情報組織化研究グループ　2011.4.16

Dublin Coreのこころ

杉本重雄
筑波大学･図書館情報メディア研究科
知的コミュニティ基盤研究センター
sugimoto@slis.tsukuba.ac.jp

はじめに

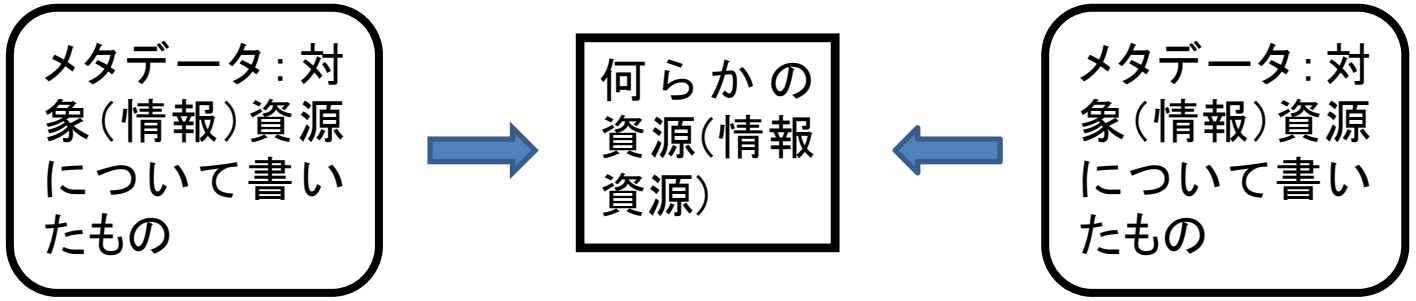
- Dublin Coreはとても有名なメタデータ標準である一方、十分に理解されていない標準であるとも思います。
- Dublin Coreの基本的な考え方を伝えること、理解していただくことを目的にお話します。
- そのため、具体的な記述方法や応用方法といったことには触れることは致しません。
- １９９６年ごろから開発の過程を見てきた情報技術的背景を持つ研究者の視点からの内容です。

構成

1. メタデータとは
   - メタデータを必要とする世界
   - メタデータのモデル
2. Dublin Coreについて
   - Simple Dublin Core
   - Resource Description FrameworkとDublin Core
   - Application Profile
3. メタデータスキーマについて
4. おわりに

1. メタデータとは

- データに関する(構造化された)データ
  (Structured) Data about Data
- 記述対象に関する「何か」を書いたもの



メタデータの例

- ペットボトルのラベル：　ボトルの中身が何かを示すメタデータ
  - ラベルをはがしてしまうと中身がわからなくなる
- 名刺：　ある人について書いたメタデータ
  - 所属、役職、名前、連絡先
- チケット：　受けられるサービスについて書いたメタデータ
  - 乗車券：　いつ、どこからどこまで、大人・子ども
- イベント案内
  - 日時、場所、内容

メタデータの例

- ペットボ
  -
-
-
- イベン
  - 日時、場所

素朴な疑問

メタデータは、データに関するデータ。とすれば、「ペットボトル」や「人」ってデータ？

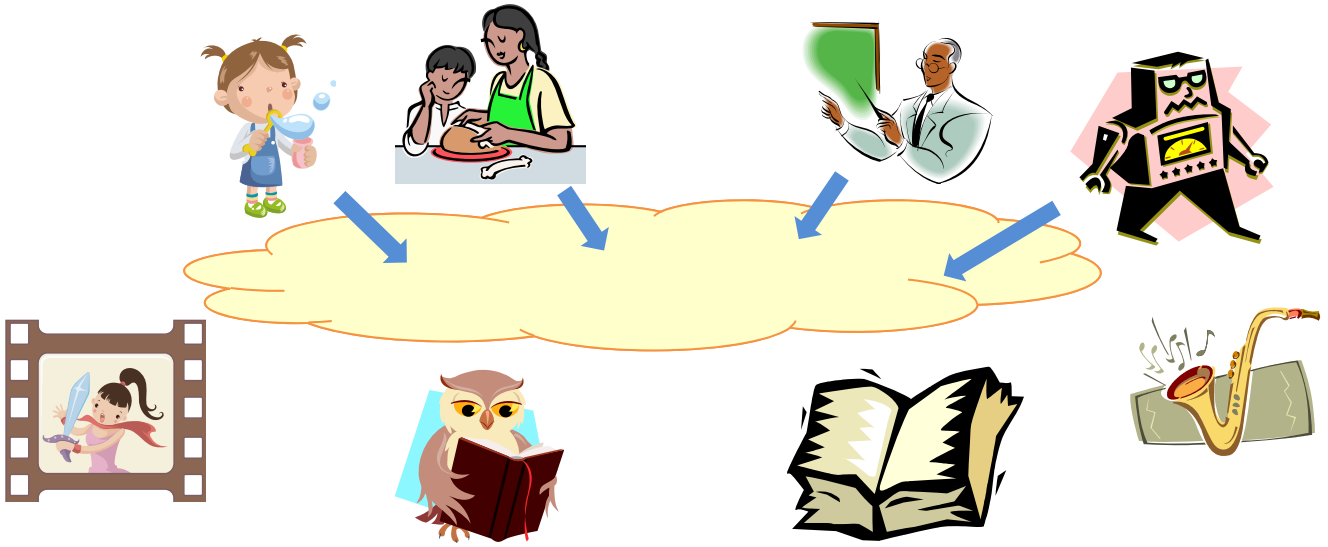
メタデータの例

- ペッ
  示
  -
-
-
- イ
  - 日時、場所

ネット上の世界では、すべての実体は、ネット上で識別できる何らかの実体として表現できねばならない。電子文書のようなもともと電子的な実体もあれば、ある「人」を表す仮想実体として作られるものもある。この実体を「データ」ということばでとらえる。物理的な実体と仮想的な実体を区別しない。

メタデータ - ネットワーク上での資源利用

- 資源： 情報資源・サービス資源
  - ネットワーク上のどこかにあるもの
- メタデータ： 資源を探し，アクセスし，評価し，利用し，取り引きするために必要なデータ



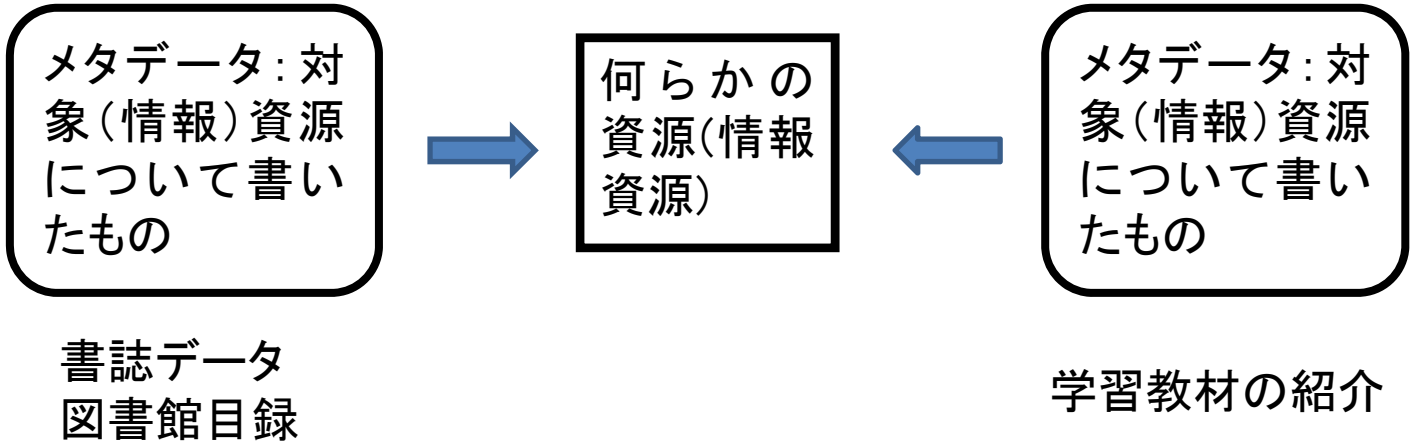
メタデータ無しに利用できるか?

メタデータとその役割

- 目録や索引なしに図書館や公文書館のサービスは考えられない。目録や索引は典型的なメタデータ
- ネットワーク時代におけるメタデータの役割
  - 探す、選ぶ、アクセスする、利用する、管理する、保存する、そのほか
  - これをネットワーク越しに行うとすれば、メタデータの重要性が直感的に理解できる

メタデータのモデル

- 記述対象に関する「何か」を書いたもの
- 「何か」は目的によって異なる



例： 名刺

- 杉本の名刺

筑波大学　図書館情報メディア研究科
知的コミュニティ基盤研究センター
教授

工学博士 杉　本　重　雄

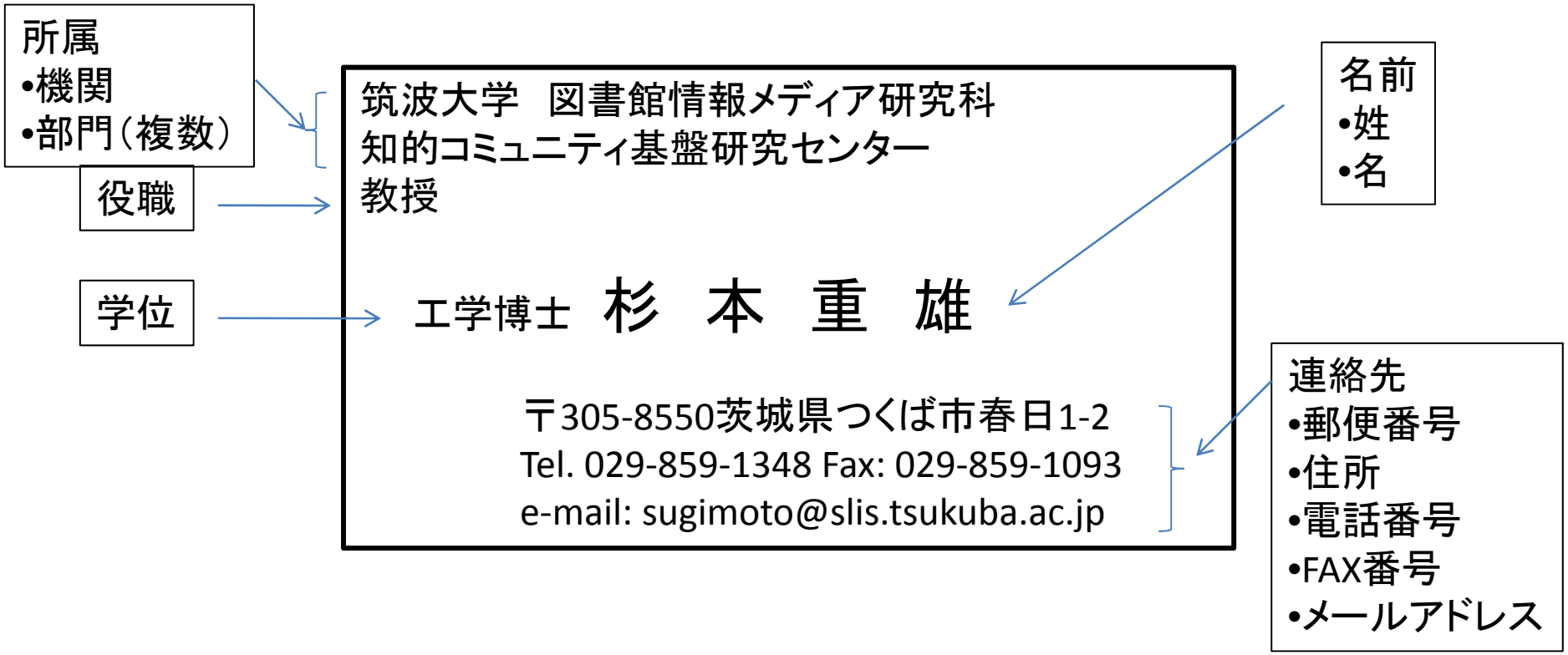
〒305-8550茨城県つくば市春日1-2
Tel. 029-859-1348 Fax: 029-859-1093
e-mail: sugimoto@slis.tsukuba.ac.jp

例： 名刺

- 人間が読むと内容は理解できる

例： 名刺

- 人間が読むと内容は理解できる（より細かく）



例： 名刺

属性	値
所属	筑波大学・図書館情報メディア研究科・知的コミュニティ基盤研究センター
役職	教授
学位	工学博士
名前	杉本重雄
連絡先	〒305-8550 茨城県つくば市春日1-2
	Tel. 029-859-1348
	Fax: 029-859-1093
	e-mail: sugimoto@slis.tsukuba.ac.jp

記述項目（属性）とその値（属性値）の組として表す

例：　名刺—より細かく

属性	属性2	値
所属	機関	筑波大学
	所属	図書館情報メディア研究科
	所属	知的コミュニティ基盤研究センター
役職		教授
学位		工学博士
名前	姓	杉本
	名	重雄
連絡先	郵便番号	305-8550
	住所	茨城県つくば市春日1-2
	電話	029-859-1348
	FAX	029-859-1093
	e-mail	sugimoto@slis.tsukuba.ac.jp

例： 名刺―元に戻す

属性		値
所属		筑波大学
		図書館情報メディア研究科
		知的コミュニティ基盤研究センター
役職		教授
学位		工学博士
名前		杉本
		重雄
連絡先		305-8550
		茨城県つくば市春日1-2
		029-859-1348
		029-859-1093
		sugimoto@slis.tsukuba.ac.jp

