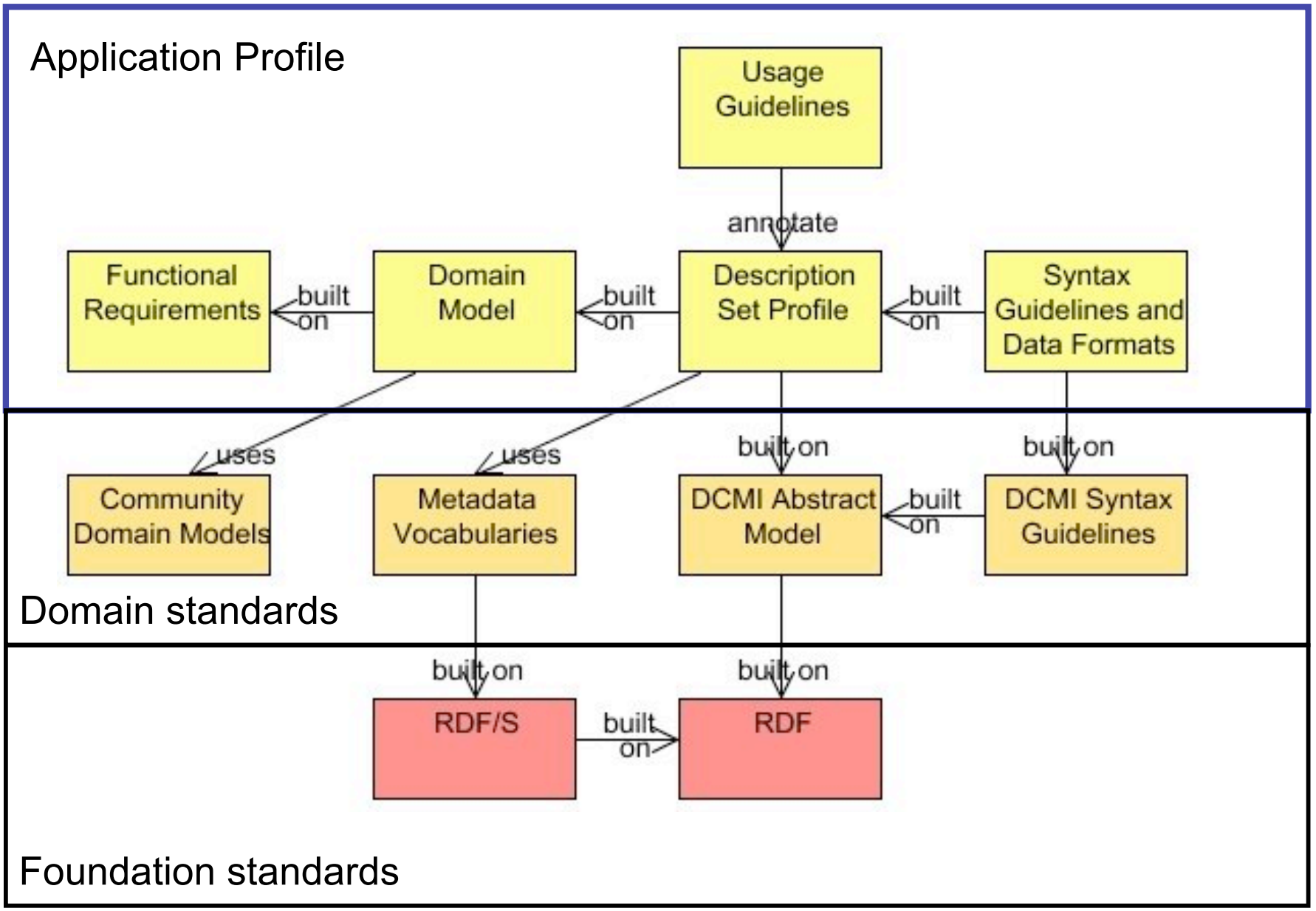
Metadata Engineering Methodology

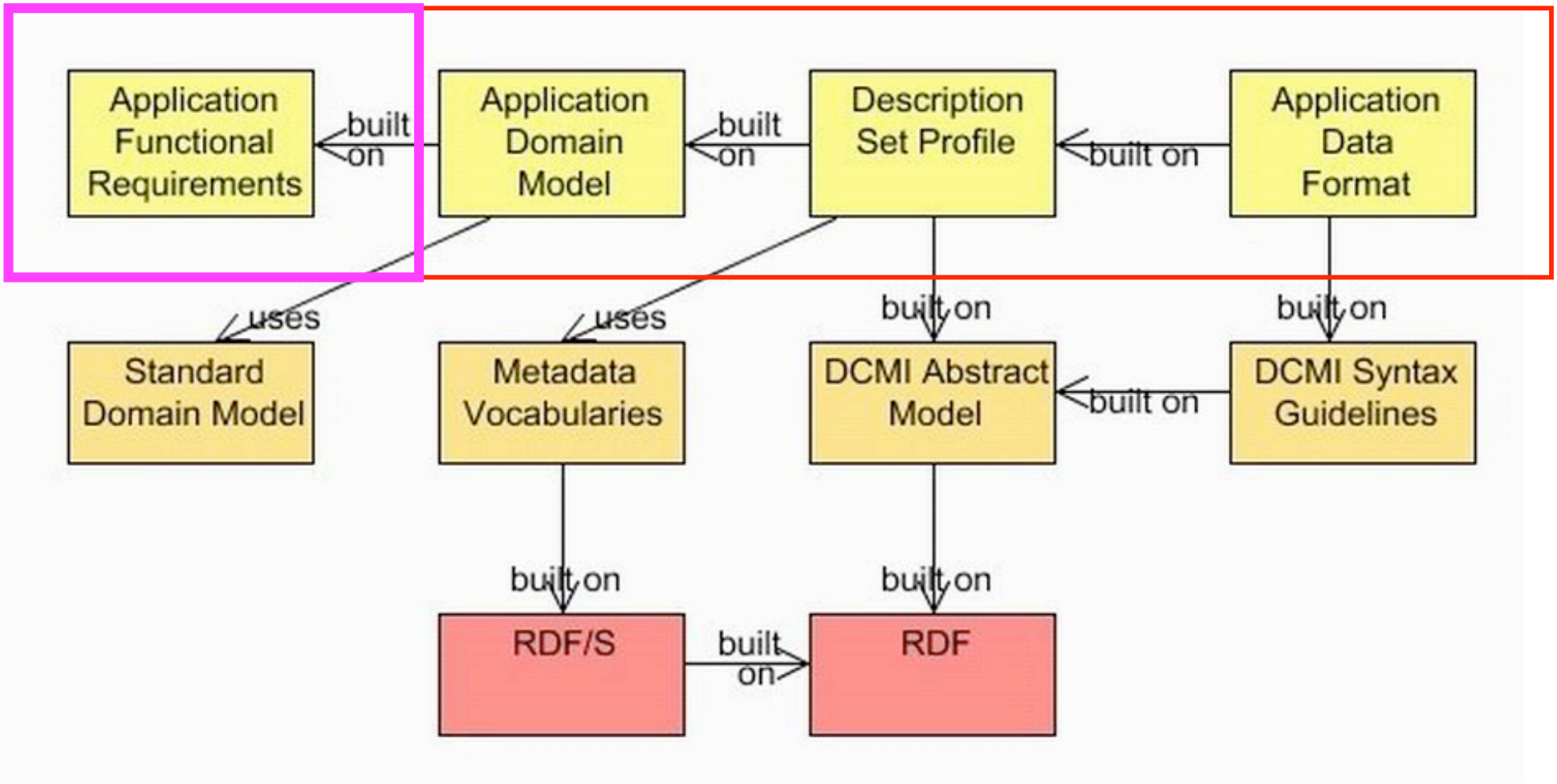
Tom Baker

MPG EScience Seminar on Metadata Infrastructures

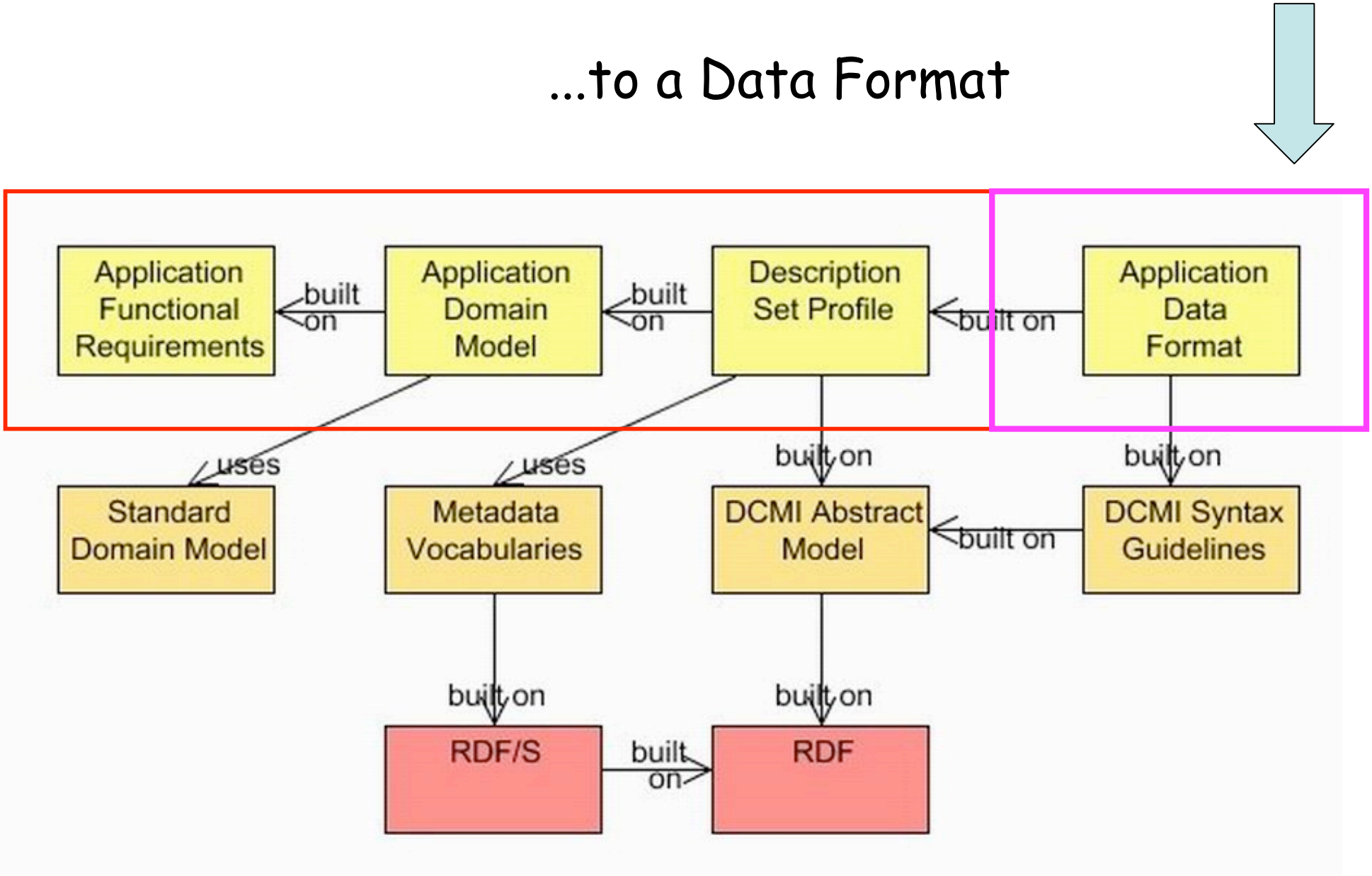
Berlin, 14 October 2008


Application Profile
Usage Guidelines
annotate
Functional Requirements
built on
Domain Model
built on
