2018年5月10日
＠東大病院企画情報運営部

「法造」を用いた オントロジー構築入門

大阪電気通信大学
情報通信工学部　情報工学科
古崎 晃司
kozaki@osakac.ac.jp

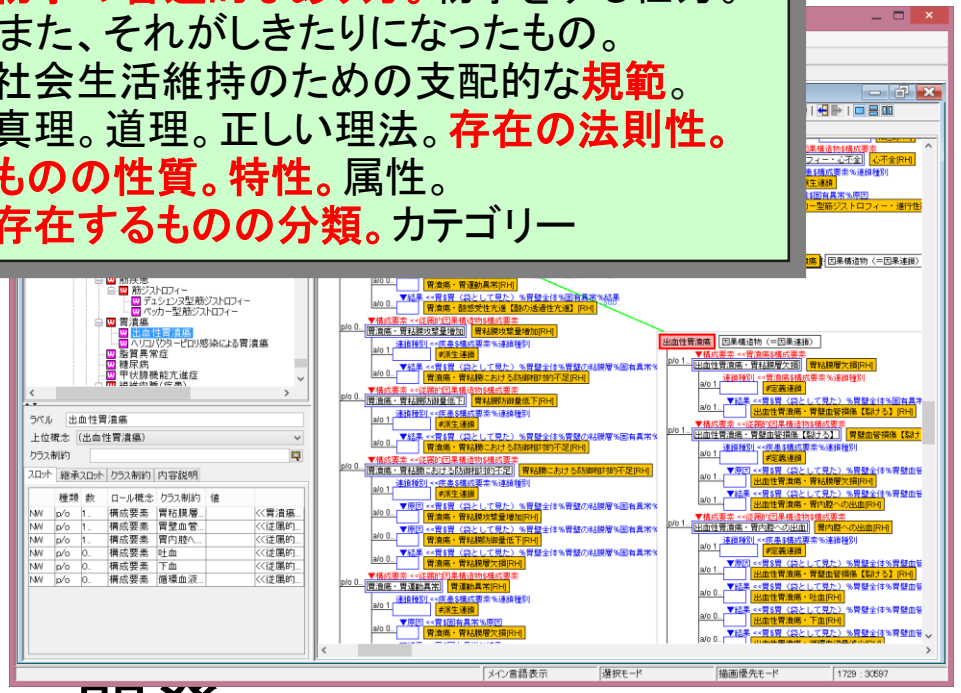
概要

- **ねらい**
  - 「法造」を用いたオントロジー構築を例にして，オントロジー工学における概念化の基本的な考え方を示す．
- **資料の内容**
  - オントロジー工学の基礎
    - オントロジーの定義，基本思想
  - オントロジーの構築方法
    - ①is-a階層・概念定義の基本的な考え方
    - ②詳細な概念定義（ロール概念・関係概念の利用）
  - オントロジーの構築・概念化の事例紹介
    - ①一般的な概念化の事例
    - ②臨床医学オントロジーにおける事例

オントロジー構築利用環境「法造」

- オントロジー工学の基礎理論に基づいて開発された，**オントロジー（＝法）の構築（＝造）および利用**を支援するソフトウェア
- 1996年より開発
- http://www.hozo.jpにて**フリーソフトウェアとして公開**
  - 登録ユーザ数：約5,000
  - ダウンロード数：約12,000
    （共に，国内外含む）
- 研究室内外のオントロジー開発プロジェクトでの多数の利用実績

ほう【法】（『広辞苑』より引用）
- **物事の普遍的なあり方。**物事をする仕方。また、それがしきたりになったもの。
- 社会生活維持のための支配的な**規範**。
- 真理。道理。正しい理法。**存在の法則性。**
- **ものの性質。特性。**属性。
- **存在するものの分類。**カテゴリー



法造の画面例

「法造」準備

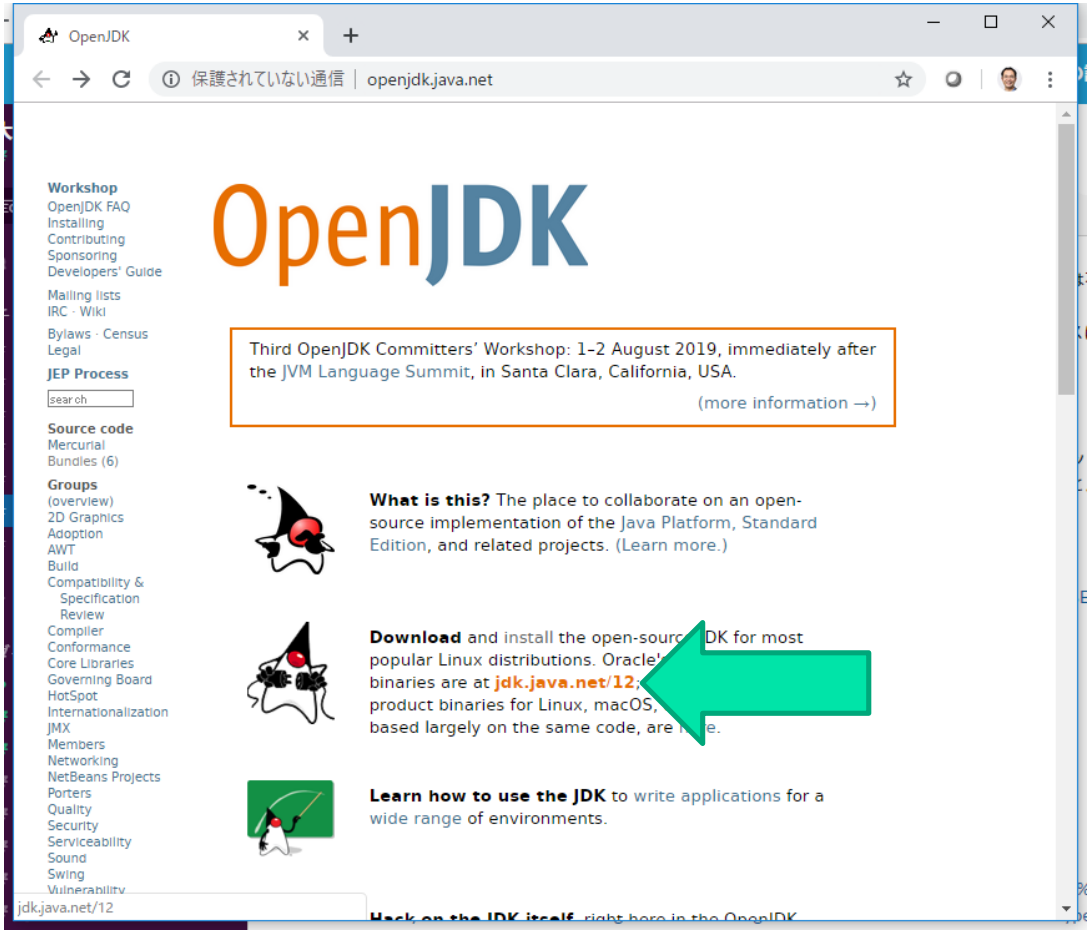
- Javaのインストール確認
- 「法造」のインストール
- サンプルファイルのダウロード