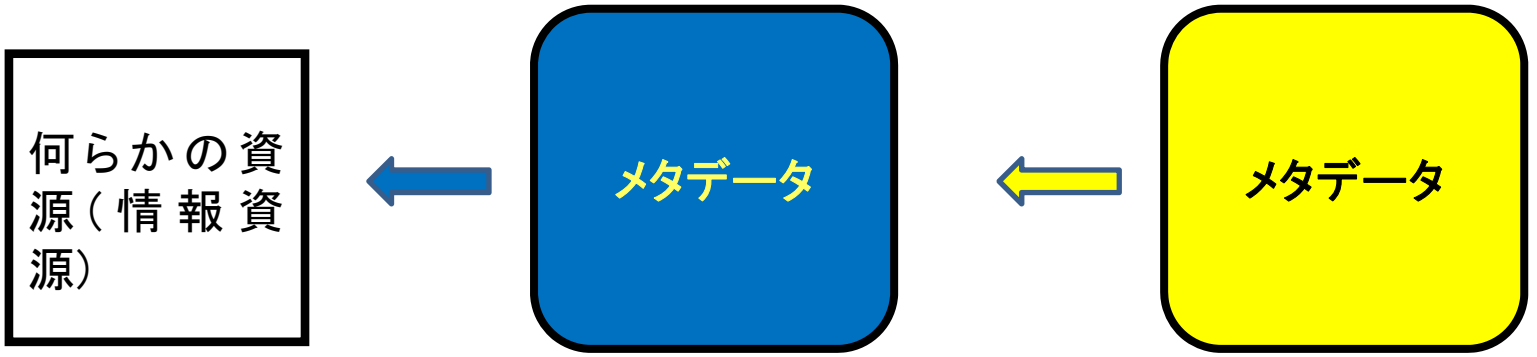
例： 名刺—この例からわかること

属性		値
		sugimoto@slis.tsukuba.ac.jp

- メタデータの基本構造
  - 記述対象の属性（記述項目）と属性値の対の集まり
- 属性値は構造を持つことがある
  - ひとつの属性値がいくつかの属性・属性値の対の集まりからできている
- 属性値の書き方
  - 属性ごとに属性値の書き方（換言すると、属性値の型）が決まる。たとえば、メールアドレスや電話番号
- 属性の集合
  - どのような属性について記述するか、どのようにして属性値を対象から取り出すかは目的に応じてきまる

メタデータとメタ・メタデータ

- メタ・メタデータ
  - 原理的には「メタ」は何回でも繰り返せる



メタデータのメタデータの例（1）

- 杉本の名刺

筑波大学　図書館情報メディア研究科
知的コミュニティ基盤研究センター
教授

工学博士　杉　本　重　雄

〒305-8550茨城県つくば市春日1-2
Tel. 029-859-1348 Fax: 029-859-1093
e-mail: sugimoto@slis.tsukuba.ac.jp

杉本のメタデータ

作成：　2010-4
印刷：　xx印刷
コスト：　yy円／枚
両面：　日本語・英語

名刺のメタデータ
杉本から見るとメタ・メタデータ

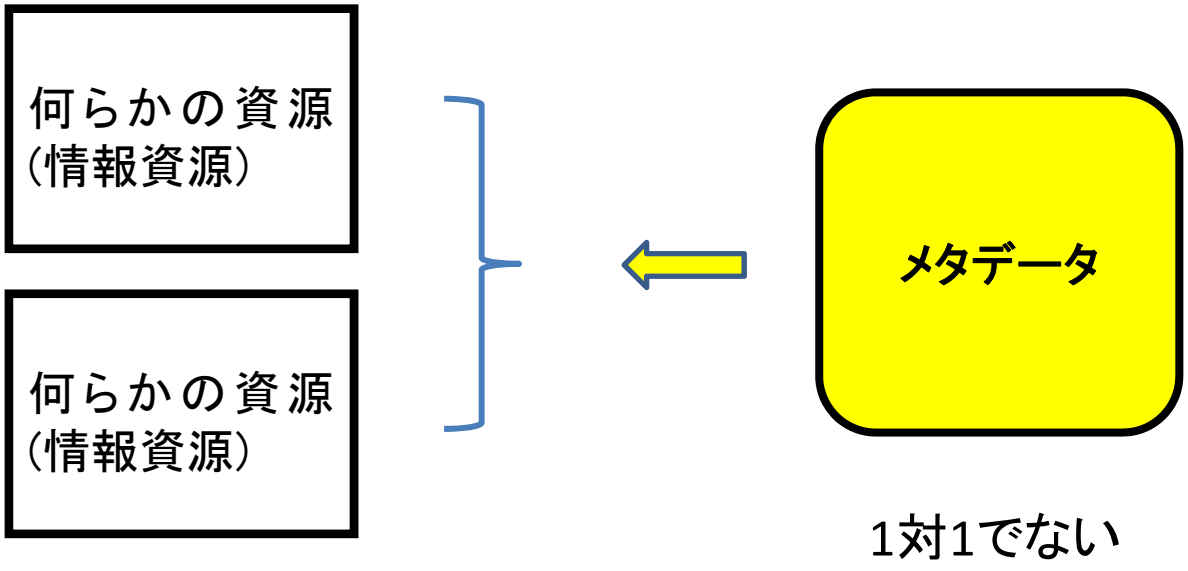
杉本

メタデータ—１対１原則

- 1対1原則:
  - ひとつの記述対象に対してひとつのメタデータ
  - Dublin Coreの基本概念

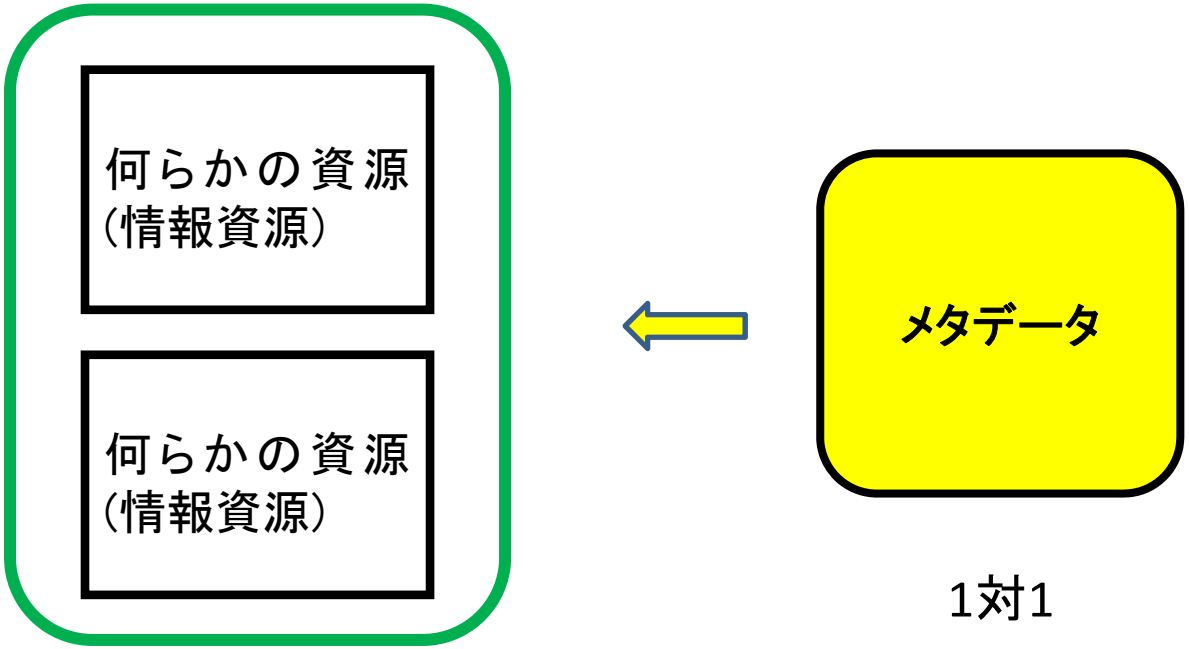
メタデータ―１対１原則

- 複数の記述対象をひとまとめに1件のメタデータとして書いたものは1対1ではない



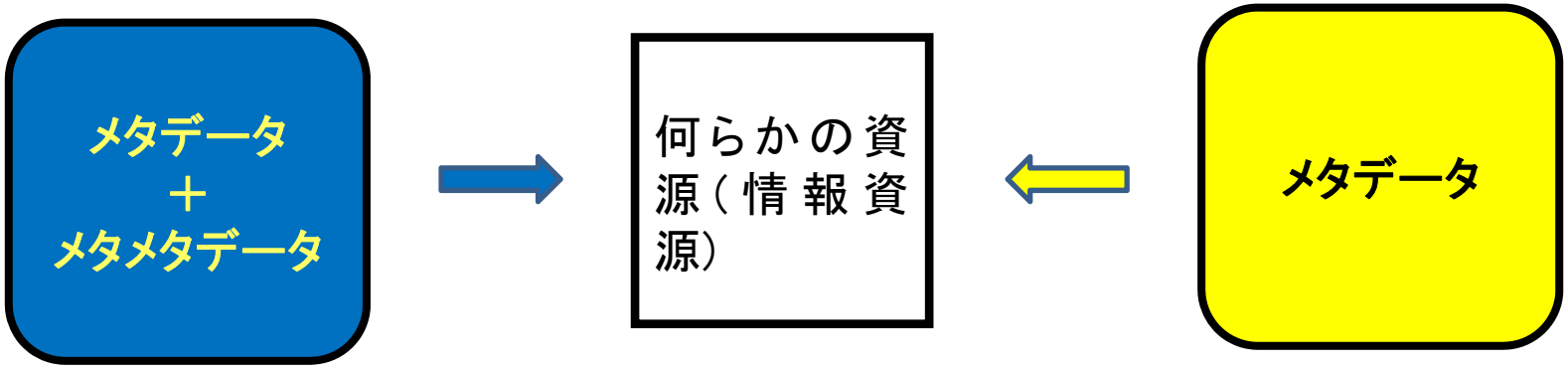
メタデータ―１対１原則

- 複数の記述対象であっても、それをひとまとめに書けば1対1



複数の資源の集まりをひとつの資源とみなしたもの

メタデータの基本モデル―1対1原則

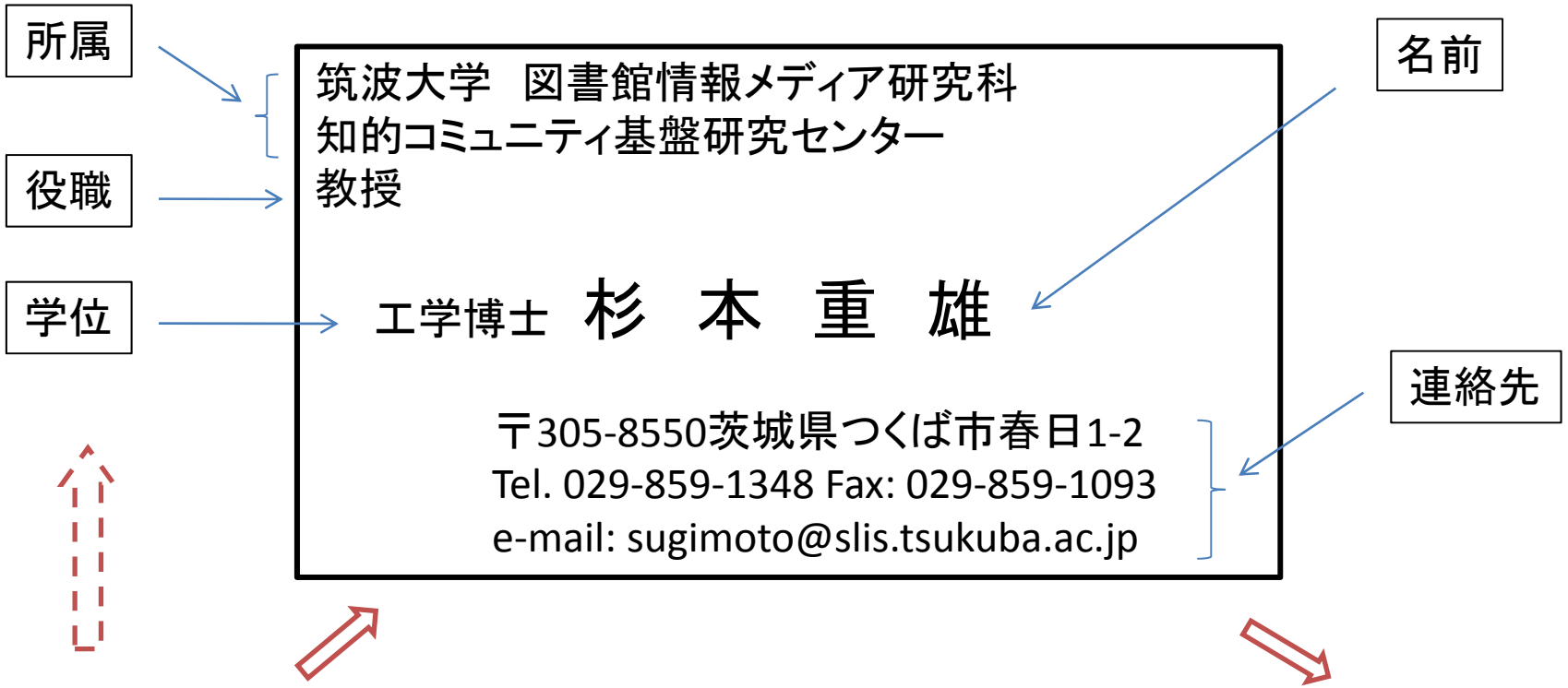


- メタデータとメタメタデータをひとまとめにして書いたものの例
  - 目録データと目録データの管理情報
  - Learning Object Metadata (LOM)のメタメタデータのカテゴリ

メタデータの基本

- １件のメタデータは属性と属性値の対の集り
- メタデータの基本
  - 記述すべき属性（＝記述項目）を決めること
  - 属性値の記述の仕方を決めること
- 属性
  - 名前とその意味によって定義する
  - 記述上の構造的制約（必須・省略可・繰り返し）
- 属性値について決めるべきこと
  - どのような形式、どのような種類の値を書くべきか
  - どのようにして記述対象から値を抽出するか

メタデータのメタデータの例（2）

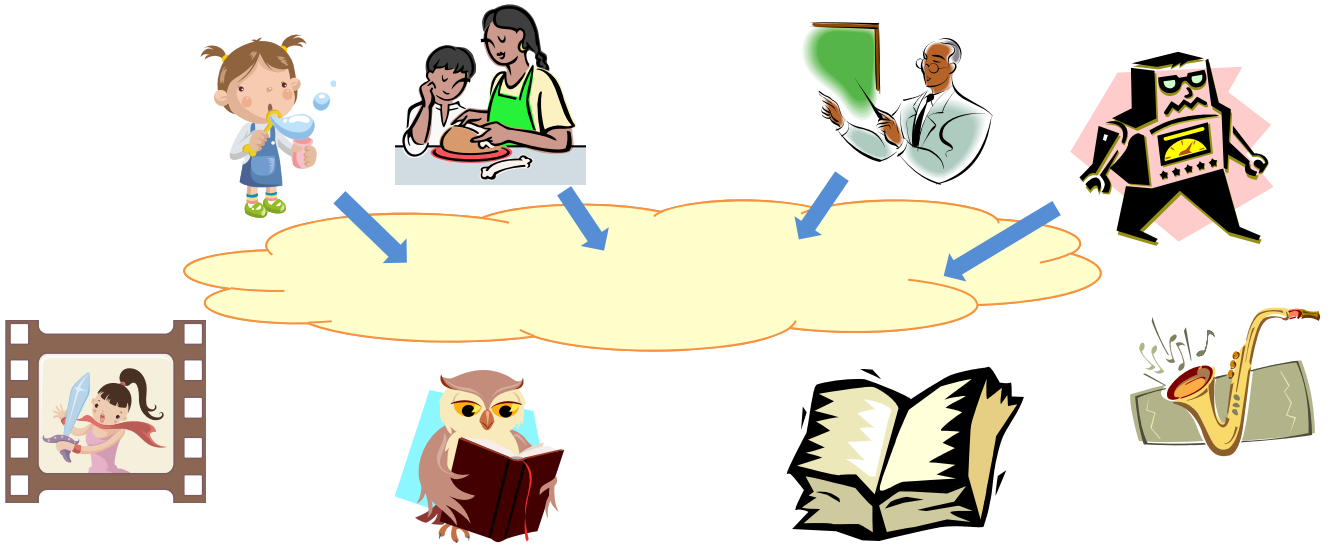


所属：所属機関の名前、部門の名前
役職：所属機関、部門の中での職名
・・・
属性や属性値の型の定義

杉本

ネットワーク上での資源利用とメタデータ

- （多くの）利用者は、まずネットにアクセスする
  - 「ググる」の持つ意味
    - 図書館や美術館を選んでアクセスするのではない
  - フラットな環境を前提に特化したサービスの提供
    - 特化とフラット化の両方に対応しなければならない



メタデータ無しに利用できるか?