Description Set Profile
built on
Syntax Guidelines and Data Formats
uses
uses
built on
built on
Community Domain Models
Metadata Vocabularies
DCMI Abstract Model
built on
DCMI Syntax Guidelines
Domain standards
built on
built on
RDF/S
built on
RDF
Foundation standards

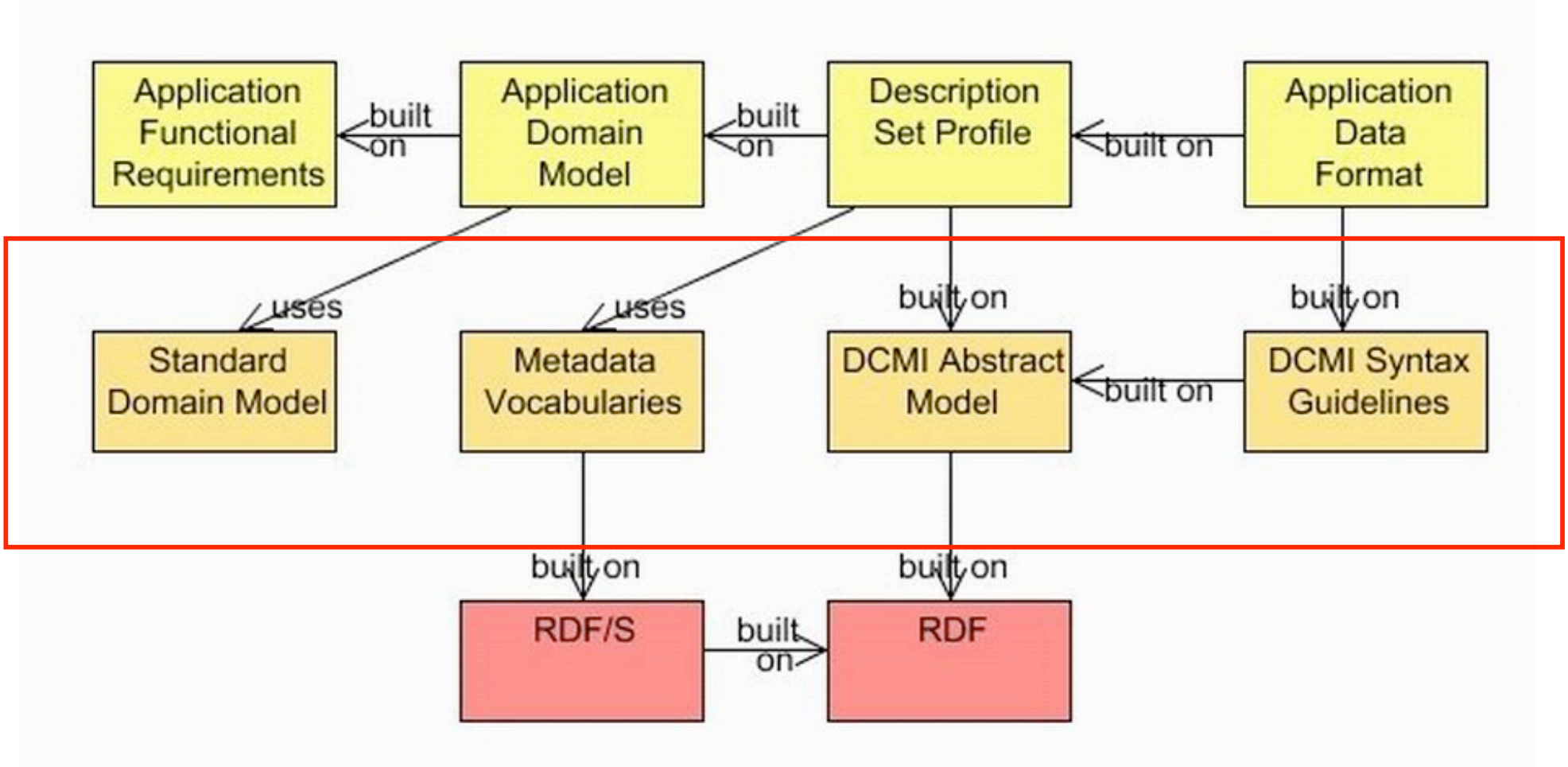
Getting from Requirements...