Javaのインストール確認

- **Javaのインストール確認**
  - コマンドプロンプトで,
    >java -version
    を実行する.
  - Javaがインストールされていれば，バージョンが表示される.
  - エラーが出たら，Javaはインストールされていないので，下記を参照してOpen JDKをインストールする
- **Open JDKのインストール**
  - Windows 版 OpenJDK インストール手順
    https://qiita.com/ryo-sato/items/87d05021fcc0519e8828

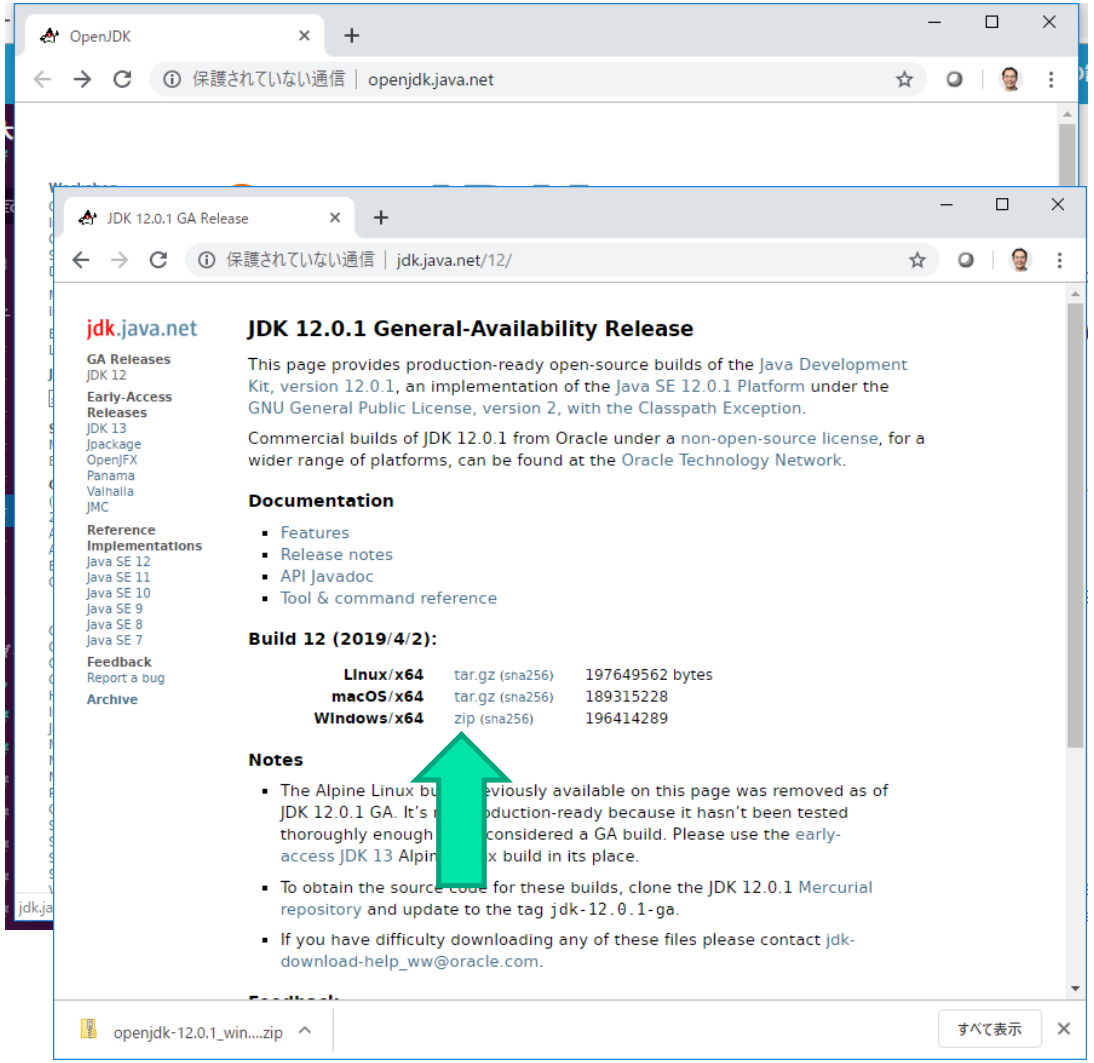
Open JDKのインストール

- 1. ブラウザで，**http://openjdk.java.net/** を開く
- 2. **Download**のところにある **jdk.java.net/12** のリンクをクリック