インターネット上のメタデータの要件

- インターネット上では、いろいろな情報資源に関する情報を一つの入り口から得ることは普通
- 情報資源に関する情報（すなわちメタデータ）は、情報資源の種類や、情報資源の提供目的によって、どのような内容になるかが決まる。
  - メタデータの記述項目、記述内容（すなわちメタデータ規則、ないしメタデータスキーマ）は、種類や目的によってきまる。
- 上の二つは、互いに矛盾する要求である
  - ひとつの入り口⇒メタデータが統一されていて欲しい
  - 種類や目的ごと⇒メタデータは独自に定義したい

インターネット上のメタデータの要件

- 要求に応じて特化することと、ネットのフラットさにあうように一般化することという、互いに矛盾する要求を満たさねばならない
  - 個別の要求を認めつつ、相互運用性を高める
- 相矛盾する要求を満たすためにDublin Coreが行ったこと
  - 異なる領域で共通に使える属性を決める
    - 属性の意味の詳細化はそれぞれの領域に任せる
  - メタデータの構造の定義と記述項目（属性）の定義を明確に分離する
    - Application Profileの概念の明確化

2. Dublin Coreについて－概要

- インターネット上の多種多様な情報資源の発見と記述のためのメタデータ
  - Dublin Core Metadata Element Set
    → 1995年ごろから
    → 多様な資源に共通な記述要素
    → 15エレメントからなるシンプルさで有名になった
  - Descriptive Metadata
    - 情報資源の内容の記述
  - Dublin Core Metadata Initiative (DCMI)
    開発と維持管理のための組織

Dublin Coreの基本15エレメント(1/3)

タイトル	Title	情報資源に与えられた名前
作成者	Creator	情報資源の内容の作成に主たる責任を持つ実体
キーワード	Subject	情報資源の内容のトピック
内容記述	Description	情報資源の内容の記述
公開者	Publisher	情報資源を利用可能にすることに対して責任を持つ実体
寄与者	Contributor	情報資源の内容への寄与に対して責任を持つ実体

Dublin Coreの基本15エレメント(2/3)

日付	Date	情報資源のライフサイクルにおける何らかの事象に対して関連付けられた日付
資源タイプ	Type	情報資源の内容の性質もしくはジャンル
記録形式	Format	物理的表現形式ないしディジタル形式での表現形式
資源識別子	Identifier	与えられた環境において一意に定まる情報資源に対する参照

Dublin Coreの基本15エレメント(3/3)

出処	Source	現在の情報資源が作り出される源になった情報資源への参照
言語	Language	当該情報資源の内容の言語
関係	Relation	関連情報資源への参照
時空間範囲	Coverage	情報資源の内容が表す範囲あるいは領域
権利管理	Rights	情報資源に含まれる，ないしは関わる権利に関する情報

Dublin Coreについて－概要

- Simple Dublin Core
  - １５エレメント＋「すべてのエレメントが繰り返し可能、かつ省略可能」
  - 標準規格
    - ISO 15836
    - JIS X 0836 ダブリンコアメタデータ基本記述要素集合
- メタデータの規格として
  - 決めているのは記述項目のみ
  - 応用によって決まる構造的な制約や具体的な表現形式は決めない
    - 構造制約:　記述が必須、繰返し可など
    - 応用毎の規則を決めるのはApplication Profile

Dublin Coreについて－概要

- 現在のエレメント定義
  - 55エレメント
    - 意味を詳細化したエレメントが増えた
  - DCMI Termsと呼んでいる
- Application Profile
  - ネットワーク上でのメタデータの相互運用性のための重要な概念
  - 種々のメタデータ標準が含む記述要素を組合わせて問題にあわせた（応用向きの）メタデータ規則（メタデータスキーマ）を作るための基本概念

参考：現在のDublin Coreに関する資料

- 記述要素集合（エレメントセット）に含まれる語は下記の文書に定義＊

  http://dublincore.org/documents/dcmi-terms/

- 重要な概念: DCMI Application Profile
  - Singapore Frameworkと呼ばれている＊

  http://dublincore.org/documents/singapore-framework/

- Dublin Coreの定義をするためのAbstract Modelの定義が進められた＊

  http://dublincore.org/documents/abstract-model/

Resource Description Framework (RDF)とDublin Core

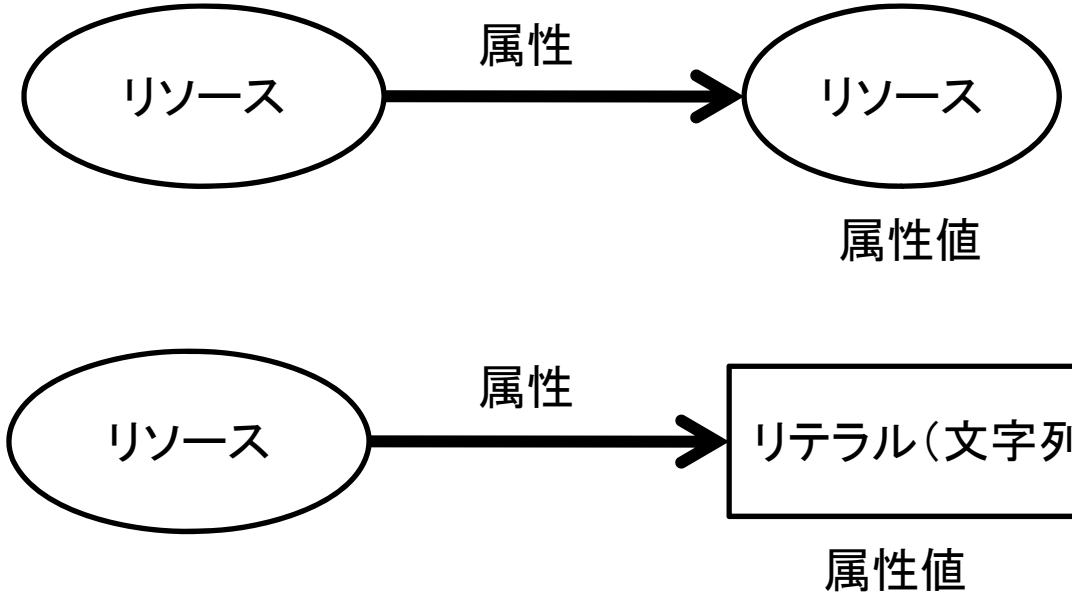
- RDFはWorld Wide Webコンソーシアム（W3C）のSemantic Webの活動の中で定義された、WWW上でのメタデータの表現と交換のための標準
  - RDF Model & Syntax：　データモデルと記述形式
    - RDF Modelは、実体間関係モデル（Entity Relationship Model）に基づく単純なデータモデル
  - RDF Vocabulary Description Language (RDF Schema)：RDFで用いる語彙（および語）の定義のモデルと形式を決める
    - 資源（リソース）が持つ属性（Property）と属性値（リソース、リテラル）の型（Class）

Resource Description Framework (RDF)とDublin Core

- Dublin Coreは、メタデータの表現のための独自のデータモデルを決めていない
  - データモデルは応用に依存するため、合意が難しかった
  - 90年代の議論には、部分構造を表すための記述子が含まれていたが、後に姿を消した。その背景には、RDFの登場、Application Profileの概念の登場がある
- RDFは、開発段階でDublin Coreの影響を受けている
  - 第2回ワークショップのWarwick Frameworkは、複数のメタデータ標準によるメタデータ記述の概念を与えている
  - Dublin Coreの会議には、常にW3Cからの参加
  - Semantic Webの最初のLeadのEric MillerはOCLC Researchの研究者でDublin Coreの議論にはやくから参加

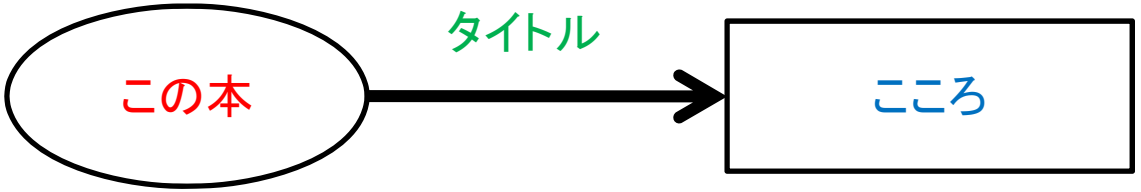
RDFのデータモデル

リソースとリソース、あるいはリソースとリテラル（文字列）の間を属性で関係付ける実体関係モデル（ERモデル）である。基本ユニットは3つ組＜リソース、属性、属性値＞で表される。

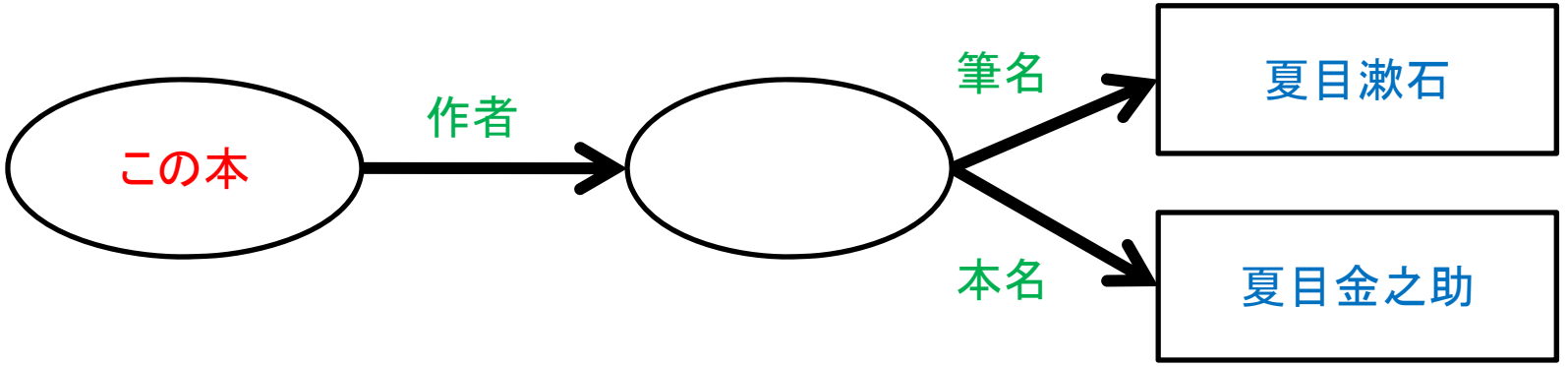


RDFのデータモデル

例：　この本のタイトルは「こころ」である
　　リソース　　属性　　属性値（リテラル）



例：　この本の作者は夏目漱石である



RDFのXML表現例（ごく簡単に）

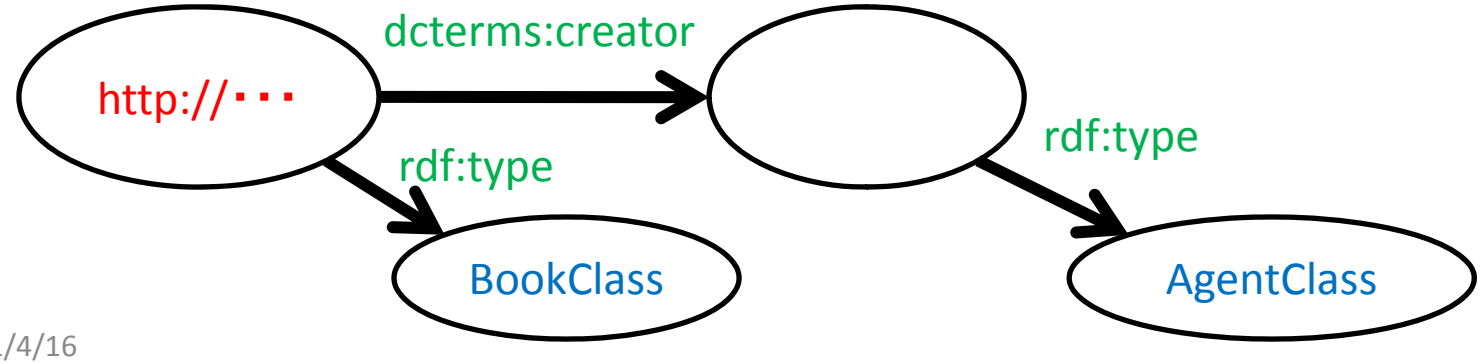
```
<rdf:RDF>
 <rdf:Description about=“この本のURI”>
  <タイトル>こころ</タイトル>
  <作者>
    <rdf:Description >
      <筆名>夏目漱石</筆名>
      <本名>夏目金之助</本名>
    </rdf:Description>
   </作者>
 </rdf:Description>
</rdf:RDF>
```

RDFのXML表現例（ごく簡単に）

```
<rdf:RDF>
 <rdf:Description about=“この本のURI”>
  <dcterms:title>こころ</dcterms:title>
  <dcterms:creator>
    <rdf:Description >
      <myScheme:penname>夏目漱石</myScheme:penname>
      <foaf:name>夏目金之助</foaf:name>
    <rdf:Description>
   </dcterms:creator>
 </rdf:Description>
</rdf:RDF>
```