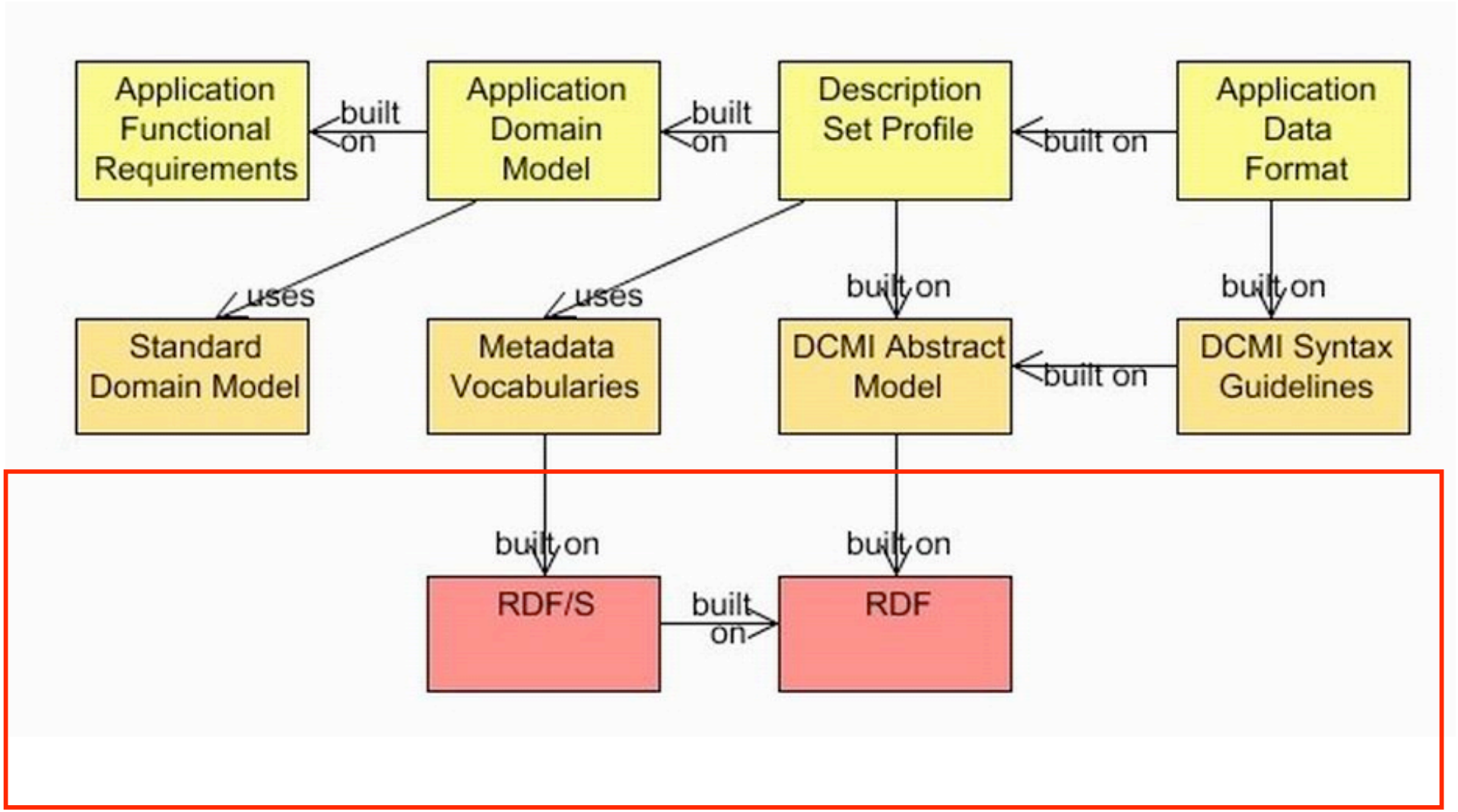
...to a Data Format



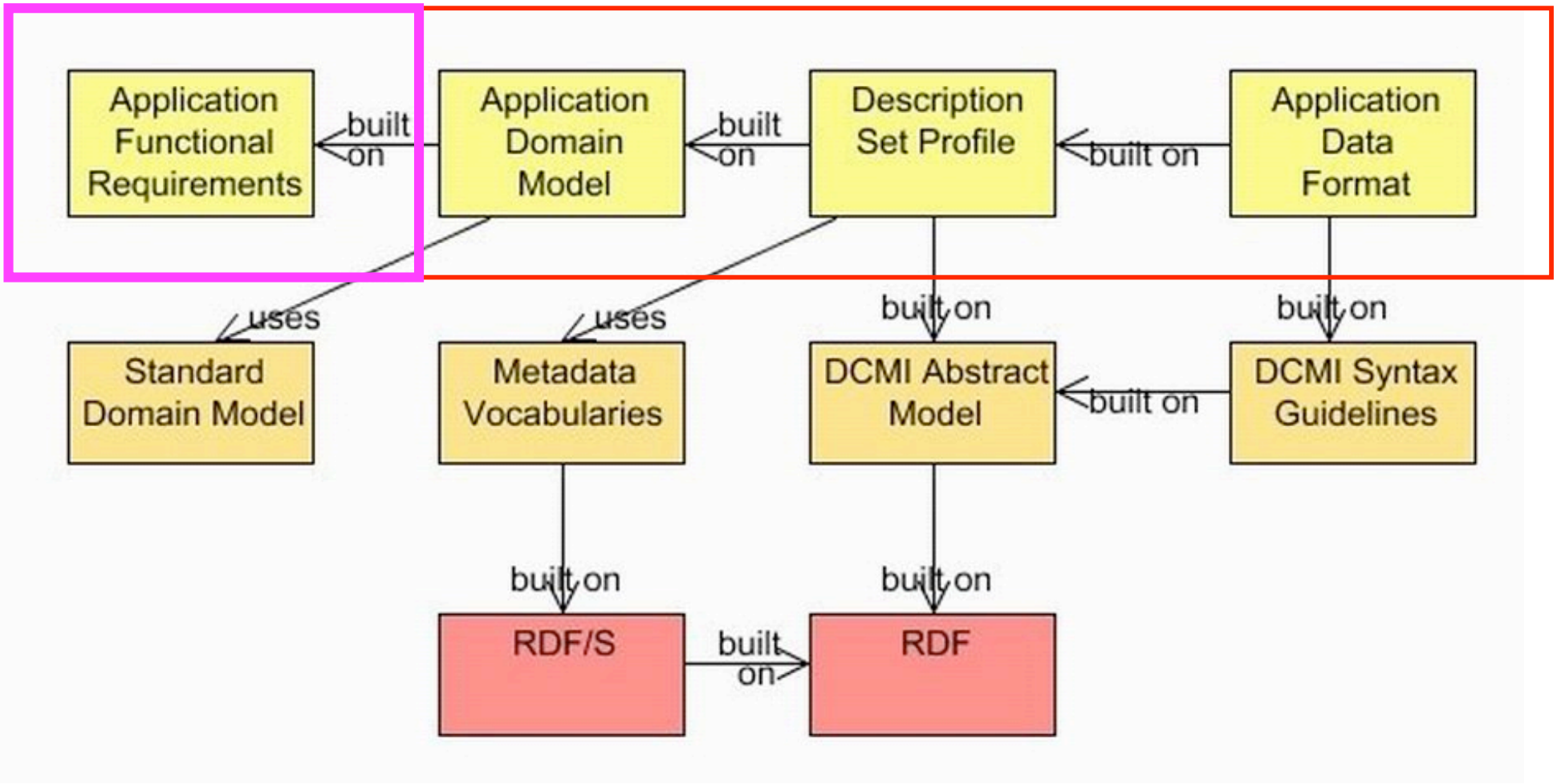
...based on Community Standards



...on the basis of Foundation Standards (RDF)



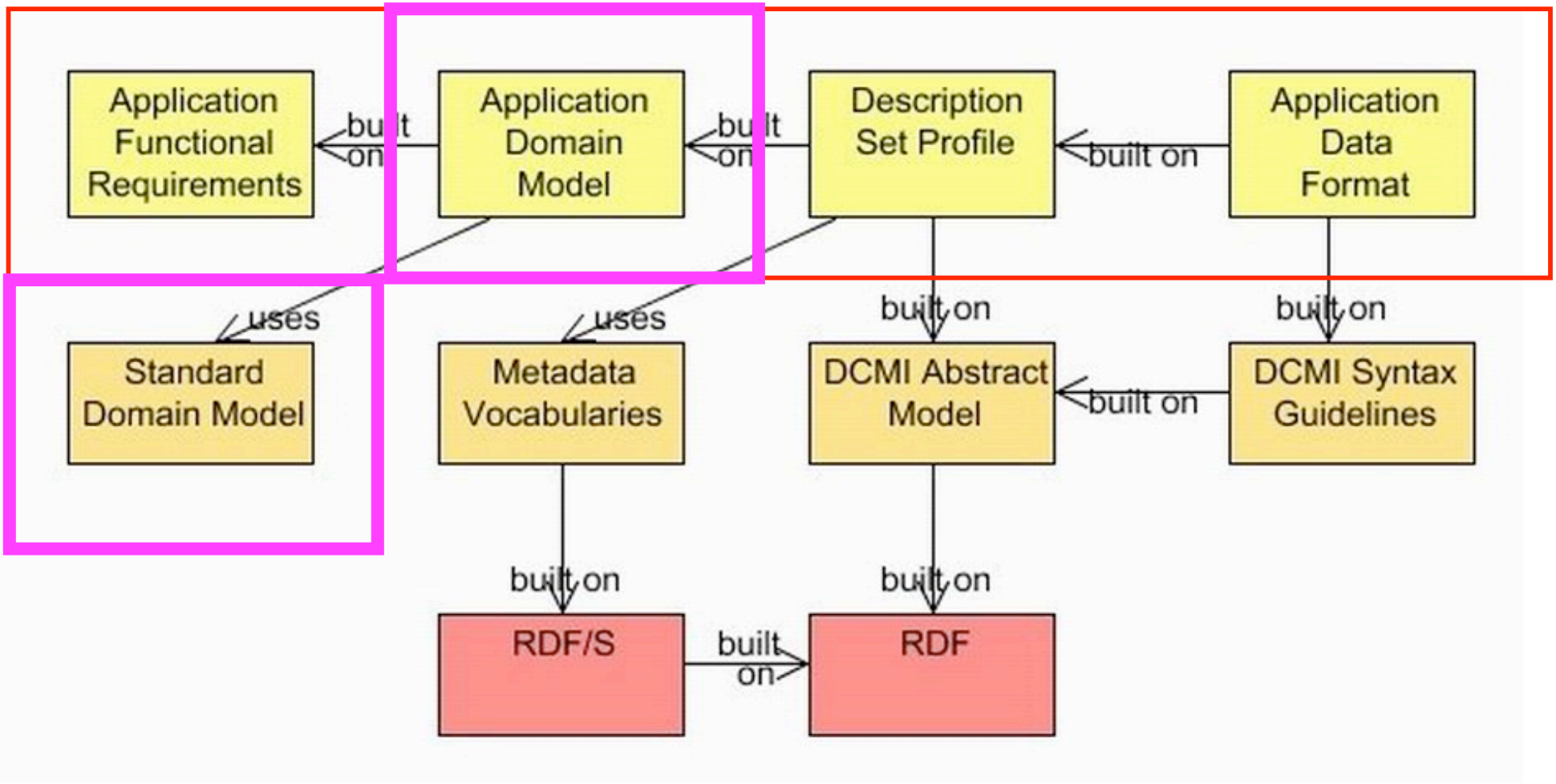
What does your application need to do?