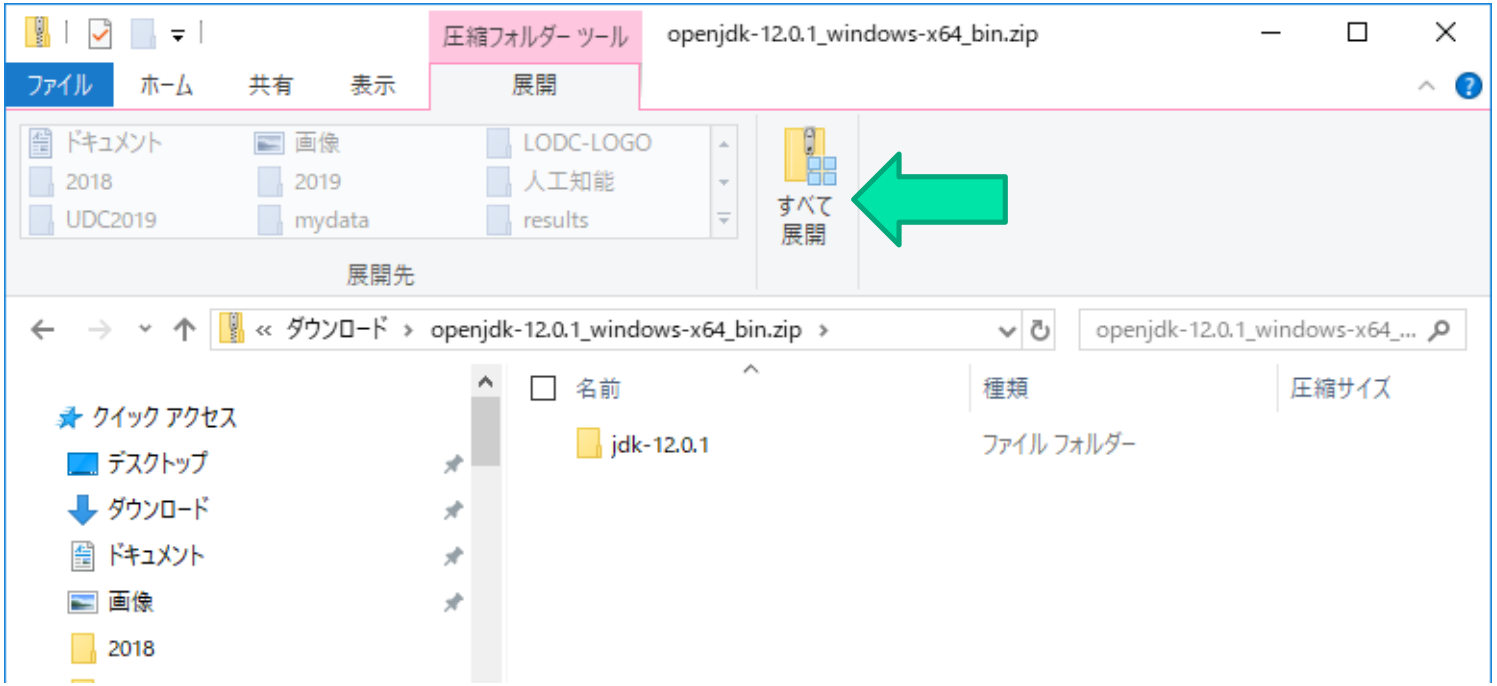
Open JDKのインストール

- 1. ブラウザで，**http://openjdk.java.net/** を開く
- 2. **Download**のところにある **jdk.java.net/12** のリンクをクリック
- 3. 開いたページで **Windows/x64** **zip** をダウンロードする



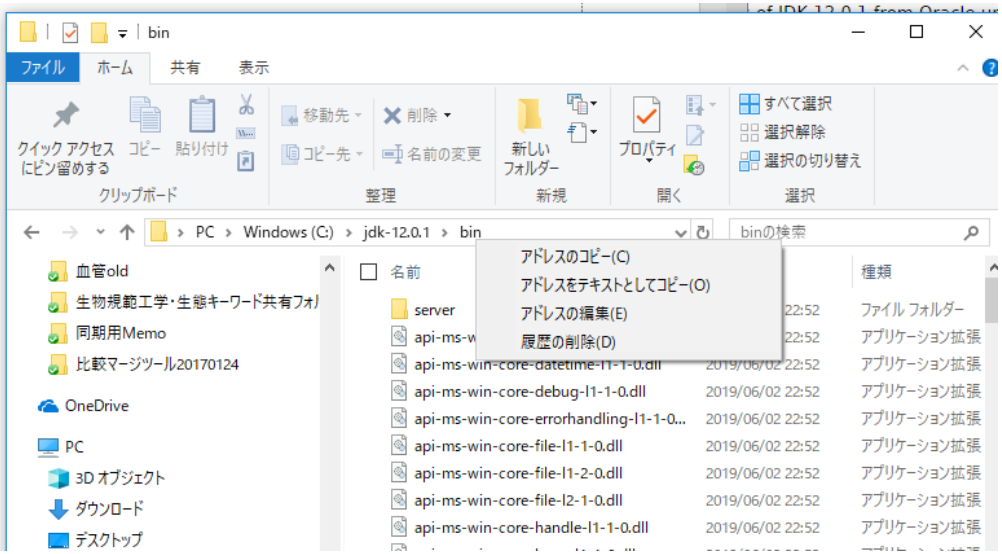
Open JDKのインストール

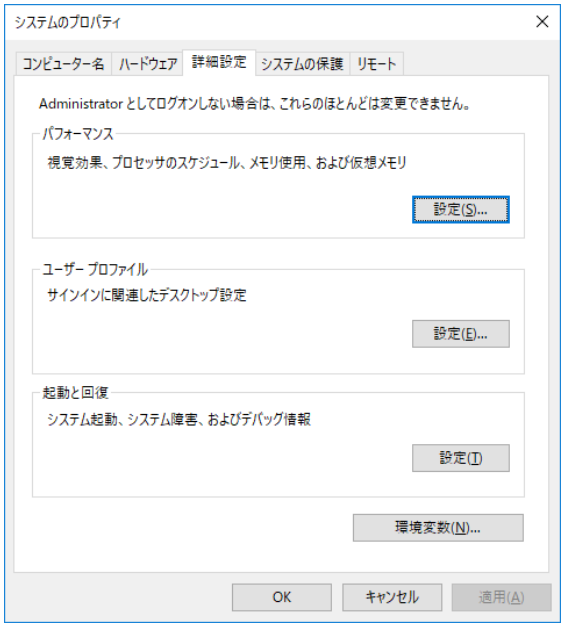
- **4. ダウンロードしたZIPファイル（openjdk-12.0.1_windows-x64_bin.zip）を適当なフォルダに展開する.**
  - **C:¥ や　D:¥ の直下を推奨**



Open JDKのインストール

- 5. 解凍されたできたフォルダ（jdk-12.0.1）の下にあるbinフォルダのアドレスを「テキストとしてコピー」する
- 6. システムのプロパティの「環境変数」のPATHに，5.のアドレスを追加
- 7. コマンドプロンプトでJava -version を実行し，正しくインストールできたことを確認する.