RDF Schema

- RDFでは、メタデータ記述に用いる属性や属性値の種類（クラス）を定義できねばならない
  - たとえば、前の例では、Dublin Coreのcreatorやmy Schemaのpennameといったものを属性として定義できねばならない
  - RDFでは属性をProperty、属性値の種類をクラスと呼び、それらを定義できるようにしている



LegacyなDC(dc:)から新しいDC(dcterms:)へ

- 1998年の合意でできたSimple Dublin Coreに含まれる15の記述項目（DCMES）から、現在DCMIで合意している記述項目（dcterms、55項目）に至るまでの道のりについて
  ⇒Dublin Coreの歴史（とでもいうべき）
- DCMESの15項目（Legacy）は、dctermsの55項目とは分離して定義されている
  - 定義はそのまま有効になっており、かつ、dctermsの項目との間の関係付けがなされている。

LegacyなDC(dc:)から新しいDC(dcterms:)へ

- 1998年のWashington DCでのワークショップ（第6回）でのAgentの提案
  - Creator, Contributor, PublisherをまとめてAgentとすべきという提案
  - Simple Dublin Coreとして標準化のためにこの提案は受け入れなかった
  - これ以外にもRelationとSourceの間の上下関係に関する問題提起

LegacyなDC(dcmes)から新しいDC(dcterms)へ

- Qualified Dublin CoreとQualifier
  - より詳細なメタデータを記述するためにDublin Coreの記述項目集合に新しい要素（Qualifier, 限定子）をいれることにした
    - 属性の意味の詳細化
    - 属性の構造の詳細化
    - 属性値の型や表現形式の指示
- 2000年、Dublin Core Qualifierの定義
  - 属性の意味を詳細化する限定子と属性値の型を表す限定子の定義
  - 属性の下部構造を表すための限定子は除外された

LegacyなDC(dcmes)から新しいDC(dcterms)へ

- 属性の意味の詳細化の例
  - 日付（Date）⇒作成日付（Date Created）
  - 内容記述（Description）⇒目次（Table of Contents）、抄録（Abstract）
- Qualifier
  - 作成日付：「作成」という意味の詳細化。「作成」が意味を限定するので、定義の中では形容詞としてcreatedが定義された。
    - Abstractとは用法が異なったが
  - ところが、RDF上では、createdもabstractもproperty
  - Createdという名前のpropertyは、明らかにネーミング上の誤り。本来はDateCreatedとすべき。

LegacyなDC(dcmes)から新しいDC(dcterms)へ

- 2000年夏に、従来の15項目とは別にQualifierのセットを定義した
  - 新しい項目を定義するためのnamespaceを準備した（http://purl.org/dc/terms/）
    - 従来の15エレメントのnamespaceは http://purl.org/dc/elements/1.1/
  - Qualifierを用いて表すものをQualified Dublin Coreと呼んだ
- 構造を表す限定子が導入されなかったのは、明らかにRDFの影響
  - データ構造はRDFのデータモデルに基づくことで表現できる

LegacyなDC(dcmes)から新しいDC(dcterms)へ

- createdのようなネーミングが行われたのは、RDFをベースにすることが十分には理解されていなかったためであろう
  - Qualifierの議論とRDFの開発は同時期
- 2002年のフィレンツェでの会議の際に、Agentを、creator、contributor、publisherの上位属性として導入することの再提案があったが、この時点では認められずに終わる
  - Legacyデータとの互換性が問題
  - 現在のものになったのは2008年の文書

LegacyなDC(dcmes)から新しいDC(dcterms)へ

- 現在のDublin Coreの記述要素集合（dcterms:）をcreatorから理解する
  - dcterms:creatorはdcterms:contributorを詳細化したもの（下位属性、sub-property）として定義
  - Legacyな15項目のdc:creatorは基本的にそのまま残す
  - dcterms:creatorはdc:creatorの下位属性としても定義している
  - dcterms:contributorの場合、dc:contributorの下位属性、かつdcterms:creatorの上位属性として定義されている
- dc:のnamespaceに定義された要素をそのまま残す一方、dcterms:に定義された要素との間での関係をつけることで両者の間の相互運用性を確保している

LegacyなDC(dcmes)から新しいDC(dcterms)へ

- こうした修正とともにQualified Dublin CoreやQualifierといったことばは使われなくなった
  - ただし、createdやmodifiedといったそれだけでは意味がわからない（形容詞的な）属性名はそのまま残されている
- 現在の慣用表現では、DCMESといった場合にはLegacyな15項目を意味し、dctermsといった場合には、dcterms:のnamespaceに定義されている（すなわち、現在の）記述項目を意味する
  - dc:creatorは前者、dcterms:creatorは後者に含まれる

メタデータスキーマのモデルの明確化

Application ProfileとMetadata Vocabulary

- メタデータ規則における記述項目の定義と構造定義の切り離し
  - 記述項目の定義は、個々の項目の名前や意味を定義する
  - 構造定義は、記述項目ごとの記述条件（省略可能性、繰り返し数など）を定義する
  - 記述項目定義と構造定義の両方が与えられ、かつ、実現形式が決まらなければ実際にメタデータを作ることはできない
  - 構造定義や記述形式などが応用毎に決めなければならないのに対し、記述項目は応用毎に特化したものばかりではなく、汎用的なものもあり、かつ個別の応用向けの記述項目を汎用的なものを詳細化することによって作ることもできる。

メタデータスキーマのモデルの明確化

Application ProfileとMetadata Vocabulary

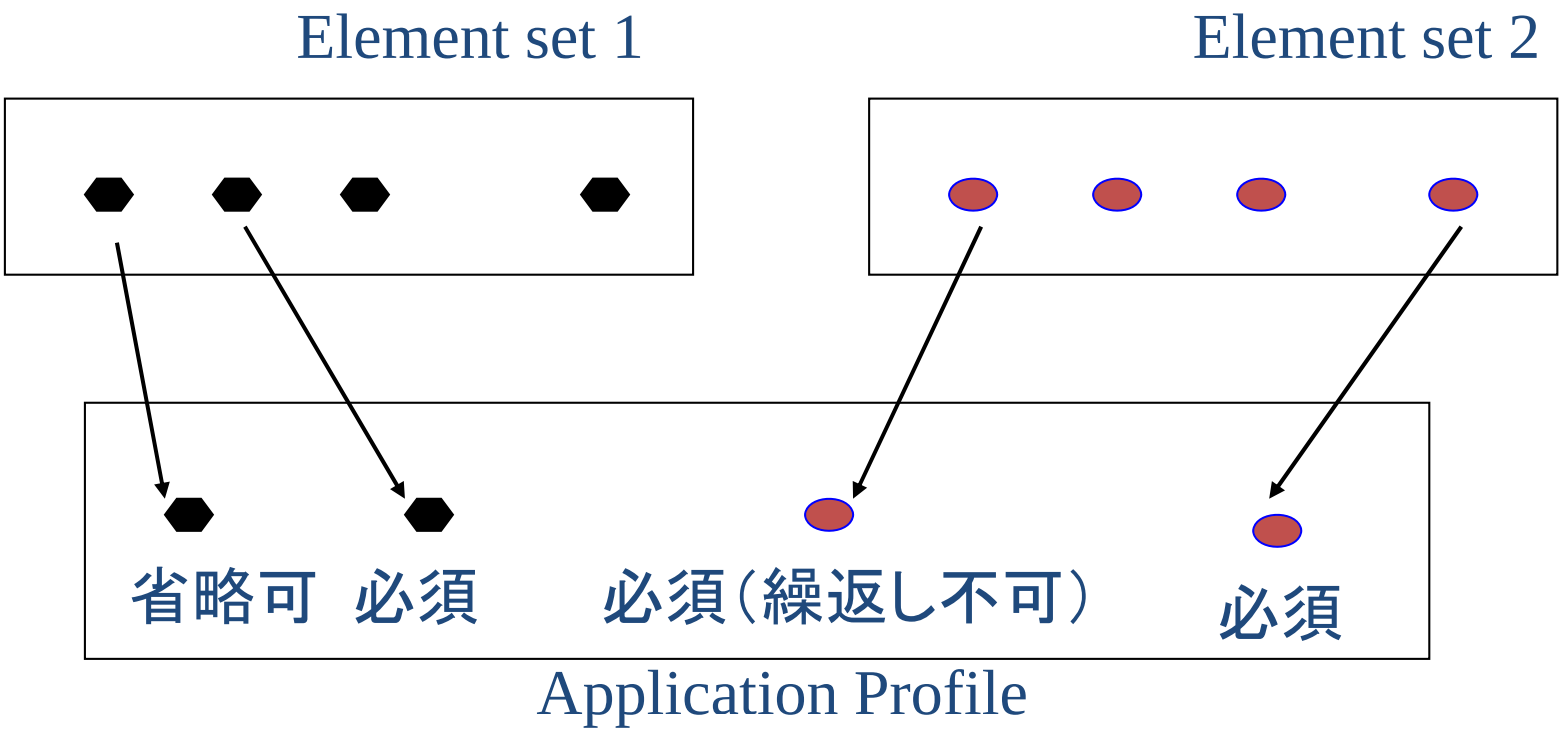
- メタデータスキーマには、属性（記述項目）として用いる語と属性値の型や記述形式を表す語の集合が含まれる。これを**メタデータ語彙**という
- DCMESやdctermsはメタデータ語彙である。
  - 換言すると、DCMESやdctermsだけが与えられてもメタデータは書けない
- 応用毎に、そこで用いるメタデータ語彙を決める必要はなく、既存のメタデータ語彙から適切な語を取り出して、応用用のメタデータ語彙を作ればよい
- そして、応用用メタデータ語彙の上に、応用のための構造制約や記述形式等を定義すればよい。こうしてメタデータ語彙の定義から切り離して定義される応用用のスキーマを**Application Profile**と呼ぶ

エレメントセットとアプリケーションプロファイル

- メタデータスキーマは、
  - 属性を表すための語の集まり(属性の語彙)、属性値の形式や統制語彙を表す語の集まり(属性値の語彙)
  - 構造的な制約の定義

  を含む。
- 前者をメタデータの語彙(あるいはElement Set)、後者をアプリケーションプロファイル(Application Profile)と呼ぶことにする。

エレメントセットとアプリケーションプロファイル



いくつかのエレメントセットから応用毎の適用規則を作る。

Metadata Schema A

Title	Mixing and Matching Metadata Schema	
Creator	**Name (FN)**	Sugimoto, Shigeo
	Affiliation (ORG)	Univ. Tsukuba
	Affiliation Unit (ORGUNIT)	Graduate School of Library, Information and Media Studies
Subject	Metadata Schema, Interoperability	
Date	2007-9-4	
Language	en	
Presentation Location	Bangalore, India	

Metadata Schema B

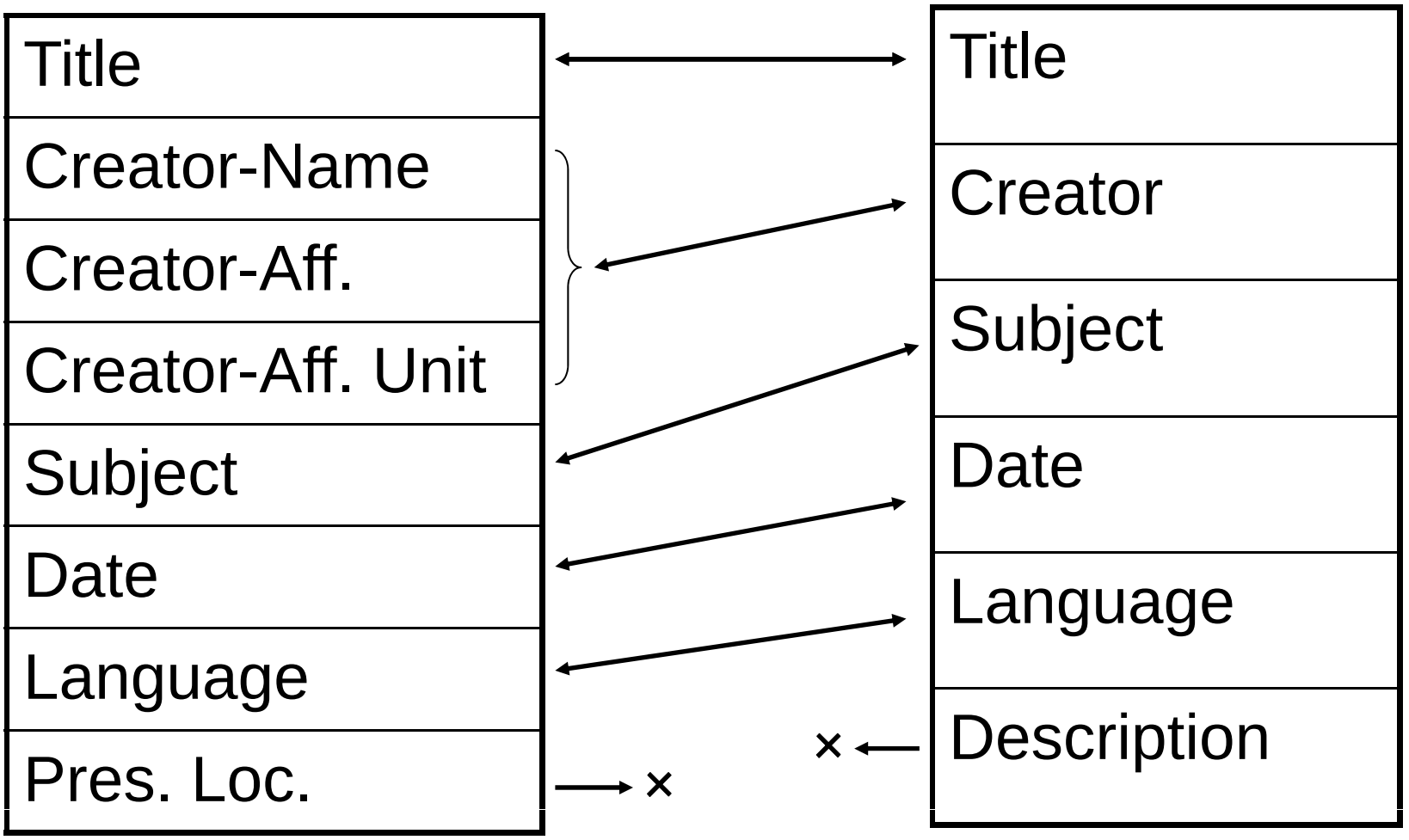
Title	Mixing and Matching Metadata Schema
Creator	Shigeo Sugimoto. Univ. Tsukuba, Grad. School of Library, Information and Media Studies
Subject	Metadata Schema, Interoperability
Date	04/09/07
Language	English
Description	Metadata schema interoperability across domains and adaptability …