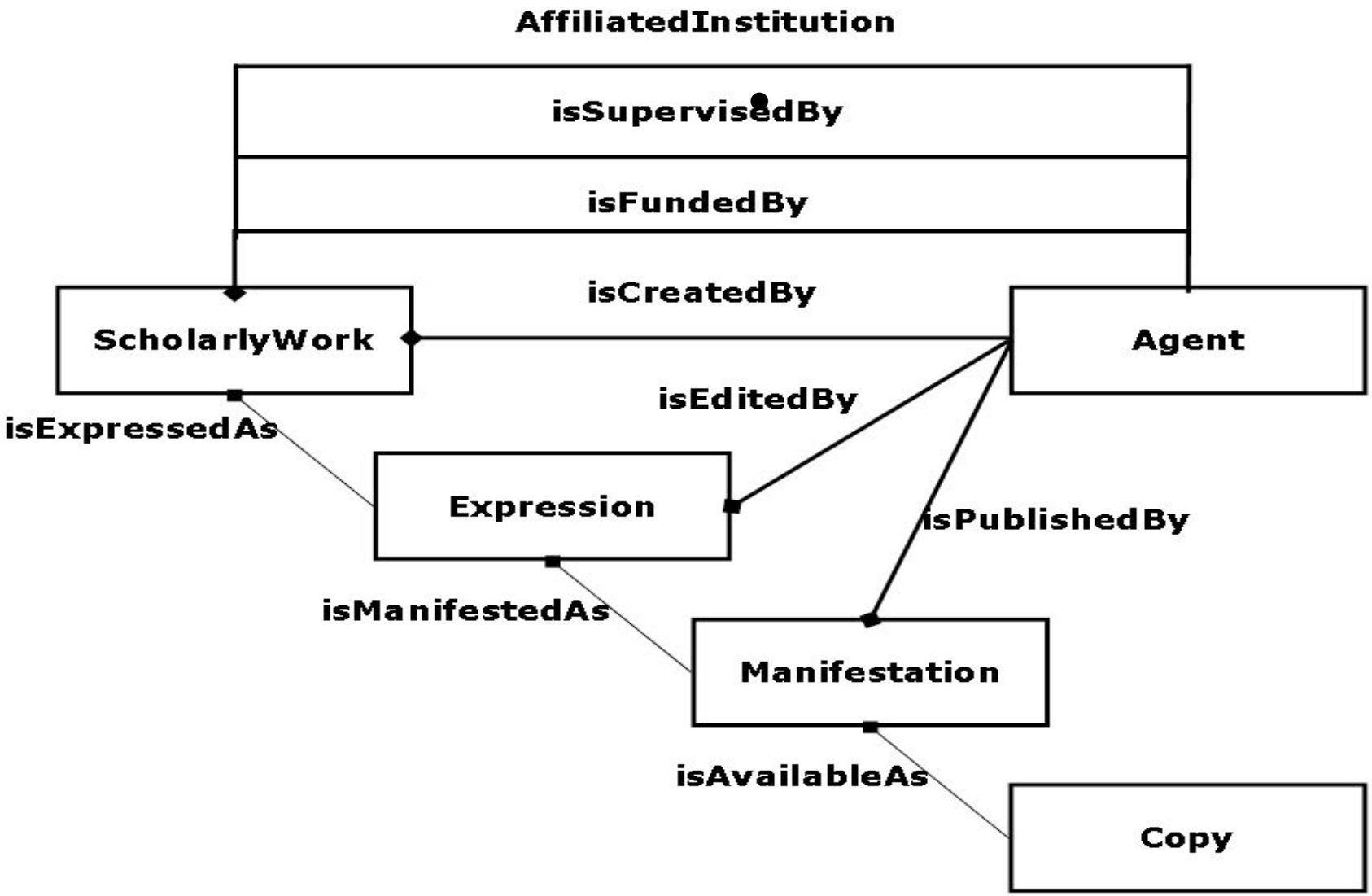
Functional Requirements

- What activities must the application support?
  - Who are the users? How expert are they?
- Scholarly Works Application Profile examples:
  - “Facilitate identification of open access materials.”
  - “Be compatible with preservation metadata approaches.”
  - “Enable identification of research funder and project code.”
  - “Support navigation between different 'versions' of the same eprint.”
- DC Collections Application Profile examples:
  - “Enable searching on the entity that owns the collection”
  - “Enable selection of a collection based on a textual description”

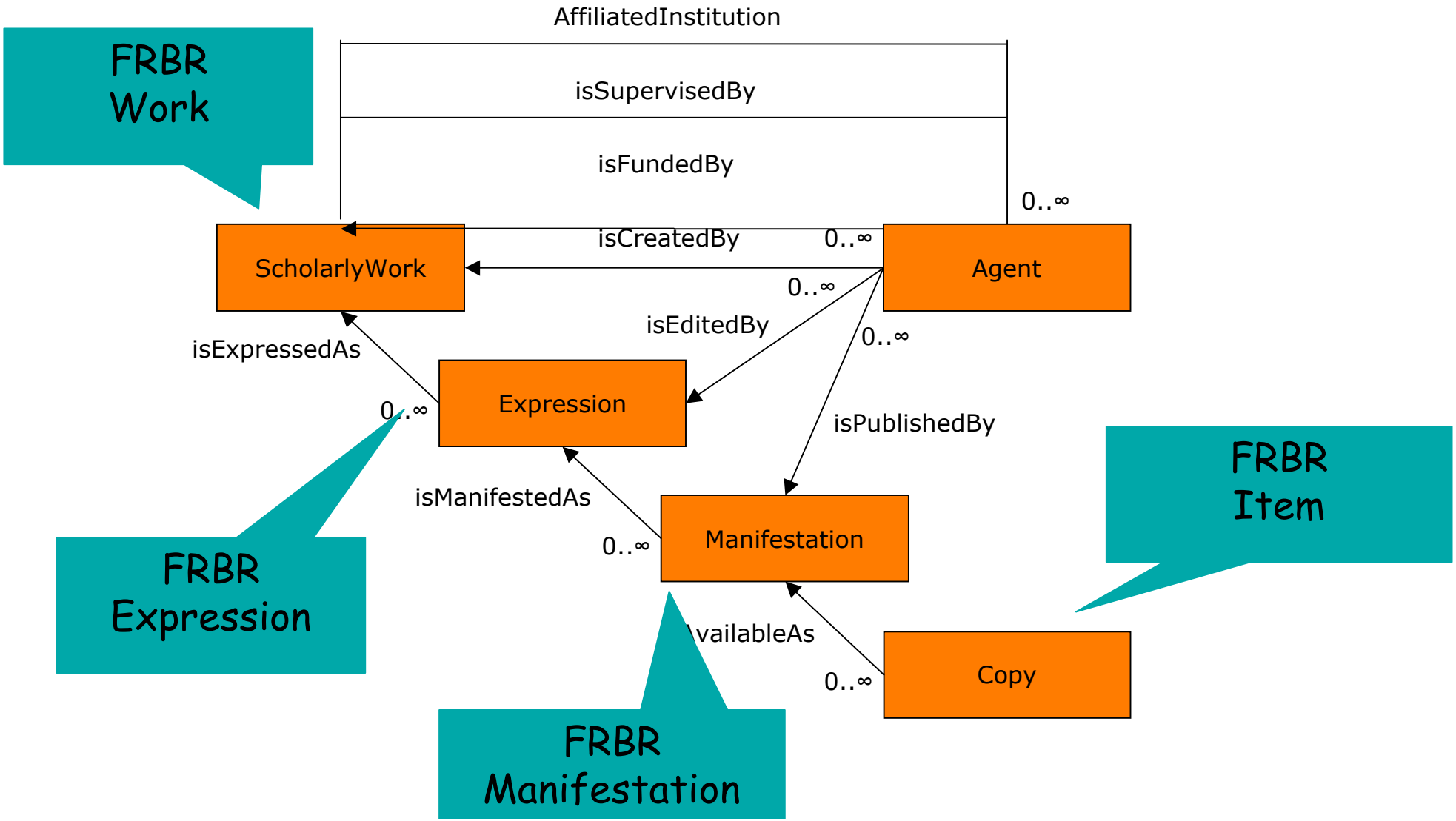
What things are being described?



