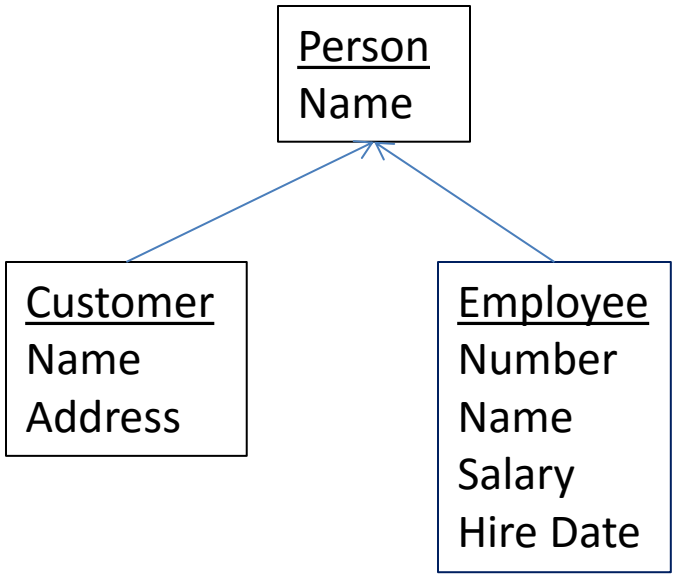
- Create ontology

Example Ontology

- Concept names for real-life things
- Relationships among concepts

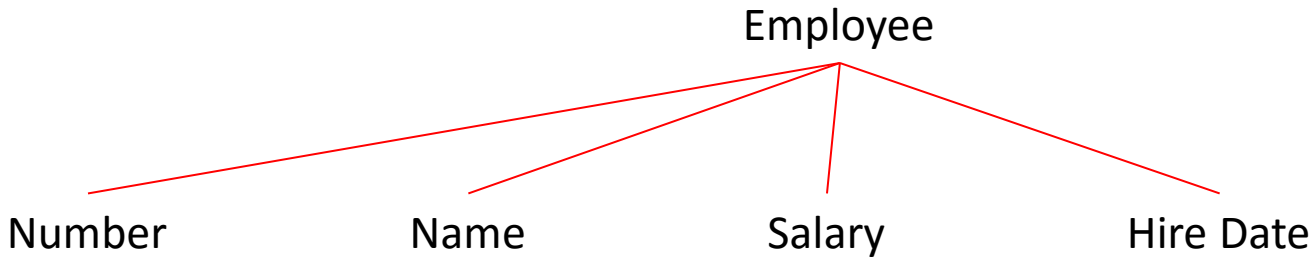


Another Ontology Format: Entity-Relationship Diagrams



Why Use an Ontology?

- Use ontology terms to describe data

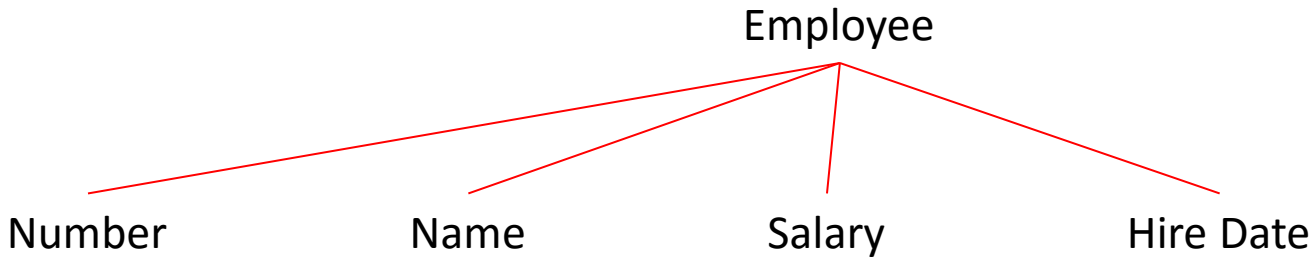


Employee

Number	Name	Salary	Hire Date
5	Able	48,000	6-18-2013
7	Baker	55,000	8-20-2014
19	Carson	88,000	5-03-2012
…			

Why Use an Ontology?