スキーマAとBの違い

Title	Mixing a
Creator	**Name (FN**
	Affiliation
	Affiliation (ORGUNI
Subject	Metadata S
Date	2007-9-4
Language	en
Presentation Location	Bangalore

Title	Mixing and Matching Metadata Schem
Creator	Shigeo Sugimoto. Univ. Tsukuba, Grad. Sc Library, Information and Media Studies
Subject	Metadata Schema, Interoperability
Date	04/09/07
Language	English
Description	Metadata schema interoperability acro domains and adaptability …

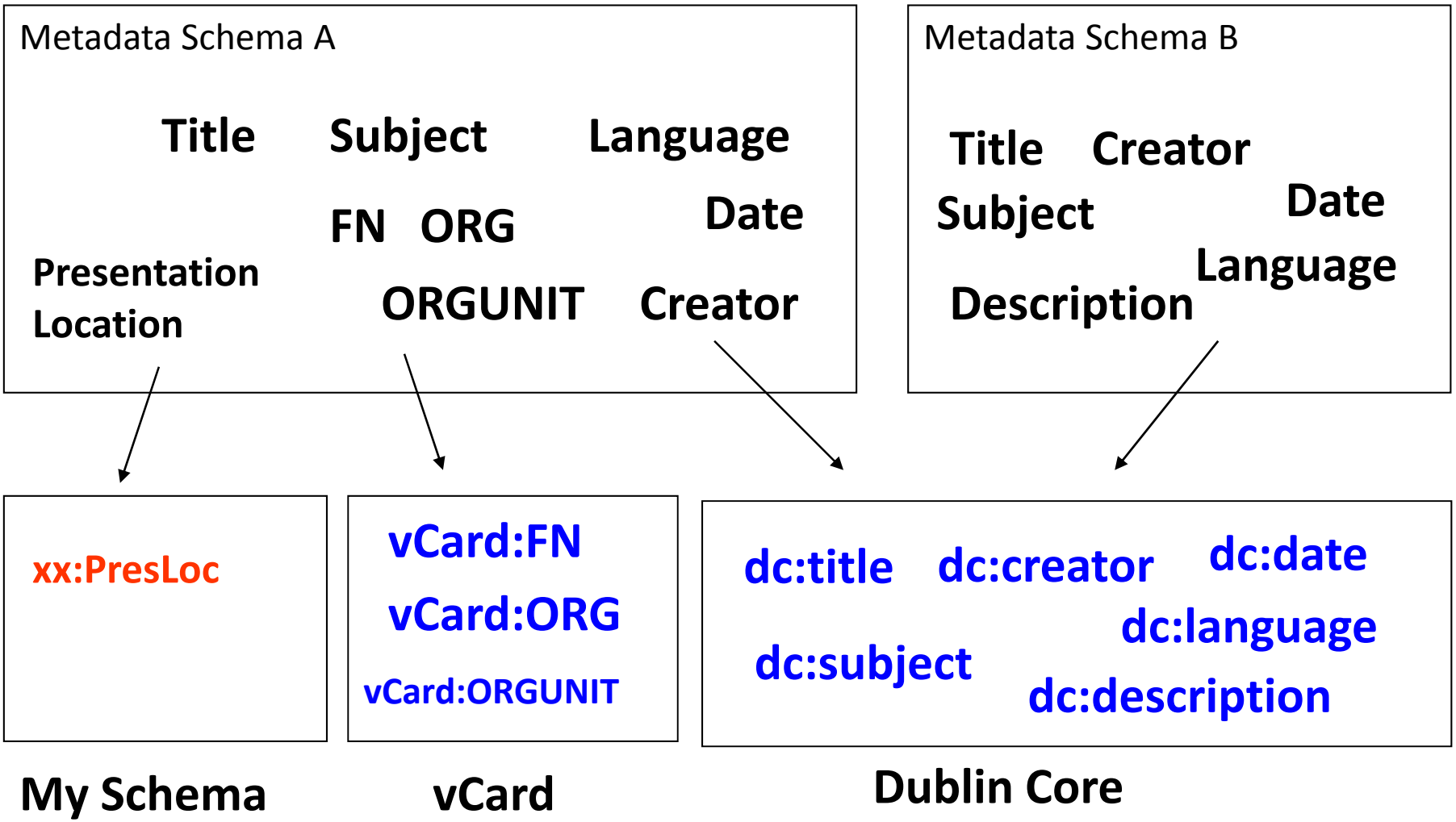
スキーマA と Bの違い


Title
Creator-Name
Creator-Aff.
Creator-Aff. Unit
Subject
Date
Language
Pres. Loc.
×
Title
Creator
Subject
Date
Language
Description
×

スキーマA と Bを対応付けて見てみる

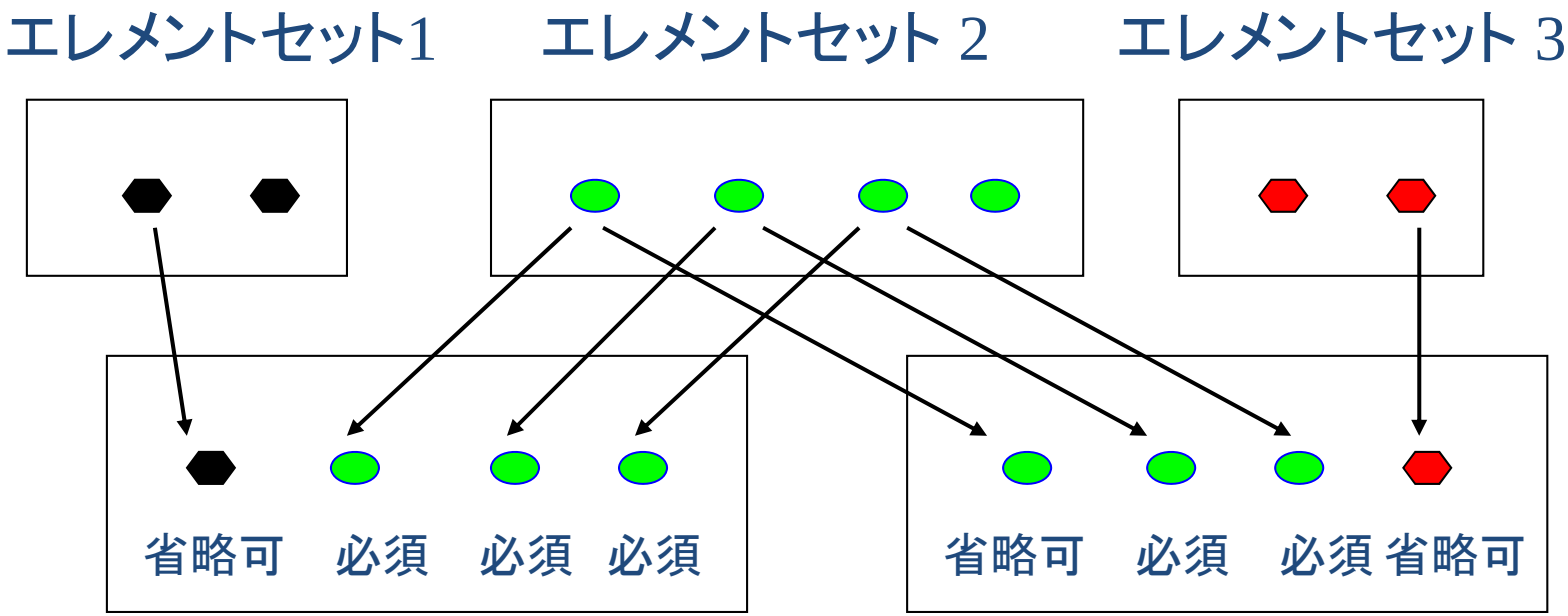
- AにもBにも現れるエレメントがある
  - たとえばTitleエレメント
  - これらの意味は本当に同じか?
    - ここでは同じと仮定する
    - 機械的に同じと判定するにはURIのような識別子が必要
- AのCreator エレメントは A サブエレメントを持つ
  - AとB のマッチングにはダムダウン（Dum-Down, 簡単化）が必要
- 値の記述の仕方が異なる

スキーマとエレメントセット



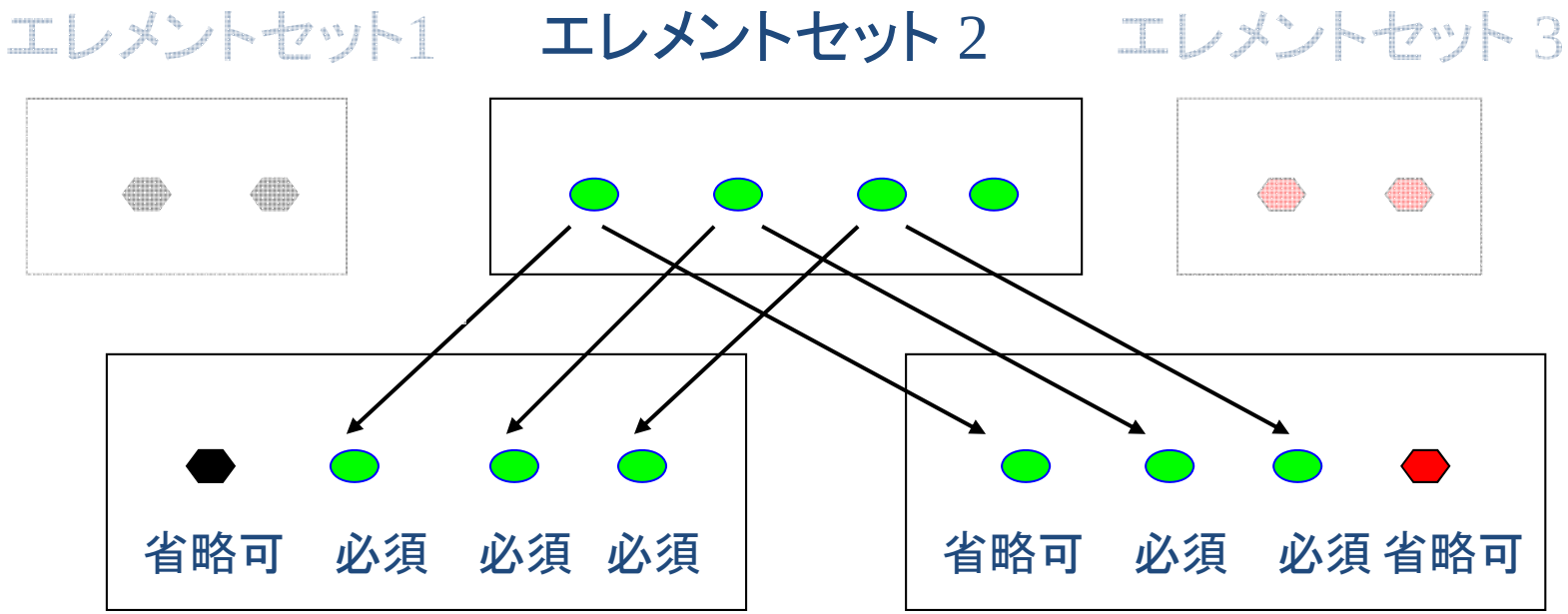
エレメントセットとアプリケーションプロファイル

- メタデータを構成するエレメントとそれらを組み合わせて作る目的向けのメタデータの構造を分離してとらえるもの



エレメントセットとアプリケーションプロファイル

- 二つのスキーマで、それぞれが独自の記述項目を定義する場合に比べて、共通部分が見えやすくなることがわかる



エレメントセットとアプリケーションプロファイル

- Simple Dublin Coreは、DCMESで決められたエレメントの中から15のエレメントを取り出し、その上に、すべての要素が省略可能かつ繰り返し可能という条件を与えて作ったアプリケーションプロファイルとみなすことができる。
- Application Profileはメタデータの相互運用性の向上に有効
  - 異なるスキーマ同士を関係付けるとき、記述項目間での意味的なマッピングをする必要があるが、Application Profileの概念に基づいて既存の項目を用いてスキーマを定義すれば、マッピングを行わねばならないところを減らすことができる
  - 記述項目間の意味的な関連付けを特定の応用スキーマに限定することなく与えることができる

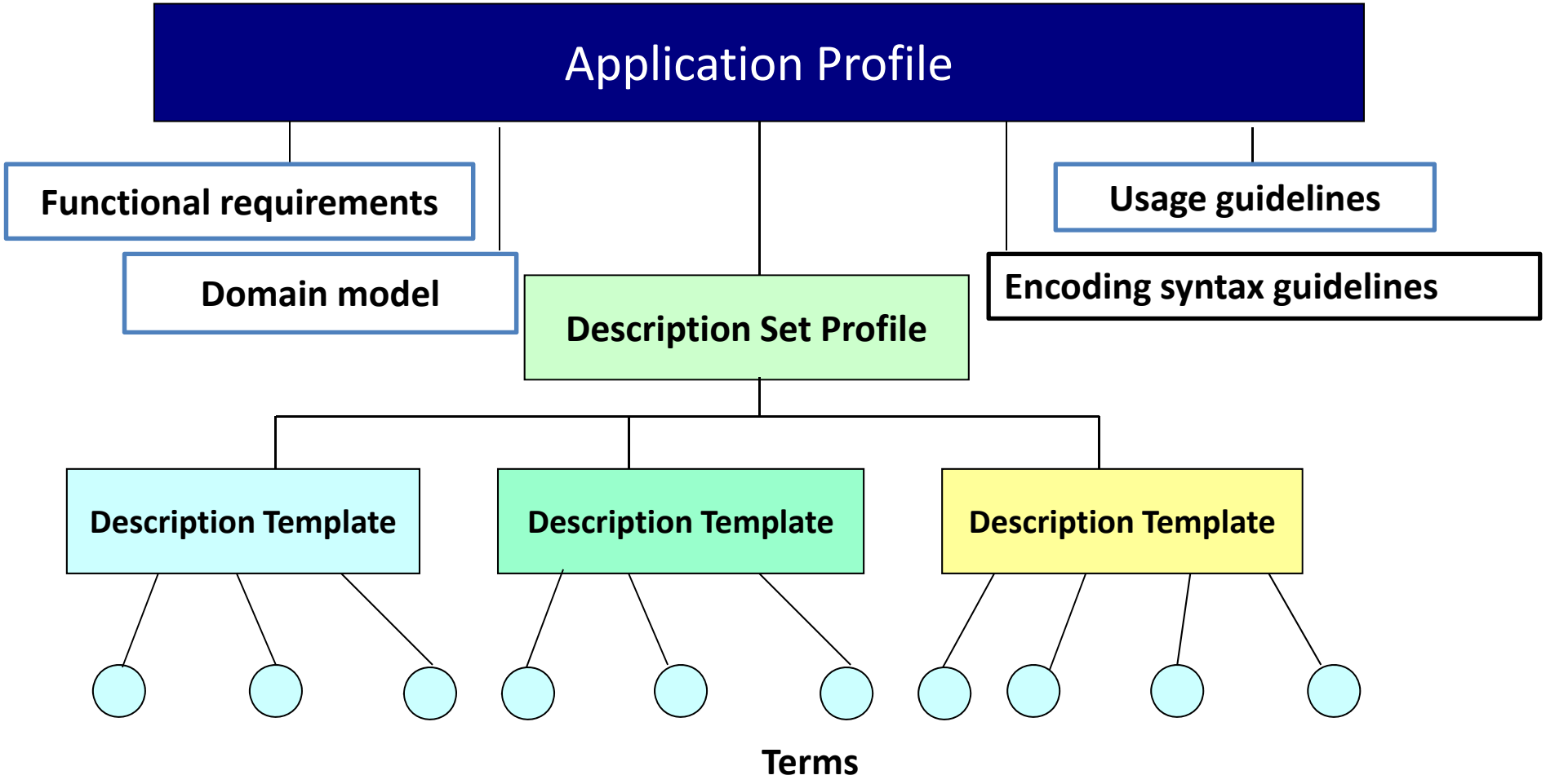
エレメントセットとアプリケーションプロファイル

- 「独自のものを作ることはせず、できるだけアリモノを使い、組み合わせてスキーマを決めよう！」
  - Mixing and Matchingの精神
- 次に述べるSingapore FrameworkはApplication Profileの構成要素を決めている