Domain Model for Scholarly Works Application Profile

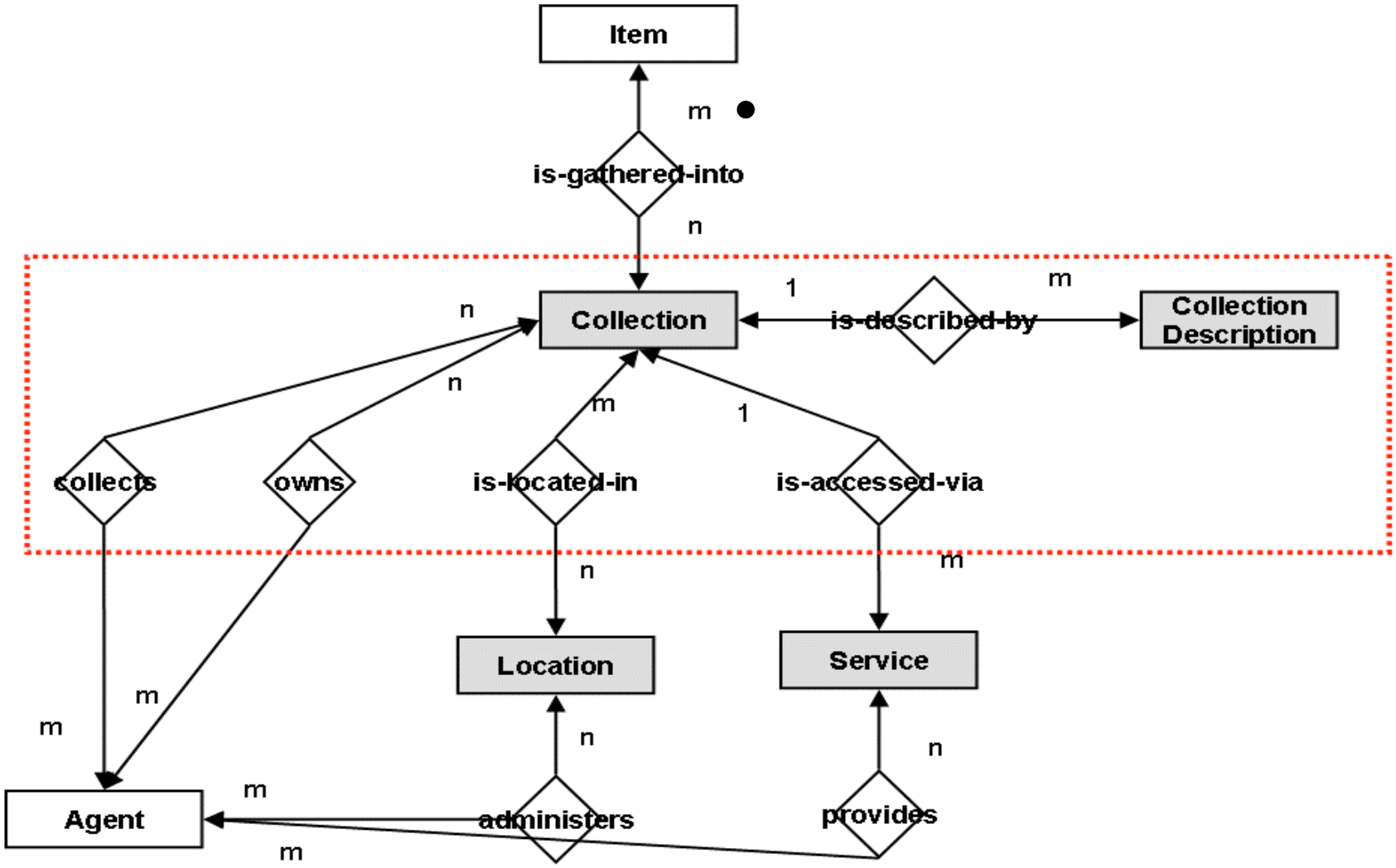


Based on Community Domain Model
Functional Requirements for Bibliographic Records

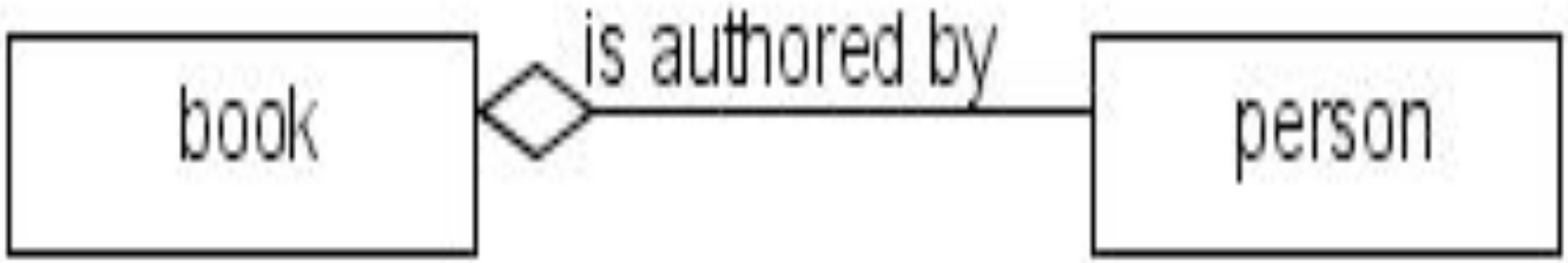


Dublin Core Collections Application Profile

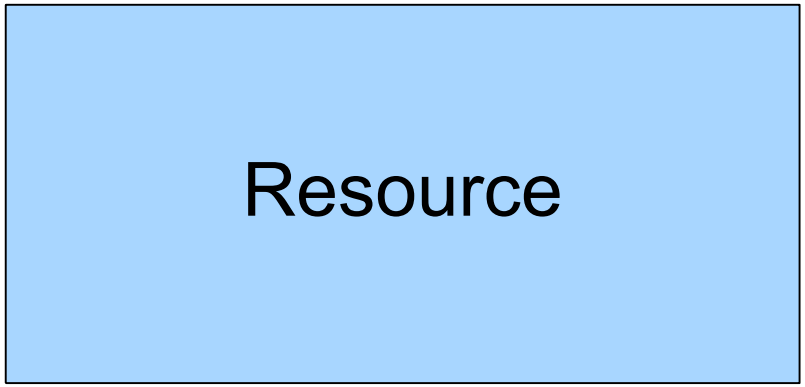
http://dublincore.org/groups/collections/collection-application-profile/2007-03-09/