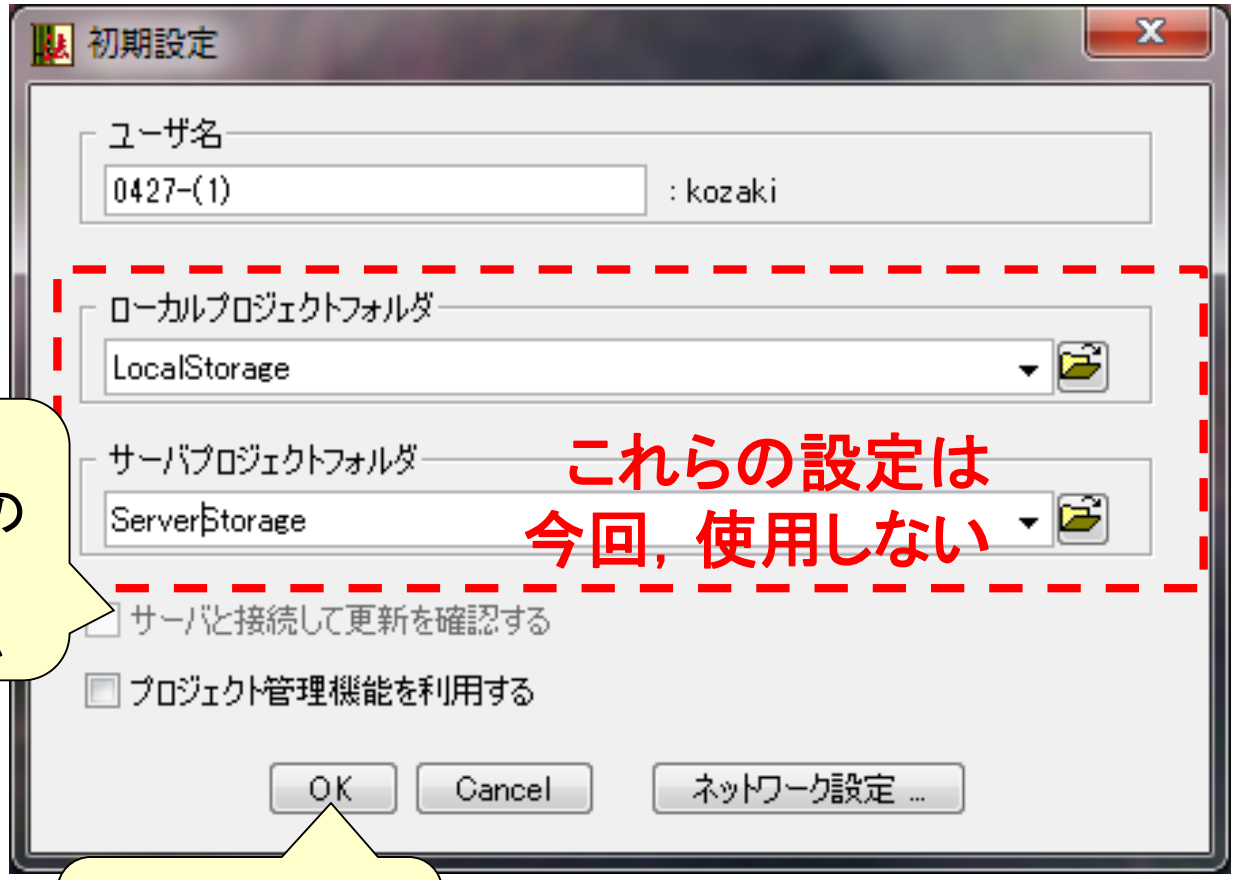




「法造」のダウンロード

- **ダウンロードサイト**
  - http://www.hozo.jp
  - 実行にはJavaの実行環境（Java8以上を推奨）が必要
    - ※Javaのインストール確認は，前ページ参照．
- **起動方法**
  - ダウンロードした圧縮ファイルを解凍したフォルダで，以下のいずれかの方法で起動
  - (1)oe57run.jarをダブルクリック
  - (2)コマンドプロンプト/ターミナルで，
    >java -jar oe57run.jar
  - (3)oe5.batをダブルクリック【Windowsのみ】