Singapore Framework Application Profileの構成要素

1. 機能要求 (Functional Requirements)　—必須
   – 開発するアプリケーションで達成したいことの定義
2. ドメインモデル (Domain Model)　—必須
   – 記述対象を定め、そのメタデータや関連を定義
3. **Description Set Profile (DSP)**　—必須
   – **メタデータの構造的制約を定める**
4. 利用ガイド (Usage Guidelines)　—省略可能
   – メタデータ作成者向けのガイド
5. エンコーディングの定義 (Encoding Syntax Guidelines)　—省略可能
   – データをどのような(機械可読な)書式にするかを決める

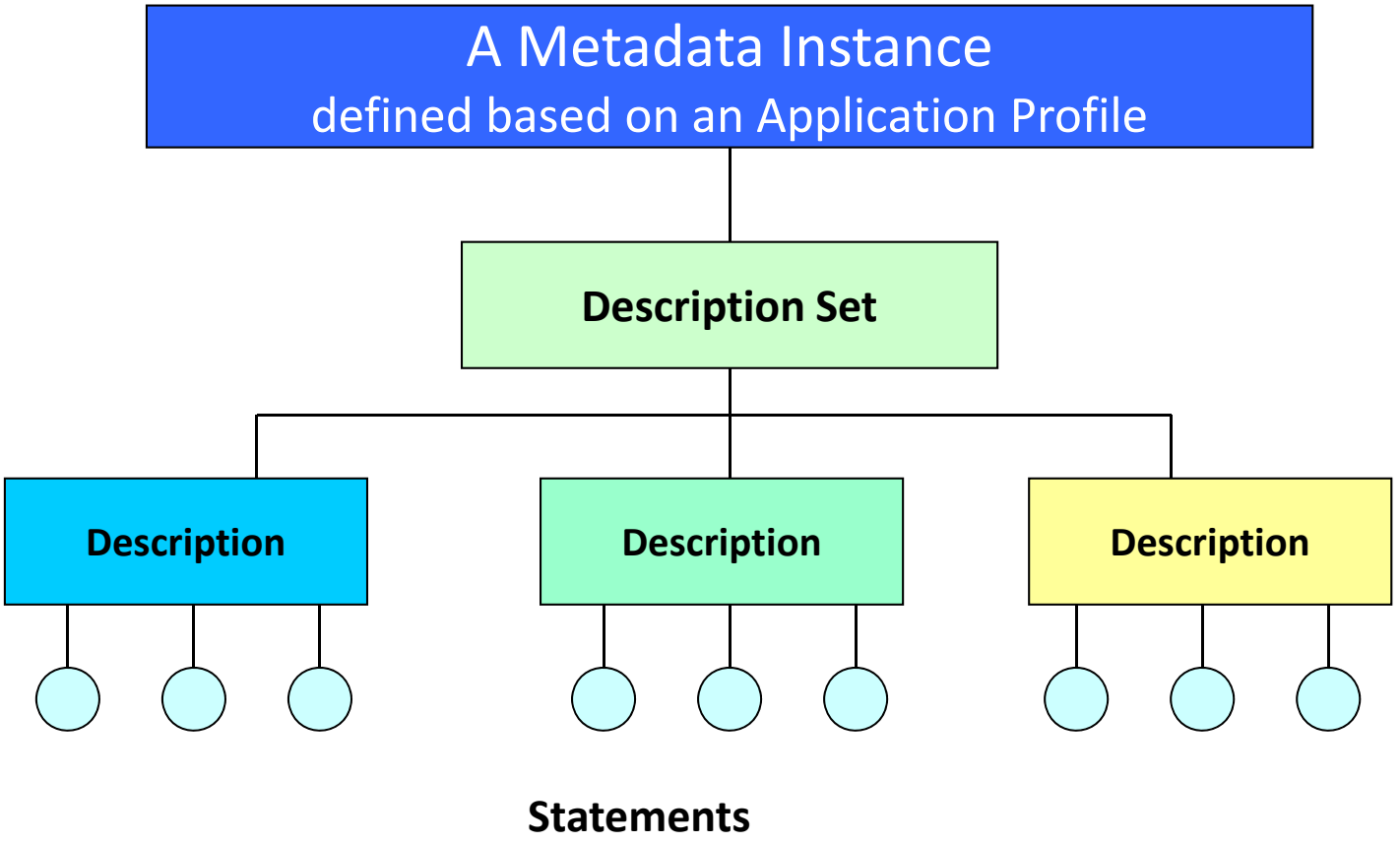
Singapore Framework - A Simplified Structural View of a an Application Profile



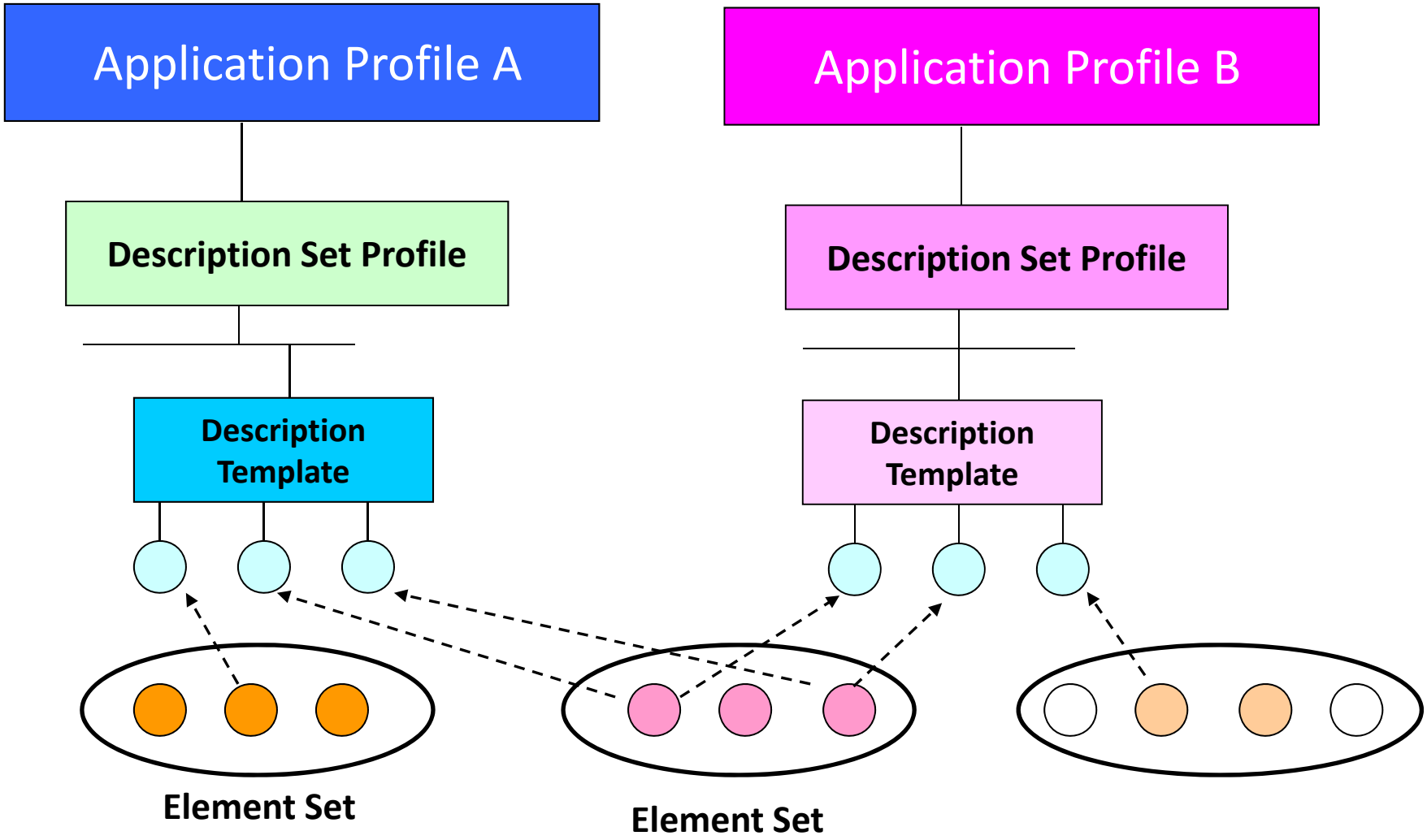
Description Set Profile

- Description Set Profile
  - メタデータの構造を決める
  - Description Templateの集まり
- A Description Template は構造的制約と値の制約に基づいて作られた属性集合
  - 構造的制約:　必須、省略可、繰返し回数など
  - 値の制約:属性に応じて決まる値のクラス(リソースの種類、文字列の形式、ある範囲の数値、統制語彙の種類など)

Application Profile Concept



Application Profile Concept



3. メタデータスキーマについて
メタデータスキーマを分解する(1/5)

(1) 対象のどのような属性について書くのか ⇒ 属性の集合

- 例、タイトル、作成者、内容の説明

(2) 属性毎の記述制約

- 記述しなければならない(必須、Mandatory)
- 記述することが強く求められる(推奨、Recommended)
- 記述しなくてもかまわない(省略可能、Optional)
- 繰返してもかまわない(0回以上、1回以上)、繰返し回数が決められる(最大、最小)

メタデータスキーマについて
メタデータスキーマを分解する(2/5)

(3) 属性毎の値の書き方

- 自由形式で書いてかまわない
- 自由形式であるが、長さに制約がある（何文字以内、何文字以上、何文字以下）
- あらかじめ決められた語彙の範囲で書かねばならない
- あらかじめ決められた形式で書かねばならない
  - 例：日付の書き方（2011-4-16、16/04/11、16/04/11、平成23年4月16日）
- 決められた言語で書かねばならない。日本語、英語
- 決められた文字の範囲で書かねばならない。
  - 常用漢字、英数字のみ、JIS第1・2水準文字

メタデータスキーマについて
メタデータスキーマを分解する(3/5)

(4) メタデータをどのような形で実現するのか

- コンピュータ上での表現と人間が読み書きするための表現
- コンピュータ上での蓄積のための表現とコンピュータ間でのデータ交換のための表現
  - 文字セット。たとえば、UnicodeでUTF8
  - データベース管理システム
  - データ交換のための形式

メタデータスキーマについて
メタデータスキーマを分解する(4/5)

(5) 属性値の取出しかた、書き方

- 対象に合わせて、かつ属性の意味と合った内容をどのように取り出すか
  - たとえば、Webページのタイトルは
    - Titleタグでかかれたものか？
    - 画面上に現れたもっともタイトルらしい文字列か？
    - メタデータ作成者が決めるのか？

メタデータスキーマについて
メタデータスキーマを分解する(5/5)

- (1)～(4)は実現上の形式を決める
  - (1)～(3)は物理的な実現形式ではなく、論理的な形式を決める
    - 特定のシステムに依存するものではない
  - (4)はメタデータを実現するために利用する実際の道具や環境によって決まる
    - システムとして実現する際に決めなければならないこと
- (5)はメタデータを作る際に記述対象から抽出すべき内容（とその取り出し方）を決める