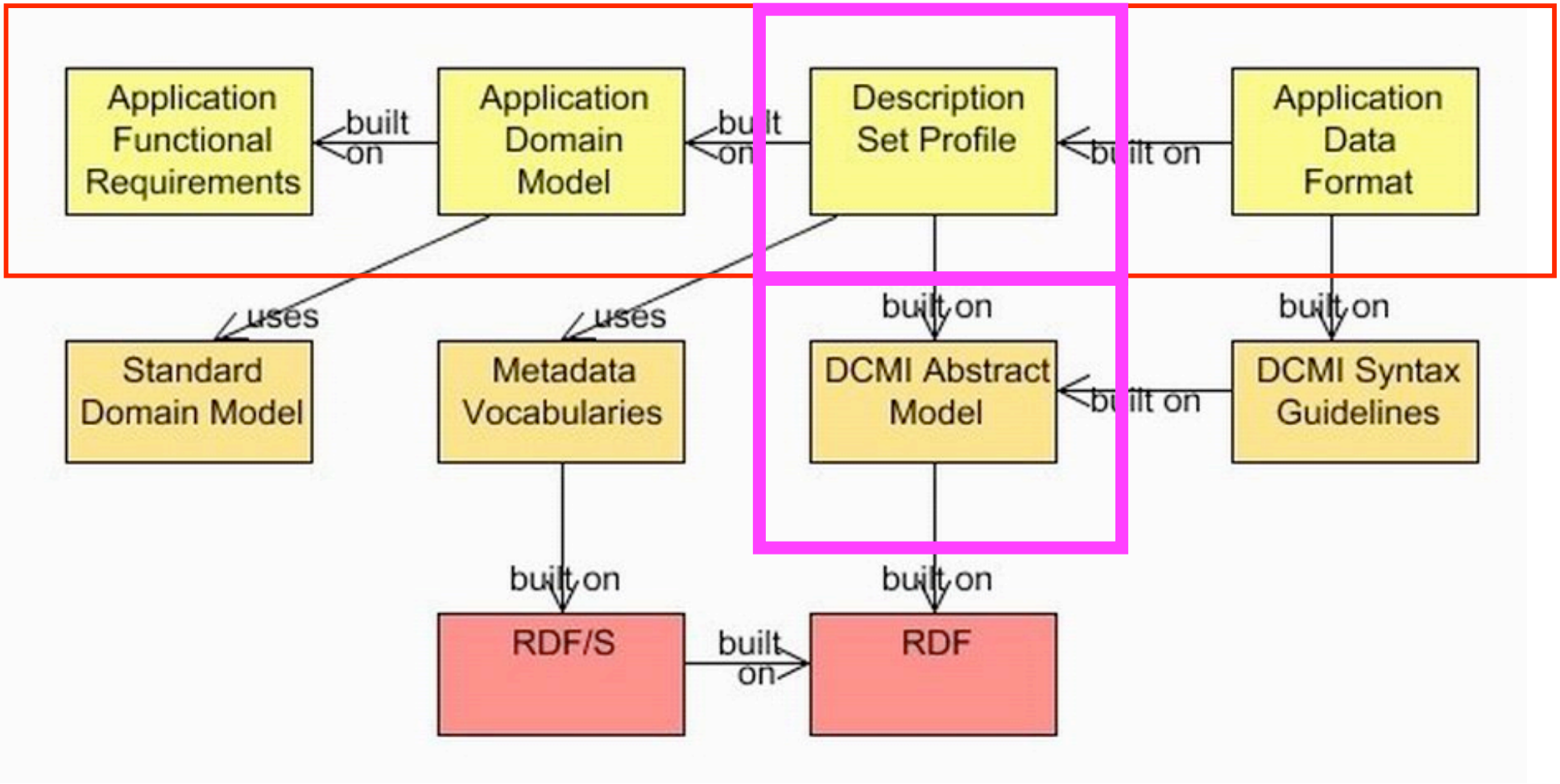
A simple domain model



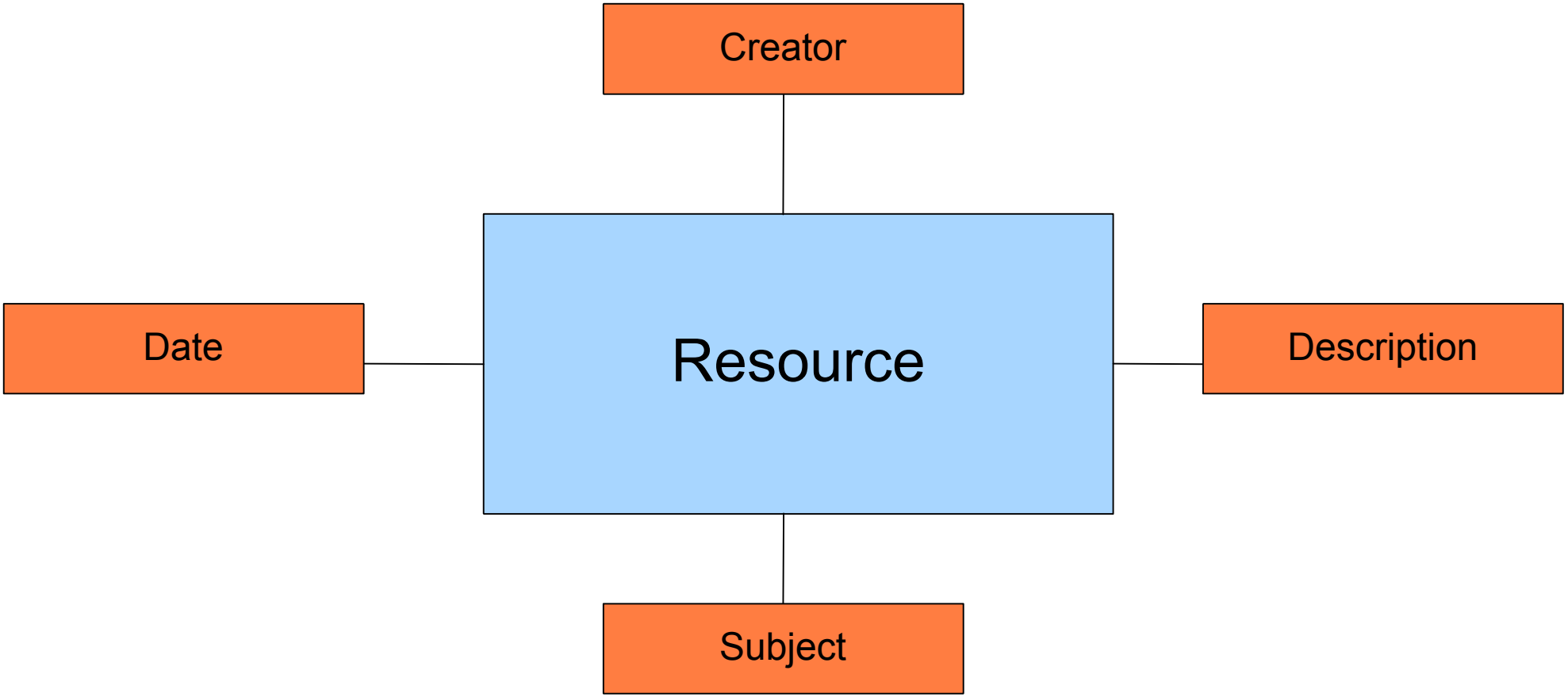
The simplest domain model...!



Description Set Profile, based on DCAM



“Decorate” the domain model with properties



Properties used to describe entities of the Domain Model

ScholarlyWork
title
subject
abstract
identifier

Expression
title
date available
status
version number
language
genre / type
copyright holder
bibliographic citation
identifier

Manifestation
format
date modified

Agent
name
type of agent
date of birth
mailbox
homepage
identifier

Copy
date available
access rights
licence
identifier

http://www.ukoln.ac.uk/repositories/digirep/index/Eprints_Application_Profile

Constraints on properties used

Description of the eprint as a ScholarlyWork

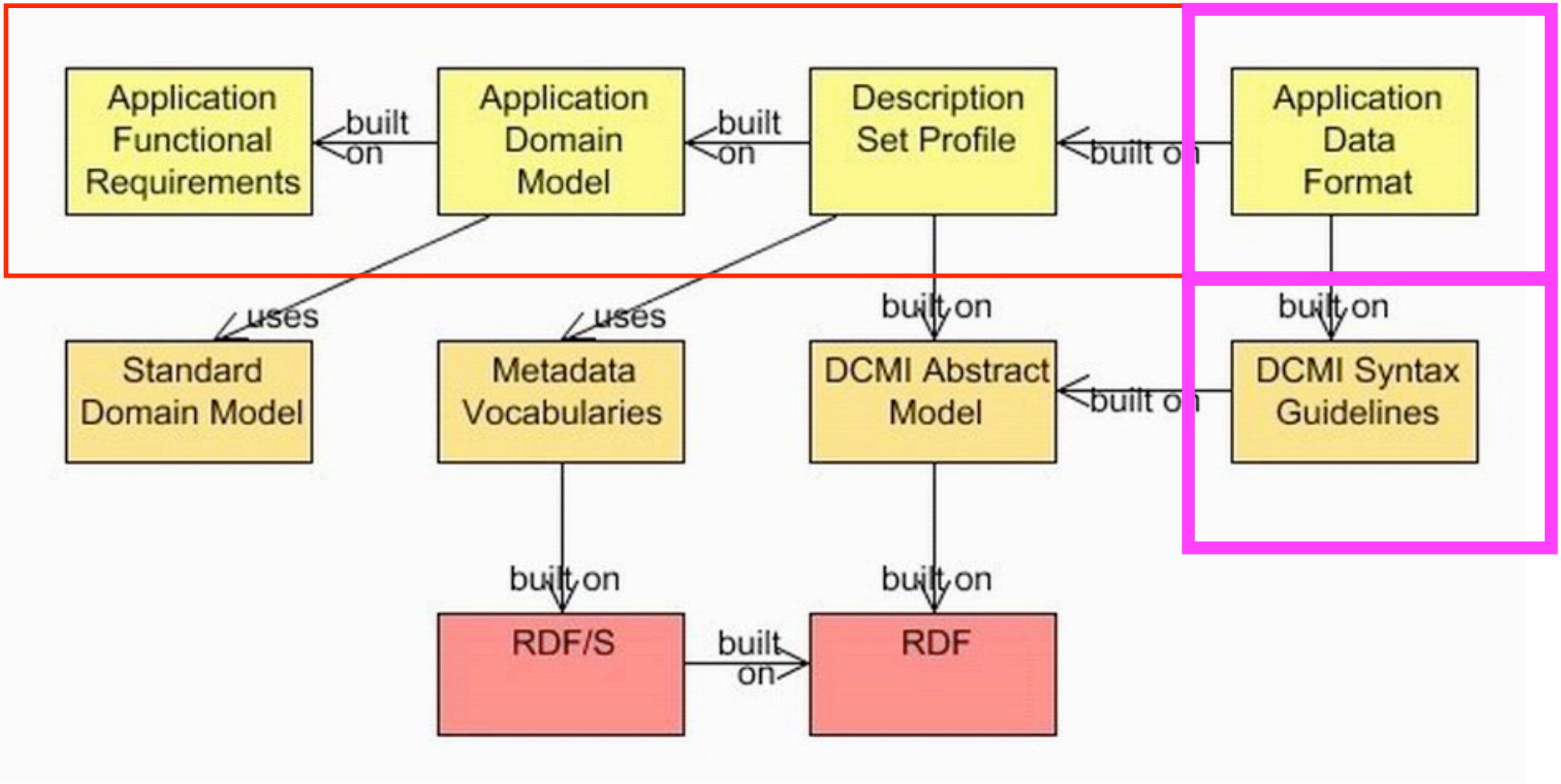
Entity type

Property	http://purl.org/dc/elements/1.1/type
Min occurrence	1
Max occurrence	1
Literal?	No
Definition	The type nature or genre of the content of the resource.
Eprint-specific recommendation	Each of the entity *descriptions* in the *description sets* conforming with this application profile will need to be explicitly typed. This is done using a dc:type *statement* with one of the following *value URIs* taken from the Eprints EntityType Vocabulary Encoding Scheme corresponding to the entity being described.
Value (Non-Literal)	**Value URI Constraint:** **Occurrence**: mandatory **Choose from**: http://purl.org/eprint/entityType/ScholarlyWork/ **Vocabulary Encoding Scheme Constraint** **Occurrence**: mandatory **Choose from**: http://purl.org/eprint/entityType/ **Value String Constraint:** Max occurrence: 0