- Use ontology terms to describe data




```
<employee>
        <number> 5 </number>
        <name> Able </name>
        <salary> 48000 </salary>
        <hire date> 6-18-2013 </hire date>
 <employee> Able </employee>
```

How We Access Data

Organize

Query

Evaluate

Browse

- Create ontology
- Create metadata

Metadata

- Metadata is data about data
  - Card catalog in library
  - Table of contents and index in book
  - Names of columns in spreadsheet
  - Tags in XML description
- Metadata uses terms from an ontology to describe data

Metadata Describes Data

Employee

Number	Name	Salary	Hire Date
5	Able	48,000	6-18-2013
7	Baker	55,000	8-20-2014
19	Carson	88,000	5-03-2012
…			

```
<employee>
        <number> 5 </number>
        <name> Able </name>
        <salary> 48000 </salary>
        <hire date> 6-18-2013 </hire date>
 <employee> Able </employee>
```

Self-describing data

How to Create Metadata?

- Manually
  - Tagging data fragments is tedious, time consuming, and expensive
- Clever AI algorithms
  - AI learning systems can be trained to tag data

How We Access Data

Organize

Query

Evaluate

Browse

- Select data set

Identify Source

- Index of indexes
- Cross indexes
- Query router

How We Access Data

Organize

Query

Evaluate

Browse

- Select data set
- Formulate request using metadata terms

Query Formats

- Use ontology terms to formulate queries and commands

SQL

```
select name
from employee
where salary > 50,000;
```

Query By Example

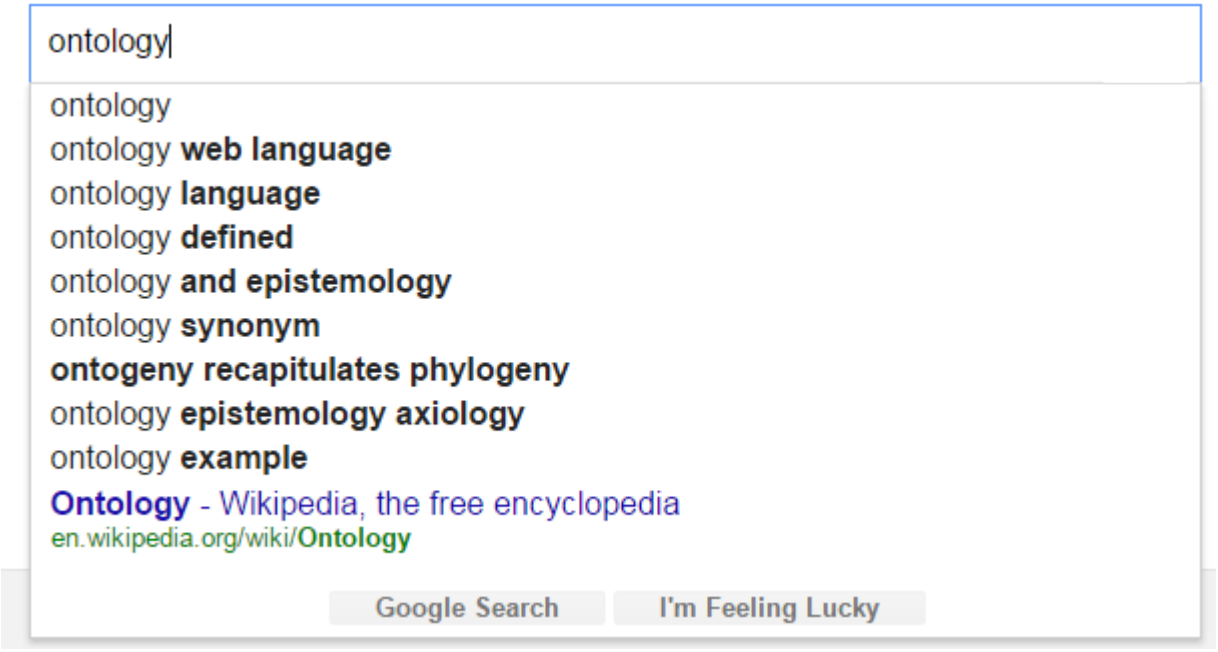
Number	Name	Salary	Hire Date
	P.	>50000	

Natural Language

Get employees with salary greater than 50,000

May be spoken or typed

Review and Modify Related Queries



How We Access Data

Organize

Query

Evaluate

Browse

- Apply to sample data
- Examine content

Compare Result to Small Data Set

Employee

Number	Name	Salary	Hire Date
5	Able	48,000	6-18-2013
7	Baker	55,000	8-20-2014
19	Carson	88,000	5-03-2012
…			

Result

Name
Baker
Carson

How We Access Data

Organize

Query

Evaluate

Browse

- Apply to sample data
- Examine content
- Refine request