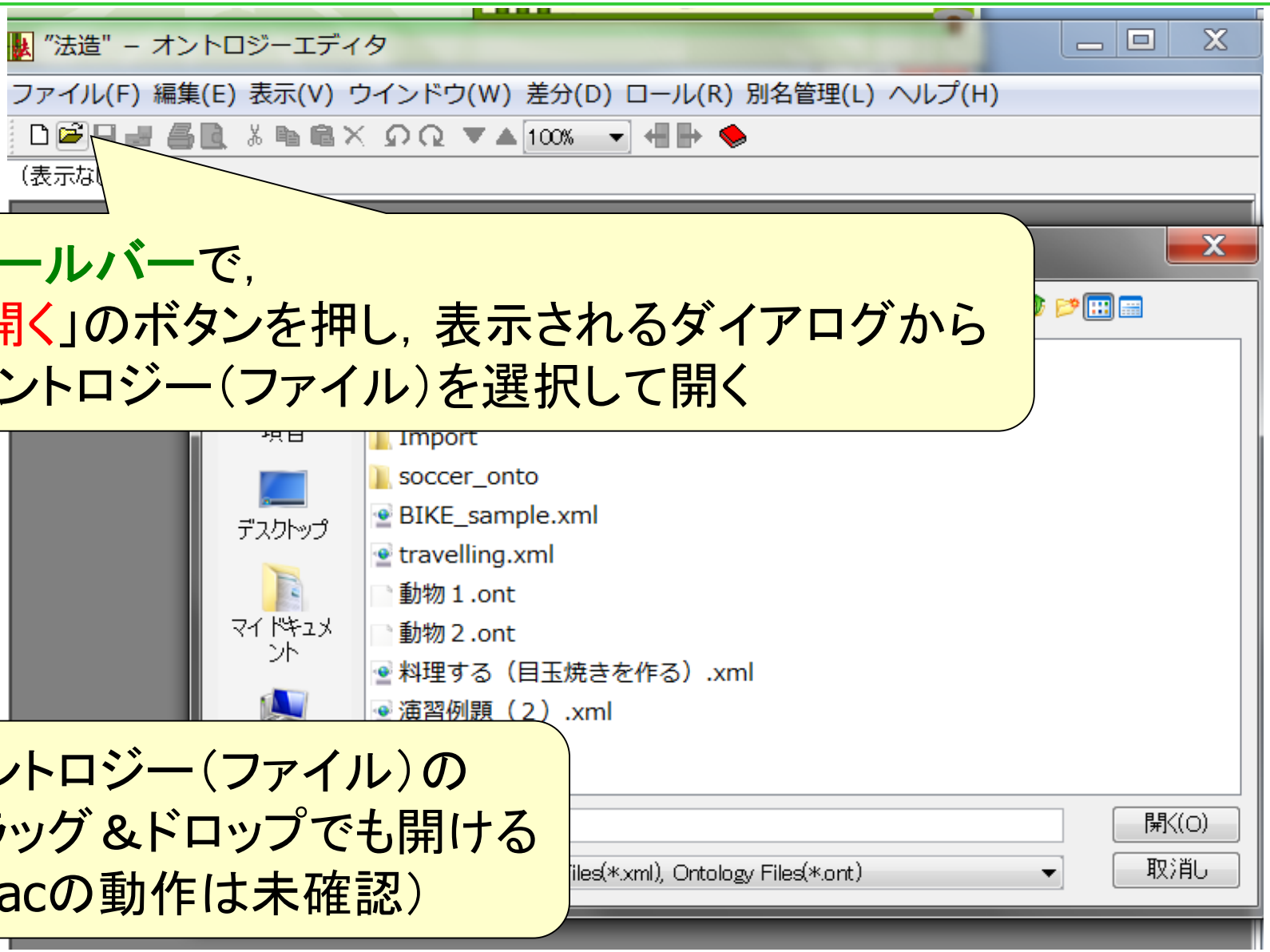
起動画面



「プロジェクト管理機能を利用する」の“チェックが**外れている**”のことを確認

OKボタンで起動

サンプルファイルの開き方



オントロジー工学の基礎

- オントロジーの定義
- オントロジー構築の基本的な考え方

オントロジーの定義

- **哲学用語**:「存在論」
- **人工知能分野**
  - 「概念化の明示的記述」

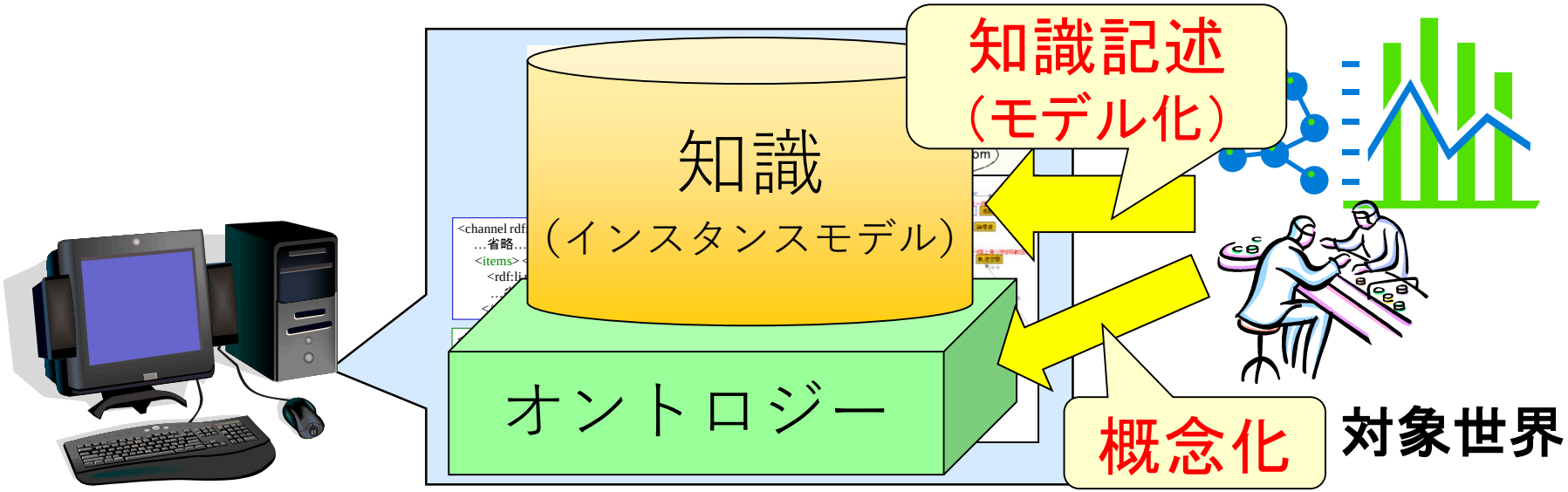
    （An explicit specification of conceptualization）

    by T. Gruber（1993）
- **オントロジー工学**
  - 人間が対象世界をどのように見ているかという根源的な問題意識を持って物事をその成り立ちから解き明かし，それをコンピュータと人間が理解を共有できるように書き記したもの

    （『オントロジー工学』:溝口理一郎）

知識記述とオントロジー

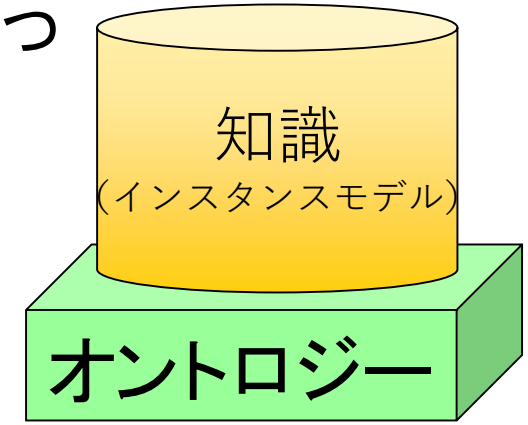
- **知識記述（モデル化）**
  - 知識処理には，対象世界のモデル化（知識記述）が必要．
    - 知識の共有・有効活用には，**一貫性をもったモデル化**が重要．
- **オントロジーの役割**
  - 「対象世界をどのように捉えたか（概念化したか）を明示し，一貫性を持って知識（インスタンスモデル）を記述するための共通概念や規約を提供するもの」



オントロジー構築・利用の意義

知識をオントロジーに基づいて記述すると...

- **知識の記述が容易になる**
  - どのような観点から対象を捉えるかが規定されている
  - 用いる概念・語彙が事前に用意されている
  - 制約違反を自動的にチェックできる
- **記述された知識の「質」が向上する**
  - 概念の統一や観点の一貫性が保てる
  - 他人が記述した知識を理解する際に役立つ
  - 知識を変換することができる
  - 知識を共有・知識を再利用できる

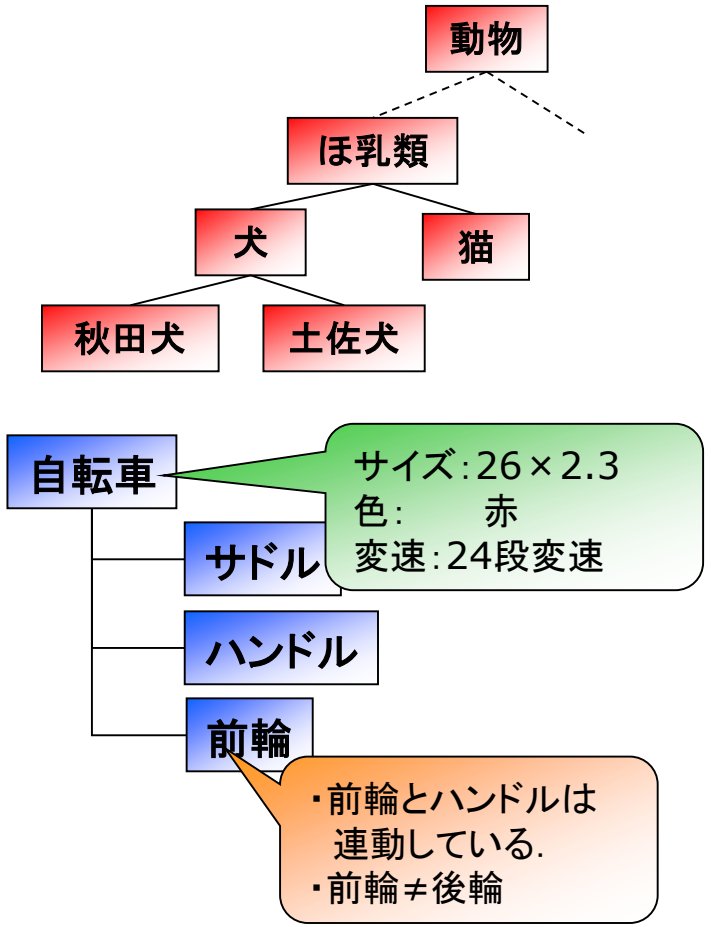


あらゆる知識を扱う際に，その基盤として役立つものがオントロジー

オントロジーの一般的な構成

- **オントロジーの構成**
  - 対象世界を説明するのに必要な「**概念**」
  - 概念間の「**関係**」

- **概念定義の内容**
  - ラベル(, コメント)
  - 上位概念／下位概念 is-a関係
  - 部分概念 part-of関係
  - 属性 attribute-of関係
  - 公理 その他の関係