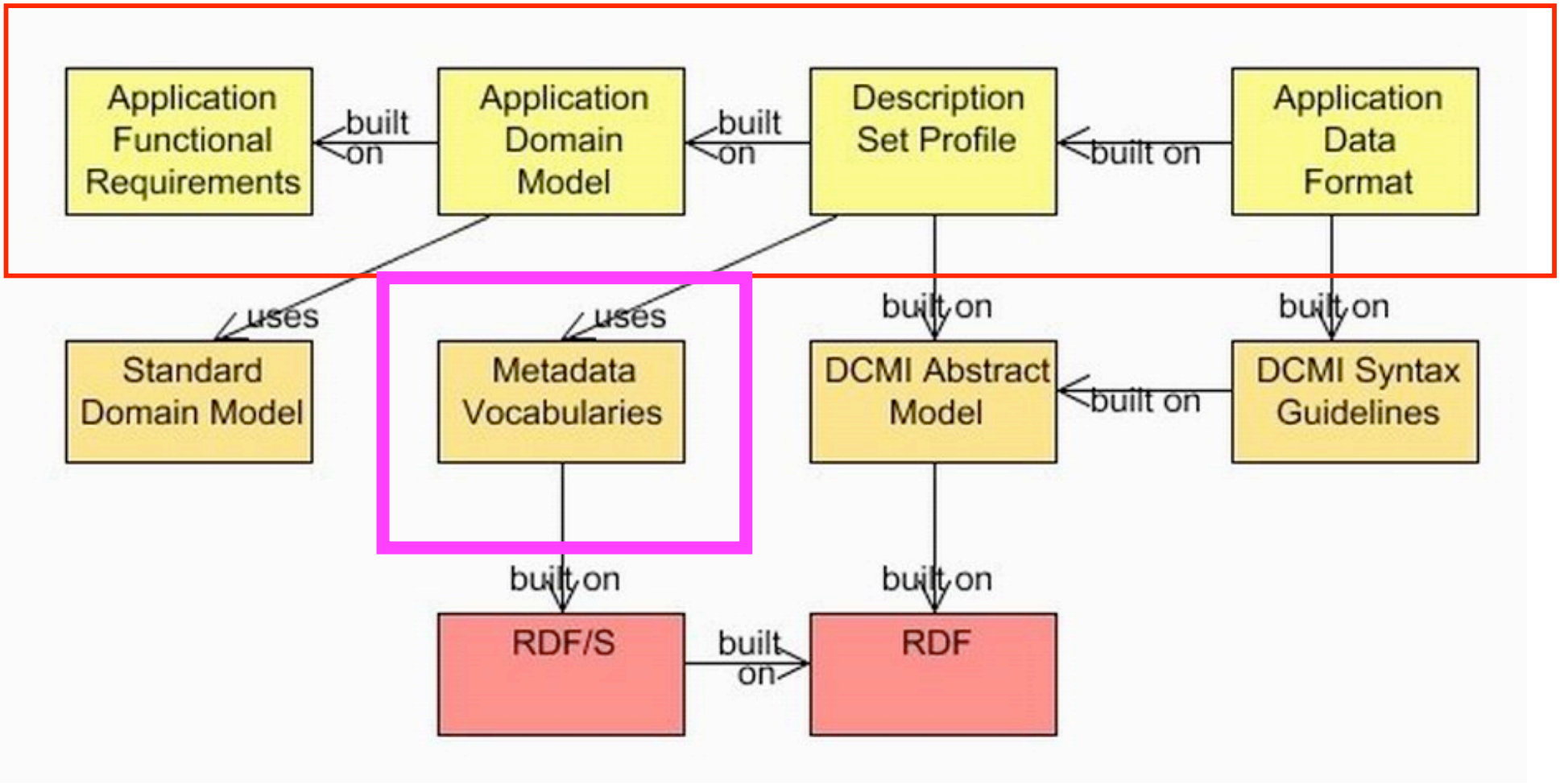
For example:

```
Statement (

        Property URI ( dc:type )
        Vocabulary Encoding Scheme URI ( eprint:EntityType )
        Value URI ( <http://purl.org/eprint/entityType/ScholarlyWork> )

)
```

Free choice of (model-based) syntax



Metadata vocabularies, built on RDF



Mixed namespaces

ScholarlyWork
dc:title
dc:subject
dc:abstract
dc:identifier

Expression
dc:title
dc:available
ep:status
ep:version
dc:language
dc:type
ep:copyrightHolder
dc:biblioCitation
dc:identifier

Manifestation
dc:format
dc:modified

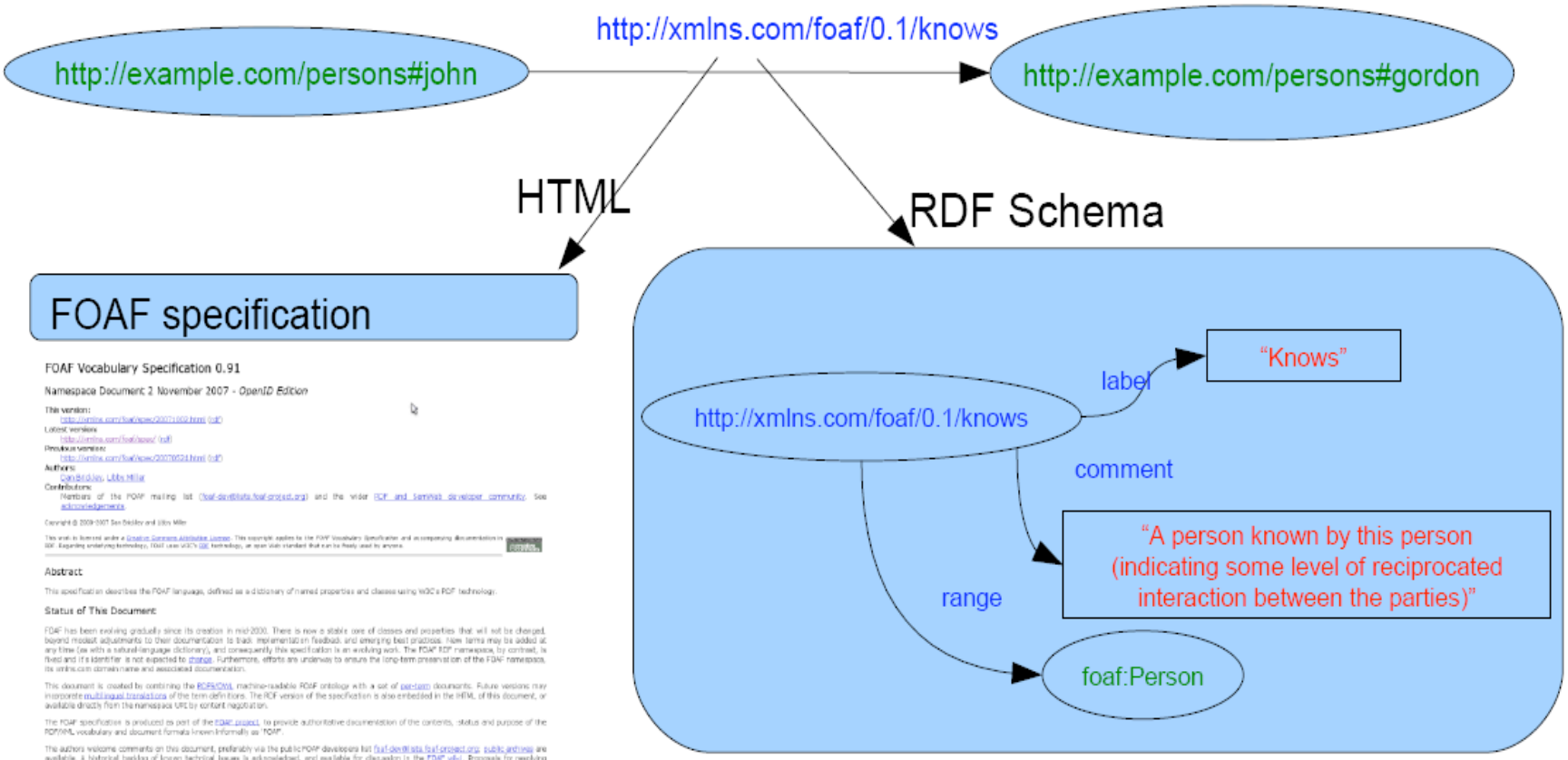
Agent
foaf:name
dc:type
foaf:birthday
foaf:mbox
foaf:homepage
dc:identifier

Copy
dc:available
dc:accessRights
dc:license
dc:identifier

ep:isExpressedAs
ep:isManifestedAs
ep:isAvailableAs
.
.
.

Each term is formally declared in a vocabulary.
If a term does not yet exist, declare it yourself!

“Follow your nose” to the definitions...