Incorporate Result into Another Request

AddressBook

Name	City	State	Zip
Able	New York City	NY	00503
Baker	Portland	OR	97006
Carson	St Paul	MN	23456
…			

Result

Name
Baker
Carson

Select city
From AddressBook, Result
Where Result.Name = AddressBook.Name;

May be spoken or typed

How We Access Data

Organize

Query

Evaluate

Browse

- Extract intent

Extract Intent Using VoiceXML Grammar Rules

- Example grammar rule with Script Syntax:

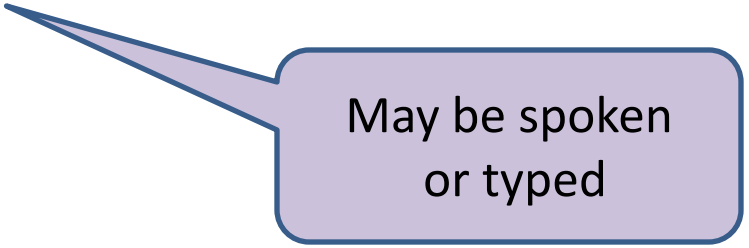
```
<rule id = "action">
   <one-of>
      <item> small <tag> out.size = "small"; </tag> </item>
      <item> medium <tag> out.size = "medium"; </tag> </item>
      <item> large <tag> out.size = "large"; </tag> </item>
   </one-of>
   <one-of>
      <item> green <tag> out.color = "green"; </tag> </item>
      <item> blue   <tag> out.color = "blue"; </tag>  </item>
      <item> white <tag> out.color = "white"; </tag>  </item>
   </one-of>
</rule>
```

- ECMAScript structure:

```
action: {
     size: "large"
     color:  "white"
     }
```

Extract Intent Using Wit.ai

- Name the intent
  - Example: temperature_get
- Provide several expressions for how users request the intent
  - What is the temperature?
  - How hot is it?
  - How cold is it?
  - Can you tell me the temperature?
- Map user request to an intent
  - “What’s the temperature” -> temperature_get



How We Access Data

Organize

Query

Evaluate

Browse

- Display
- Page
- Scroll
- Zoom
- Sort an manipulate
- Etc.

Larson's Manifesto

1. Data suppliers
   - Use standard ontologies
   - Create self-describing data
   - Use standard metadata formats

- List of ontologies
  - https://www.w3.org/wiki/Lists_of_ontologies
- List of XML languages
  - https://en.wikipedia.org/wiki/List_of_XML_markup_languages
- List of metadata standards
  - https://en.wikipedia.org/wiki/Metadata_standard

Larson's Manifesto

2. System developers enable users to:
   - Use aids for formulating queries
   - Review and modify other people's queries
   - Review result, resolve ambiguities
   - Iteratively refine query

Questions