オントロジー構築の基本的な考え方

- **オントロジー構築の基本姿勢**
  - 「何が本質か？」を追求する
    - 例）**人間 vs 教師，もの vs プロセス**（川や滝はどちら？）
    - この基本姿勢が，**データスキーマ**や**シソーラス**などとの違い
  - 「対象世界をどのように捉えたか（概念化したか）を明らかにすることにより，諸概念の共通性と相違点を明確にする」
- **具体的な構築指針**
  - 概念間の意味的相違点（分類視点）の明確化
    - 概念の「違い」を理解する（分かる＝分ける）
    - 概念間の違いを明確にすることが，is-a階層構築の基本指針
  - 概念の共通性・本質属性
    - 対象とする概念群に**共通する性質**や**本質的な性質**（**本質属性**）を捉える

分類視点の明確化の例：車両

- 例1）概念（分類）階層のみ

車両
- ー二輪車
  - ー自動二輪
  - ー自転車
- ー三輪車
- ー
- …

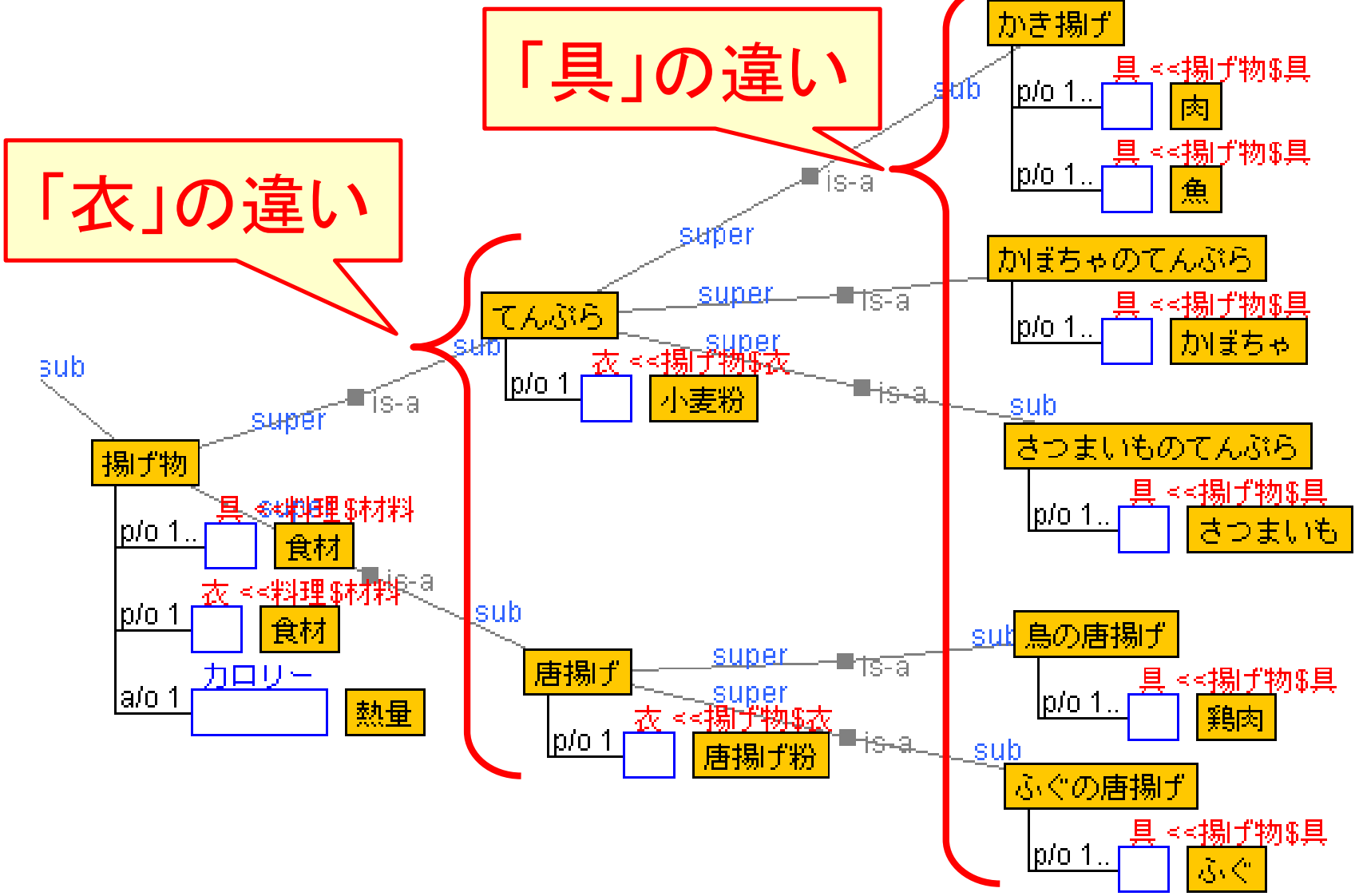
- 例2）概念の定義を追加

車両
- ー二輪車
  - →車輪の数 ＝２
  - ー自動二輪
    - →動力源 ＝エンジン
  - ー自転車
    - →動力源 ＝人
- ー三輪車
  - →車輪の数 ＝３
- ー
- …