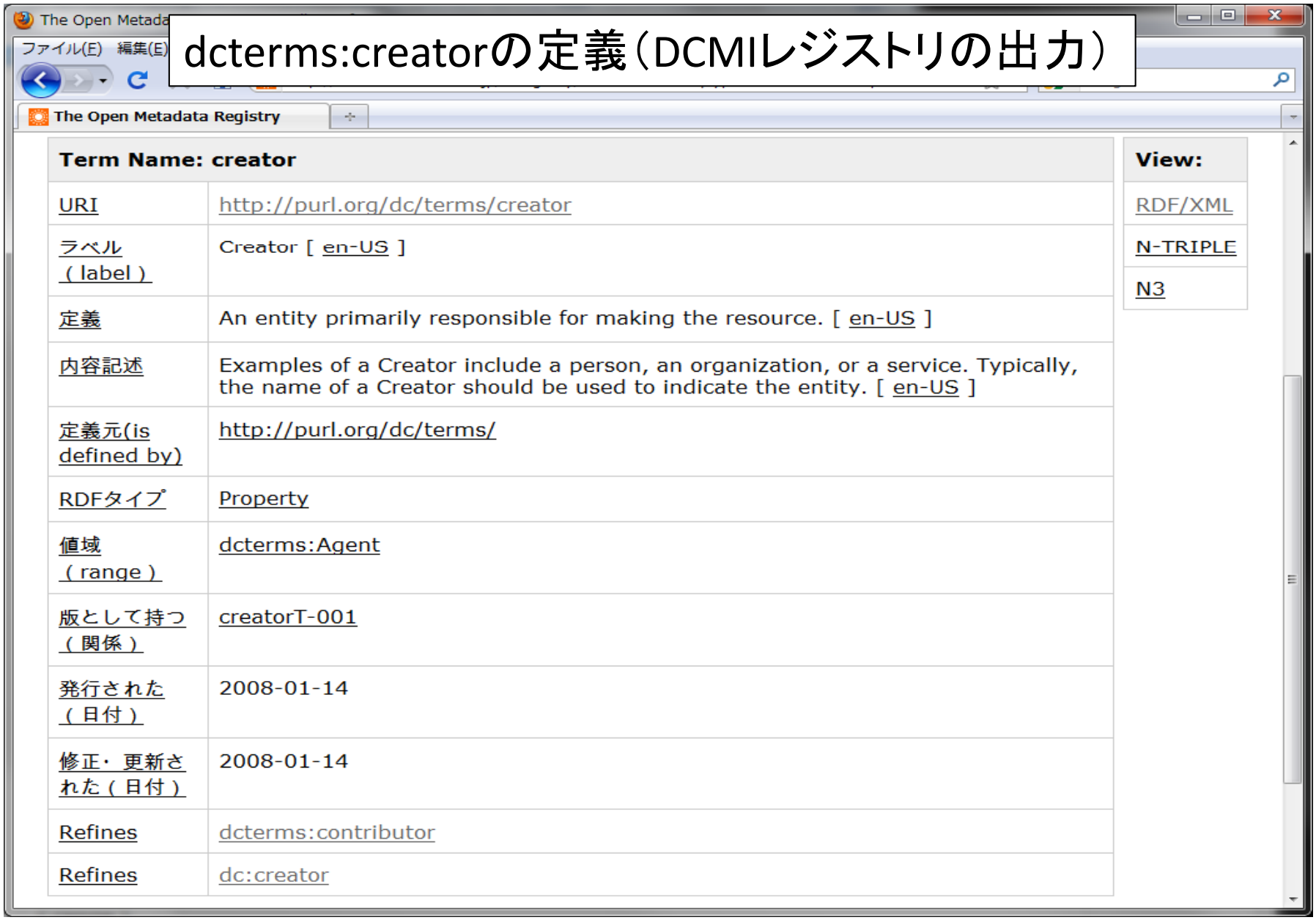
メタデータスキーマについて
メタデータ語彙

- メタデータスキーマの基盤
  - 意味の基盤（Semantics）　ーーー　メタデータ語彙
    - 記述項目の名前とその意味の定義
    - 属性値の型の名前とその意味の定義
      - 属性値の表現形式の定義
    - 属性値に用いる統制語彙の定義
  - 構造の基盤
    - 必須、省略可能、繰返し回数といった構造制約の定義
    - データの表現モデル
    - 具体的表現形式（記述形式、実現定式）の定義

メタデータスキーマの基盤
「ことば（概念）」を定義する

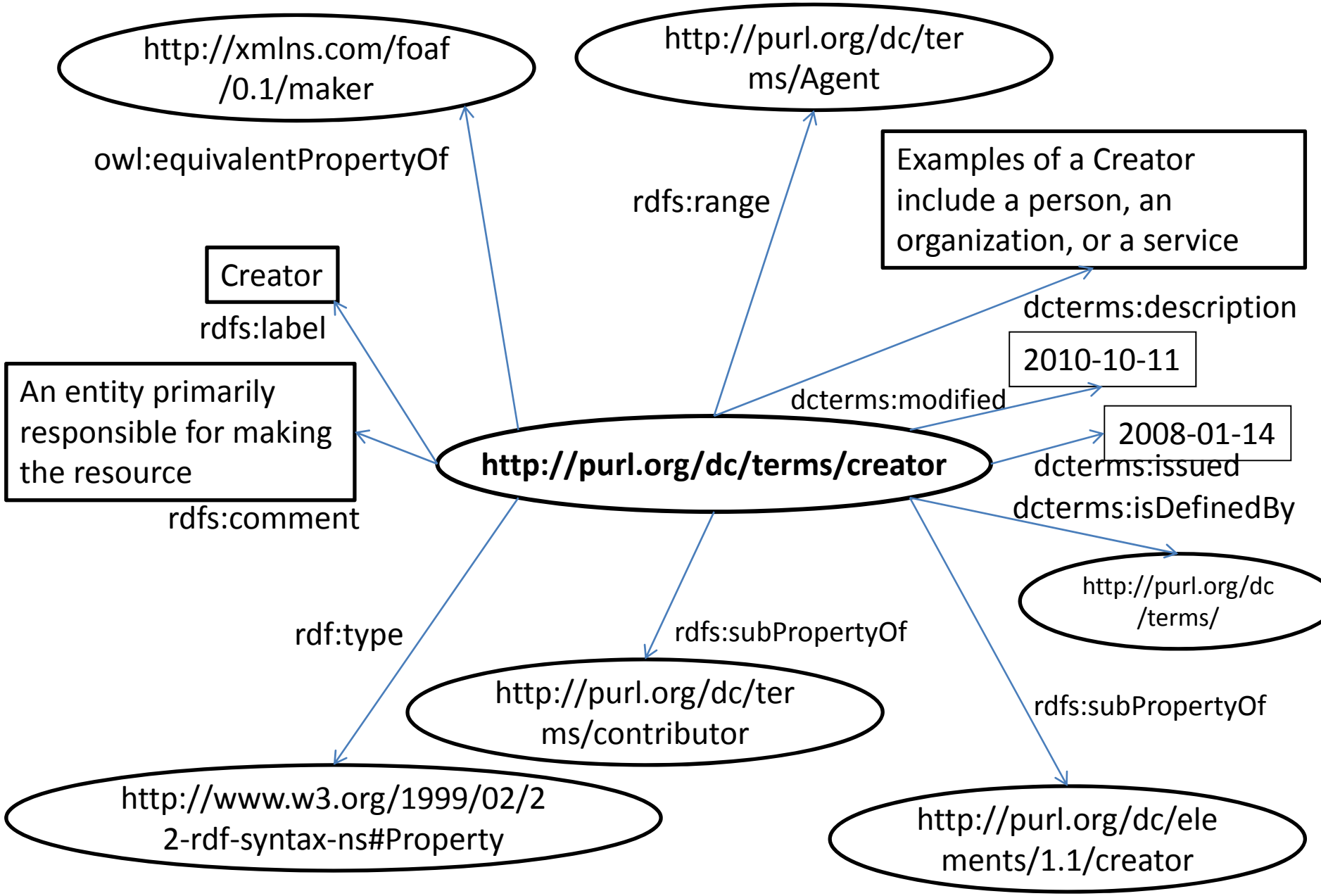
- メタデータの基本はことば（語）
  - 属性をあらわす語
  - 属性値を表すために用意された語（型、統制語）
- 属性、属性値の型や統制語彙等を、コンピュータで機械的に扱えるように形式的に定義する
  - RDF Vocabulary Description Language (RDF Schema)
  - RDFではPropertyやClassとして定義する
- メタデータスキーマを決めるには、属性や属性値の型を表す語句の定義が必要
  - Property定義の例： dcterms:creator
  - Class定義の例： dcterms:agentの定義

dcterms:creatorの定義（DCMIレジストリの出力）

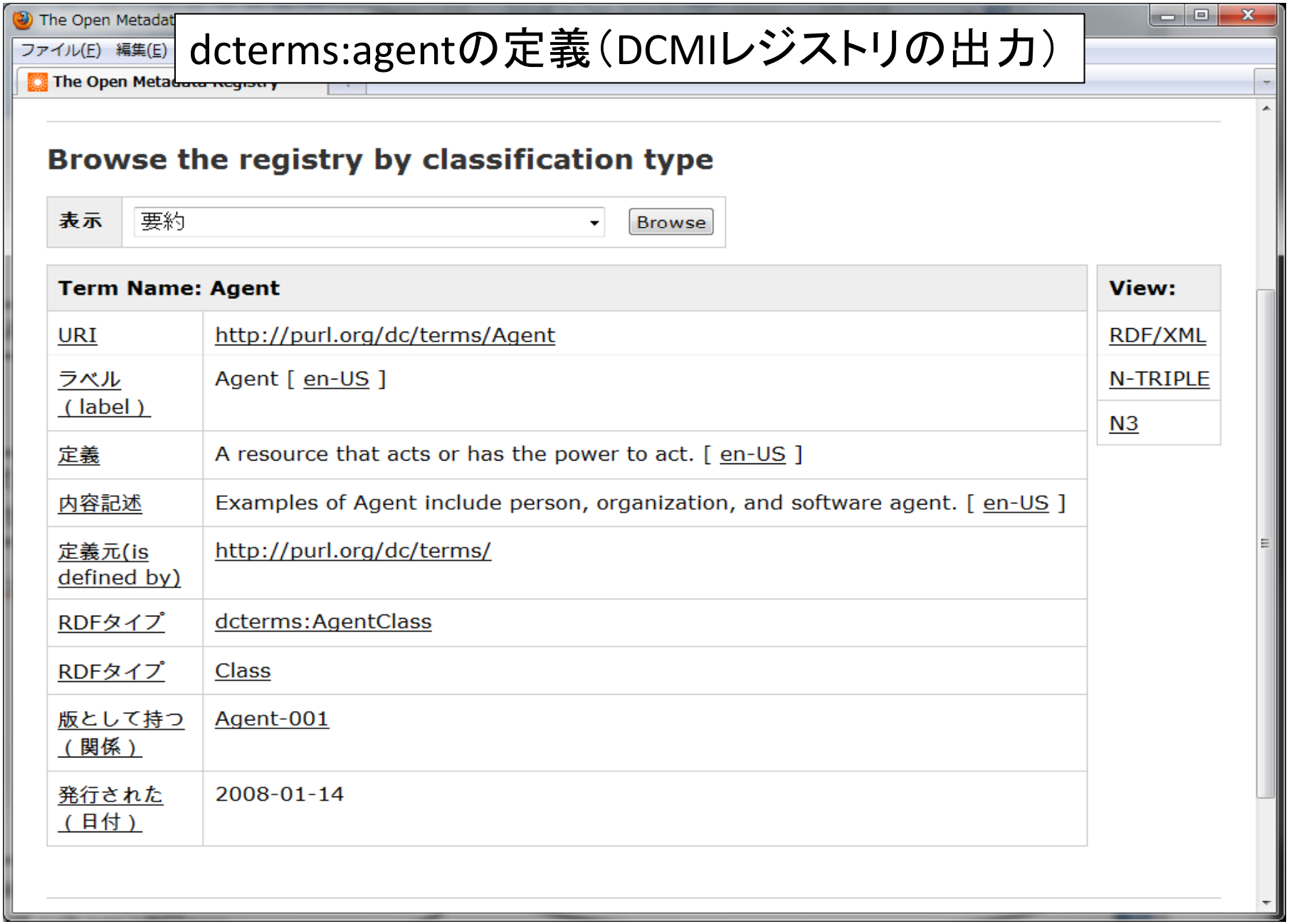


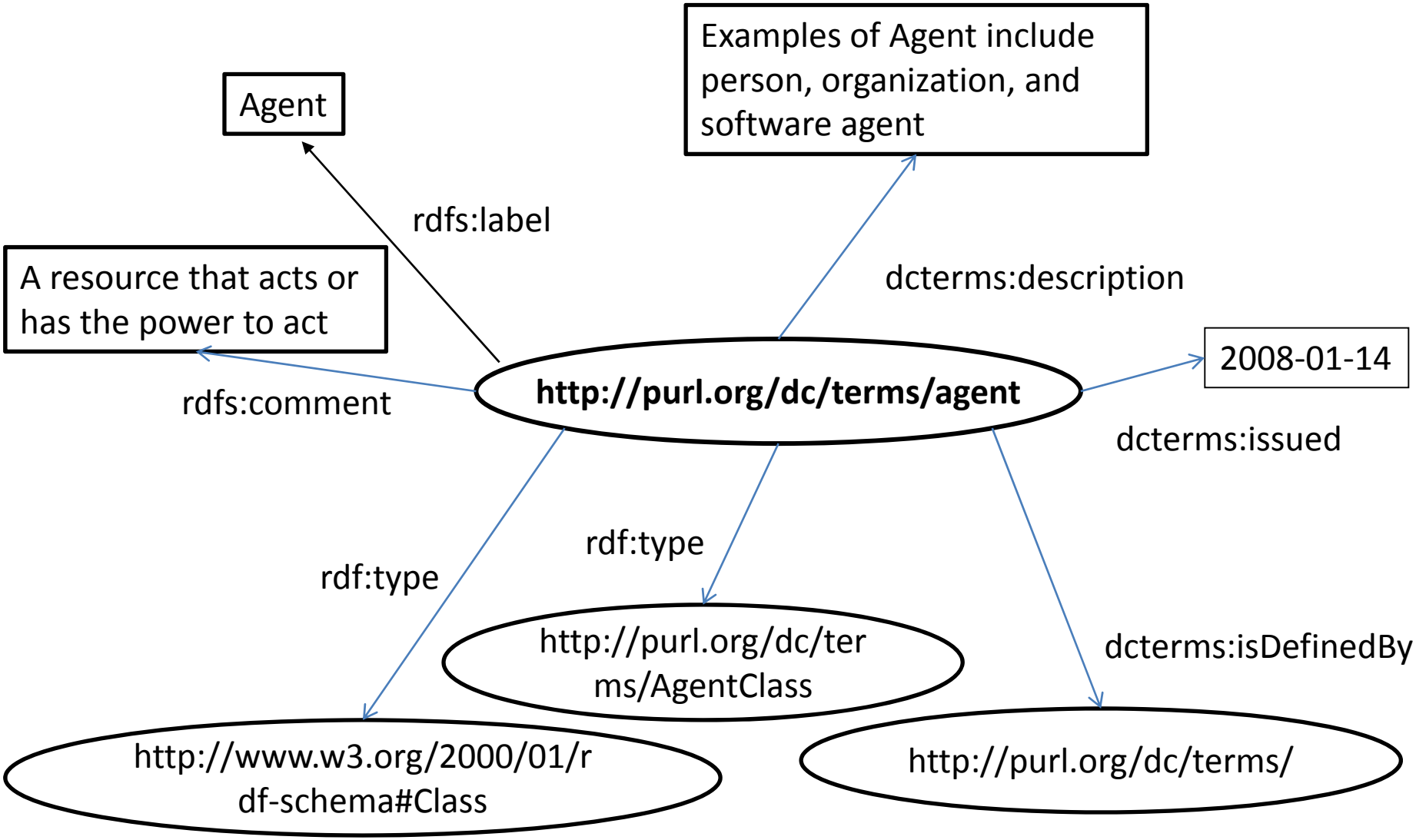
Term Name: creator	
URI	http://purl.org/dc/terms/creator
ラベル（label）	Creator [en-US]
定義	An entity primarily responsible for making the resource. [en-US]
内容記述	Examples of a Creator include a person, an organization, or a service. Typically, the name of a Creator should be used to indicate the entity. [en-US]
定義元(is defined by)	http://purl.org/dc/terms/
RDFタイプ	Property
値域（range）	dcterms:Agent
版として持つ（関係）	creatorT-001
発行された（日付）	2008-01-14
修正・更新された（日付）	2008-01-14
Refines	dcterms:contributor
Refines	dc:creator