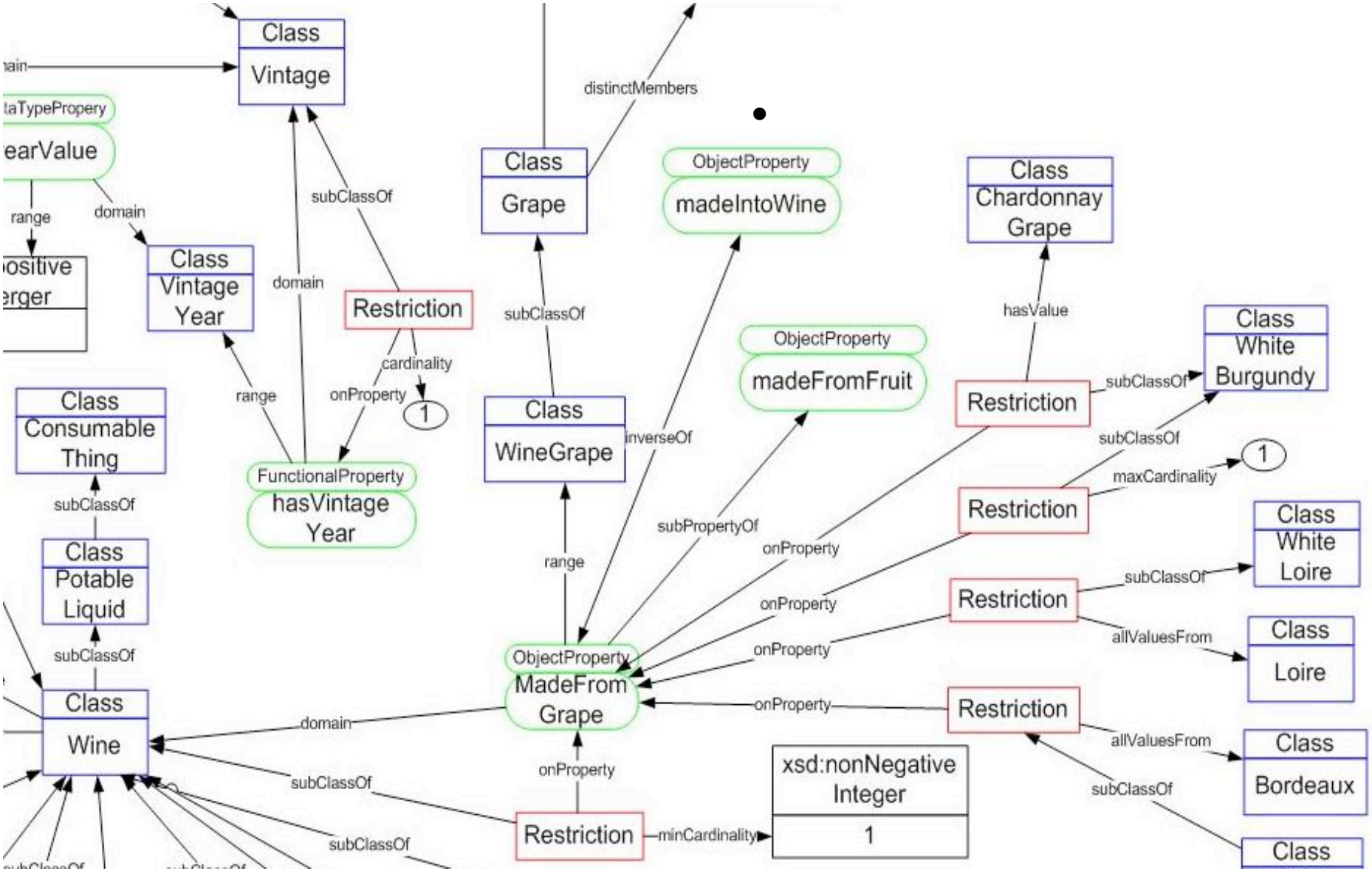
Value Vocabularies

- Simple lists...
  - “Animal”, “Vegetable”, “Mineral”
  - “Yes”, “No”, “Maybe”
- Controlled vocabularies...
  - Subject headings (“China – History”)

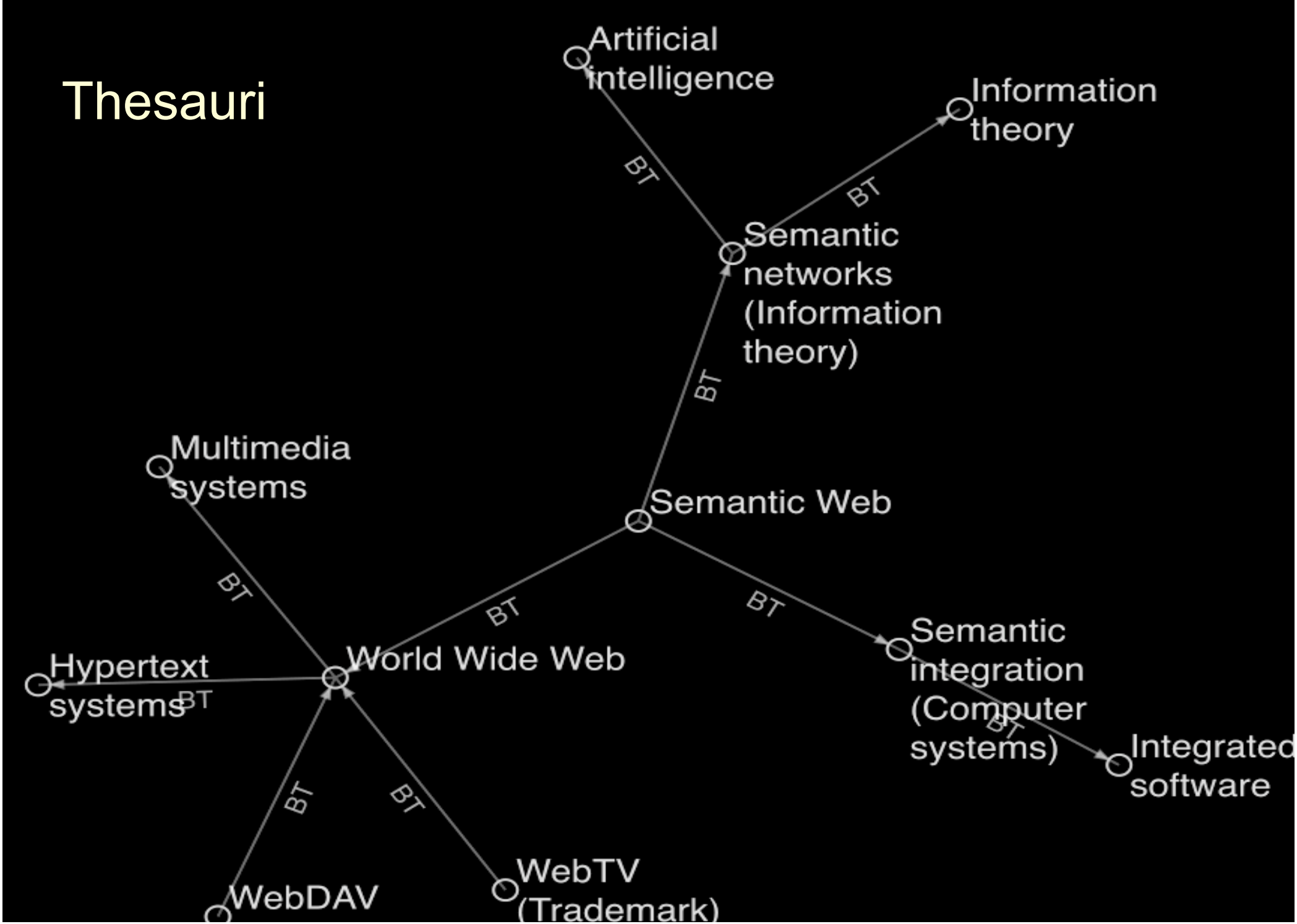
Formal Ontology versus KOSes

- Formal Ontology
  - Provides an interpretation of reality
  - Asserts axioms or facts about things in the world
  - Inference using logical entailments
- Knowledge Organization System
  - Concepts may be linked pragmatically
  - Convenient or intuitive “maps” of domains
  - Aid to navigation or for finding relevant objects
  - Re-engineering as formal ontology may be hard

OWL Wine Ontology



Thesauri


Artificial intelligence
Information theory
BT
BT
Semantic networks (Information theory)
BT
Multimedia systems
Semantic Web
BT
BT
BT
Hypertext systems
BT
World Wide Web
Semantic integration (Computer systems)
BT
Integrated software
BT
BT
WebDAV
WebTV (Trademark)

Modeling Advice (from Sam Oh)

- There are many incorrect ways to model a domain, but no single correct way
  - Choice between viable alternatives depends on application
  - Modeling choices may be subjective, even aesthetic
- Ontology development is an iterative process
  - Early prototyping is good
  - Revisit earlier decisions as ontology develops

Interoperability Levels for Dublin Core metadata

- 4: Description Set Profile Interoperability
  - Shared formal vocabularies and constraints in records
- 3: Description Set syntactic interoperability
  - Shared formal vocabularies in exchangeable records
- 2: Semantic interoperability
  - Shared vocabularies based on formal semantics
- 1: Informal interoperability
  - Shared vocabularies defined in natural language