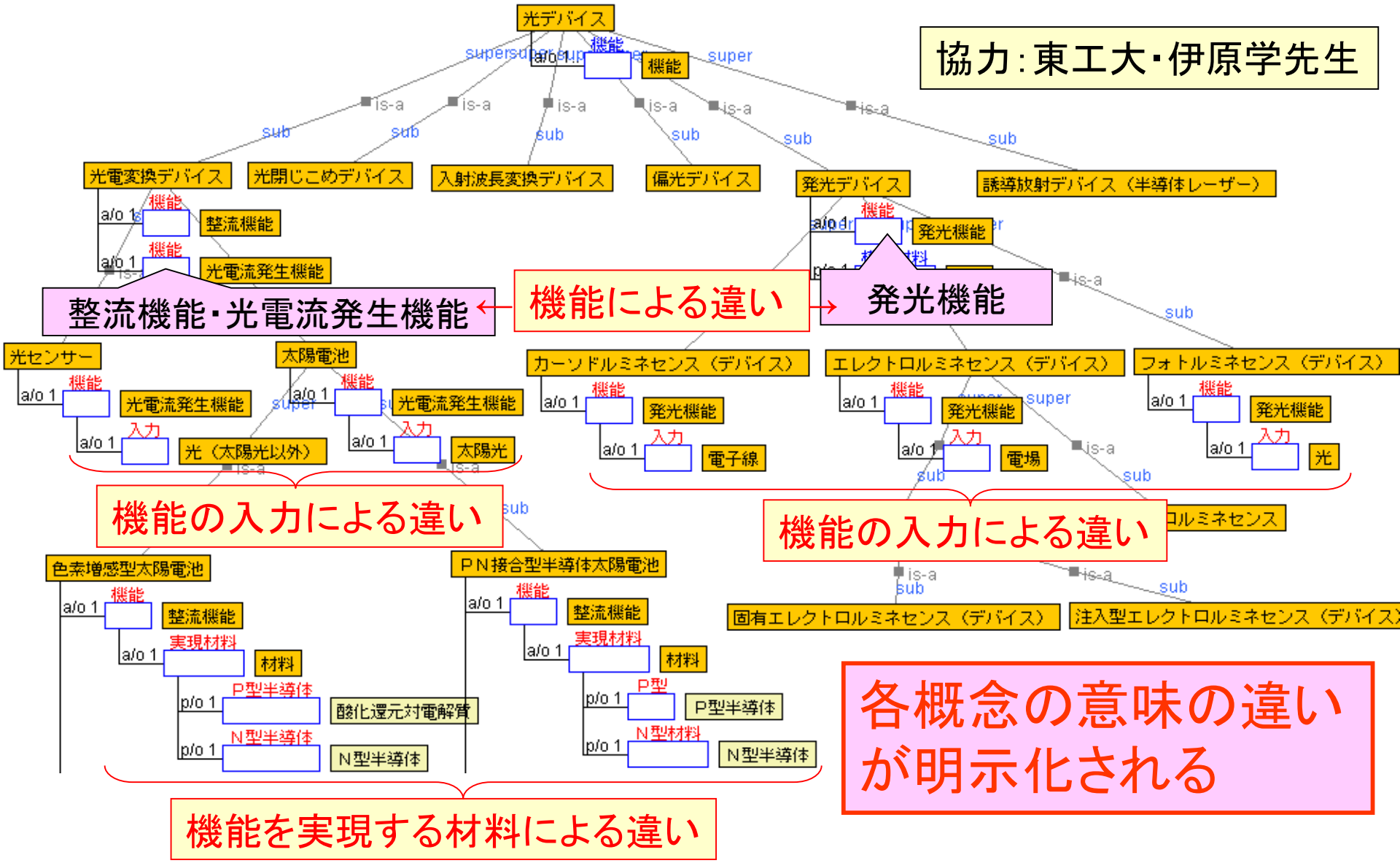
各概念の意味の違いは暗黙的

各概念の意味の違いが明示化される

分類視点の明確化の例 :料理(揚げ物)



分類視点の明確化の例 :光デバイス



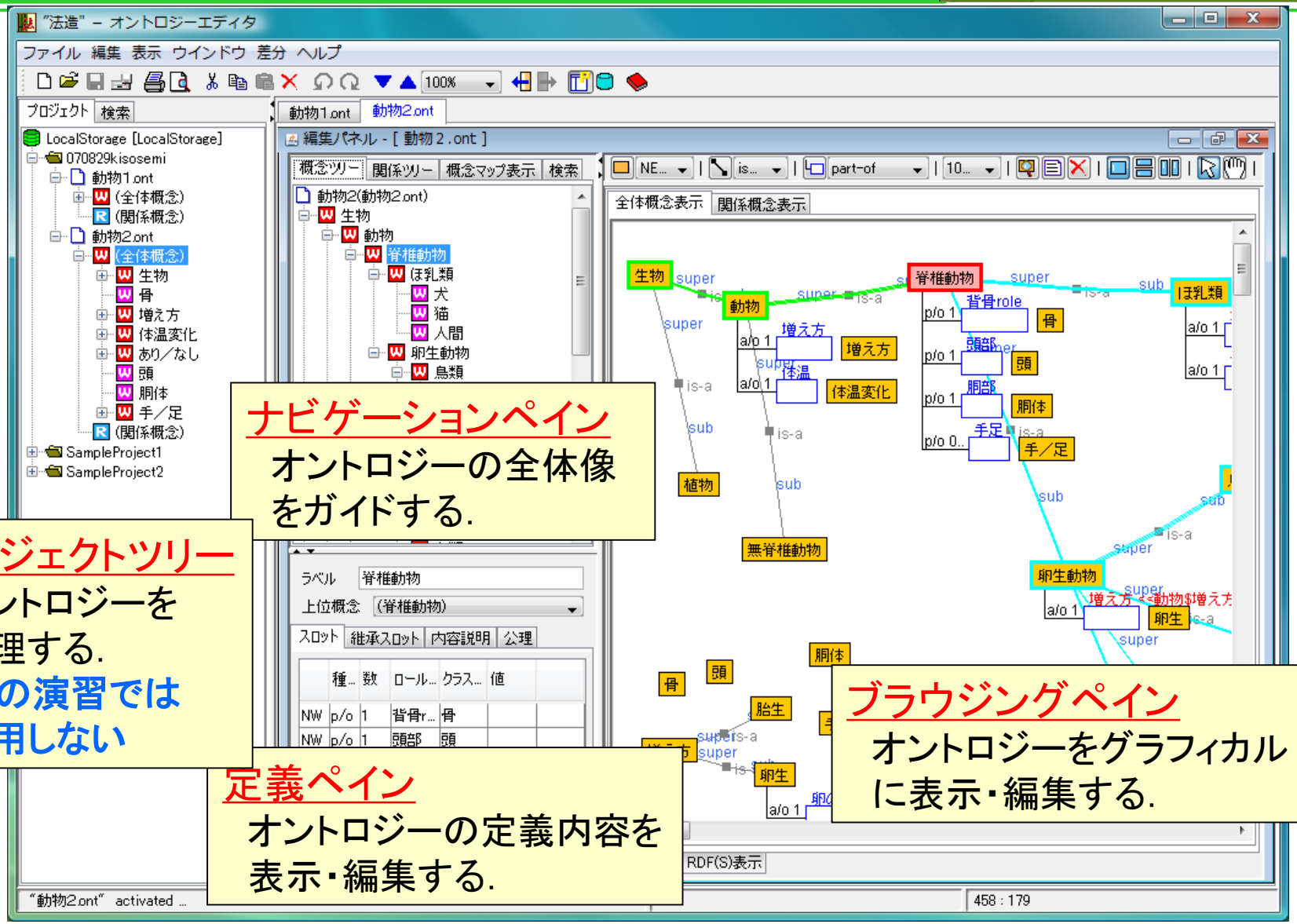
まとめ：
オントロジー工学の基礎

- **オントロジー**
  - 対象世界を“どのように捉えた（概念化した）か？”を（コンピュータ可読な形で）体系化したもの
- **オントロジー構築の基本的な考え方**
  - 「何がその概念化にとって本質か？」を追求する
- **オントロジー構築の具体的指針**
  - 概念の「共通性」と「相違点」を明確にする

オントロジーの構築方法①

- オントロジーの表示・閲覧
- オントロジー構築の基本操作
  - Is-a階層の構築
  - 各概念の意味定義（スロットの記述）
- Is-a階層構築の推奨基準