View:
RDF/XML
N-TRIPLE
N3


http://xmlns.com/foaf/0.1/maker
owl:equivalentPropertyOf
http://purl.org/dc/terms/Agent
rdfs:range
Examples of a Creator include a person, an organization, or a service
dcterms:description
Creator
rdfs:label
2010-10-11
dcterms:modified
An entity primarily responsible for making the resource
rdfs:comment
http://purl.org/dc/terms/creator
2008-01-14
dcterms:issued
dcterms:isDefinedBy
http://purl.org/dc/terms/
rdf:type
rdfs:subPropertyOf
http://purl.org/dc/terms/contributor
rdfs:subPropertyOf
http://www.w3.org/1999/02/22-rdf-syntax-ns#Property
http://purl.org/dc/elements/1.1/creator

dcterms:agentの定義（DCMIレジストリの出力）



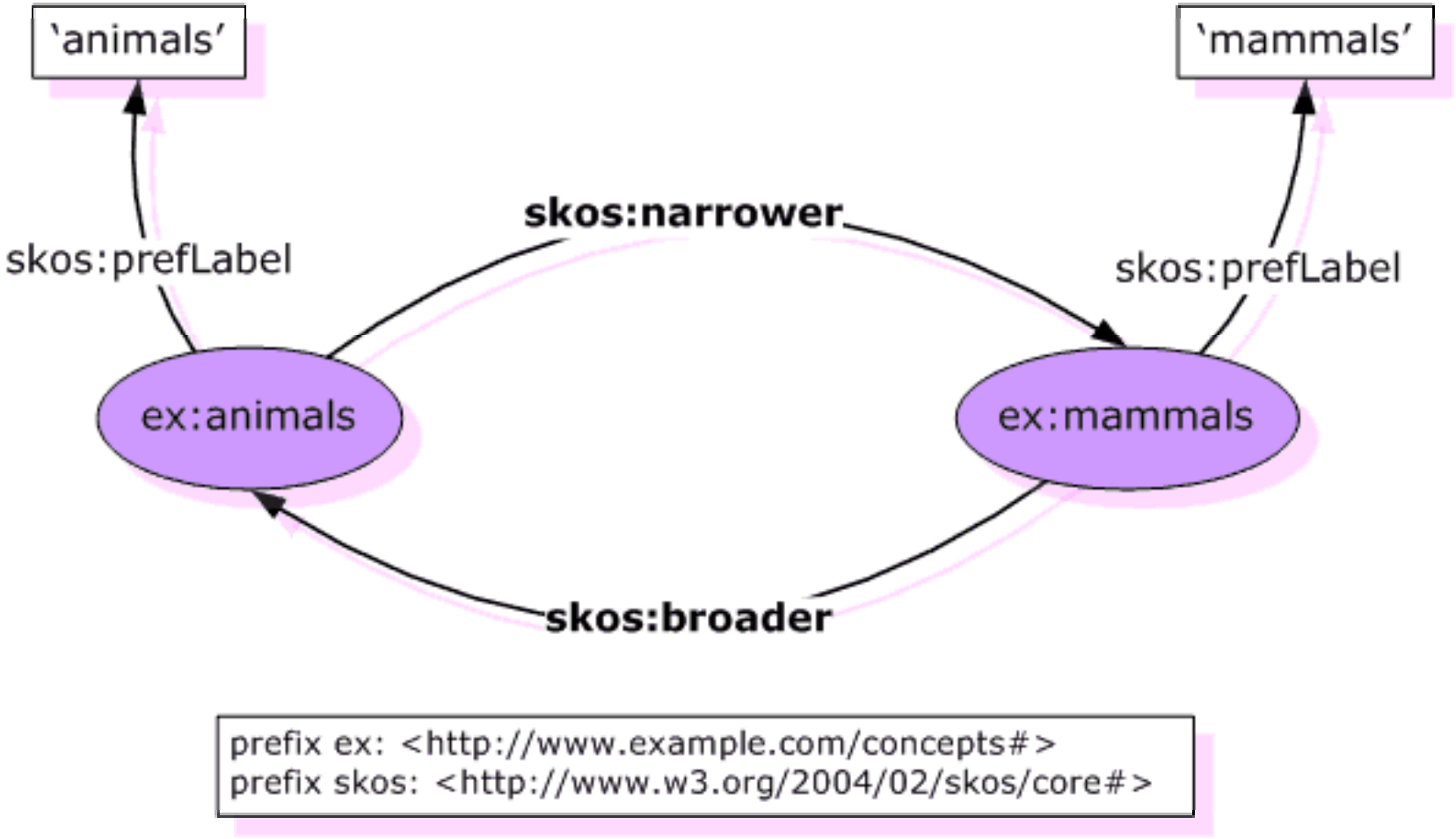

Agent
Examples of Agent include person, organization, and software agent
rdfs:label
dcterms:description
A resource that acts or has the power to act
2008-01-14
rdfs:comment
http://purl.org/dc/terms/agent
dcterms:issued
rdf:type
rdf:type
http://purl.org/dc/terms/AgentClass
dcterms:isDefinedBy
http://www.w3.org/2000/01/rdf-schema#Class
http://purl.org/dc/terms/

SKOS: Simple Knowledge Organization System

- 分類や件名標目表、シソーラスのようにシンプルな構造で作られた語彙を表すために、RDF上に定義された語彙の表現形式
  - SKOS homepage http://www.w3.org/2004/02/skos/
- SKOS Core vocabulary
  - SKOSで決めた、語彙表現のために用いる属性集合
  - http://www.w3.org/TR/2005/WD-swbp-skos-core-guide-20051102/
  - **SKOS Core Vocabulary Specification** http://www.w3.org/TR/2005/WD-swbp-skos-core-spec-20051102/
- SKOSのReference（Working Draft）
  - http://www.w3.org/TR/2008/WD-skos-reference-20080829/

SKOS: Simple Knowledge Organization System

http://www.w3.org/TR/2005/WD-swbp-skos-core-guide-20051102/ より



SKOS: Simple Knowledge Organization System

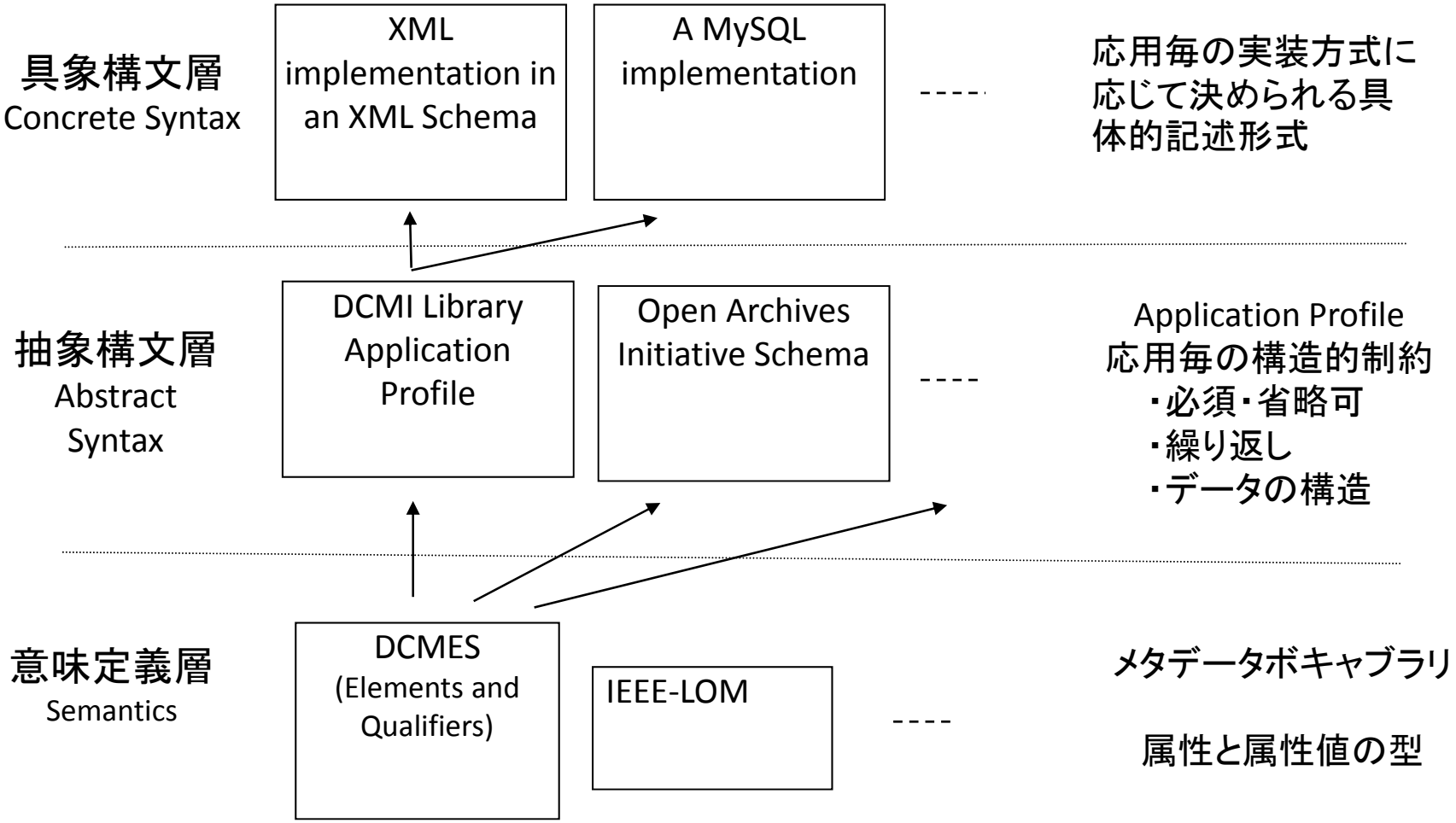
http://www.w3.org/TR/2005/WD-swbp-skos-core-guide-20051102/ より

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:skos="http://www.w3.org/2004/02/skos/core#">
 <skos:Concept
       rdf:about="http://www.example.com/concepts#mammals">
      <skos:prefLabel>mammals</skos:prefLabel>
     <skos:altLabel   xml:ang="ja">哺乳動物</skos:altLabel>
      <skos:broader
          rdf:resource="http://www.example.com/concepts#animals"/>
  </skos:Concept>
  <skos:Concept
          rdf:about="http://www.example.com/concepts#animals">
       <skos:prefLabel>animals</skos:prefLabel>
       <skos:narrower
            rdf:resource="http://www.example.com/concepts#mammals"/>
     </skos:Concept>
</rdf:RDF>
```

メタデータスキーマを階層的に見る

- スキーマの構成要素を階層的にとらえてみると
  - メタデータボキャブラリの定義
    - 記述項目の定義：属性のボキャブラリ
      - 情報資源の属性，メタデータエレメント
    - 属性値の型、形式：属性値のボキャブラリ
  - 構造的制約
    - 必須・省略可、繰返し条件、値の構造など、値の構造的制約
    - 抽象構文
  - 具象的表現形式
    - HTML, XML, 特定のデータベースでのスキーマなど
    - 具象構文

メタデータスキーマの階層モデル[1]



[1] Nagamori, M., Sugimoto, S., A Metadata schema framework for functional extension of metadata schema registry, Proc. DC2004, Shanghai, 2004

メタデータを階層的に見る

- 相互運用性のレベルが見やすくなる
  - 具象構文まで同じであれば、相互運用可能
  - 抽象構文まで同じであれば、具体的な表現形式をあわせばよい
  - 同じメタデータ語彙を利用していれば、語彙のレベルでのマッピング（すなわち、意味的なマッピング）は行わずにすむ。
  - メタデータ語彙が異なっていれば、記述項目同士のマッピングからはじめなければならない

4. おわりに
Dublin Coreのこころ

- メタデータスキーマの概念モデルを作上げてきた
  - Application Profileの概念
    - Mixing and Matching
    - アリモノを使おう
      - 「車輪の再発明」のようなことはしてはならない
- リソースの発見と記述のために、分野にまたがって使うことのできる記述項目（属性）の語彙を作り上げてきた
- メタデータはことば（＝概念）の定義が基本
  - メタデータの流通性を高めるには、共通の語彙を使うことが重要
  - Linked Open Dataの活動、Library Linked Dataの活動

おわりに

将来に向けて

- メタデータはネット上での情報流通の要
- メタデータの相互運用性を高めるためのインフラの必要性
  - 相互運用性の向上と応用毎への特化の要求の両方を満たすための基盤
- 相互運用性の向上のためのメタデータスキーマの共有
  - メタデータの基盤となる「メタデータ語彙」の共有
  - 応用毎に作られる規則の共有と再利用の推進

おわりに
将来に向けて

- メタデータ情報基盤構築事業
  - 平成22年度総務省新ICT利活用サービス創出事業による事業（http://www.meta-proj.jp/）
    - 筑波大学、インフォコム株式会社ほか
  - メタデータスキーマを収集、蓄積、共有するための基盤となるシステム（Meta Bridge）の構築
    - 本年度から筑波大学・知的コミュニティ基盤研究センターを中心に運営
    - メタデータスキーマの共有・利用のための活動の推進の継続
- メタデータ情報基盤研究会
  - 筑波大学・知的コミュニティ基盤研究センターで組織
  - メタデータスキーマの収集、蓄積、共有のための基盤に関する意見交換、知識共有、上記事業等への助言
  - 国立国会図書館、国立公文書館、国立情報学研究所、東京国立博物館、国立近代美術館、凸版印刷ほかからの参加をいただく
  - 産官学民でのメタデータに関する情報共有を進めたい

おわりに

将来に向けて

- 最後に、図書の目録等について思うこと
  - 図書館等の目録は組織だって作られてきた、長い歴史を持つメタデータ
  - それぞれのコミュニティの中でニーズにあった発展を遂げてきた
  - その一方、ネットワーク情報環境の高度な発展による利用者の行動パターンの変化と、出版・情報発信メディアの進化による情報獲得（本の探し方、読み方）の変化についていなければならない
  - こうした目録等は、情報の組織化のための知識を具現化してきたものであり、そうした知識をより広い範囲で利用できるようにしていかねばならない