オントロジーエディタの画面構成



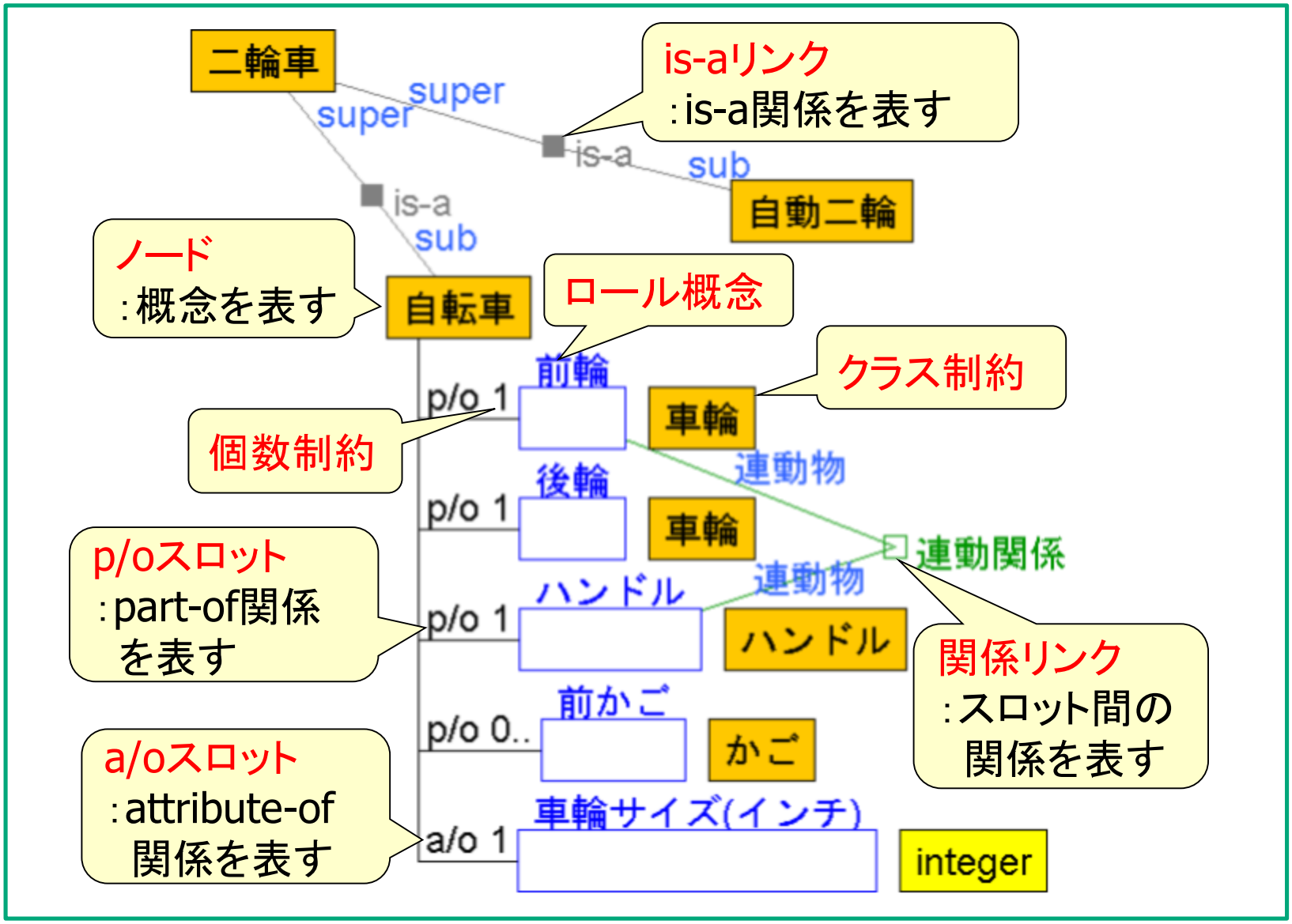
ナビゲーションペイン
オントロジーの全体像をガイドする.

プロジェクトツリー
オントロジーを管理する.
※この演習では使用しない

定義ペイン
オントロジーの定義内容を表示・編集する.

ブラウジングペイン
オントロジーをグラフィカルに表示・編集する.

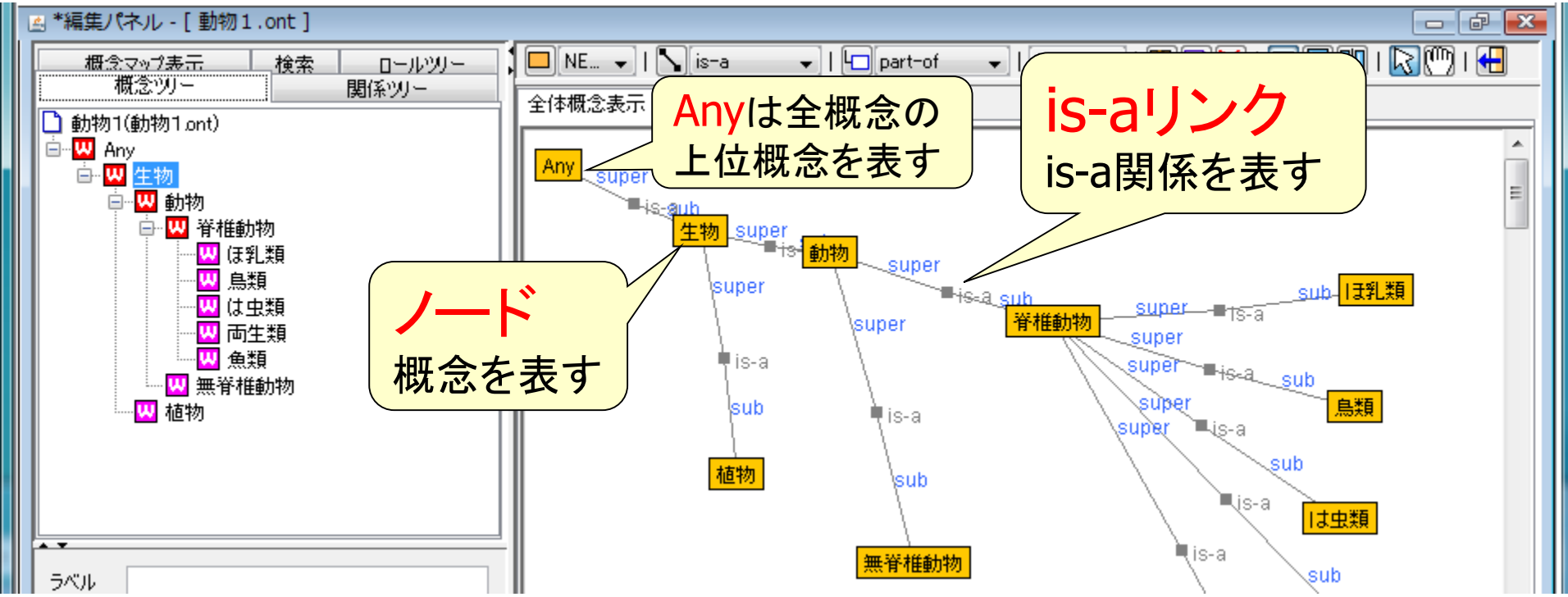
「法造」のオントロジー表現



オントロジーの表現①

サンプル：
動物1.ont

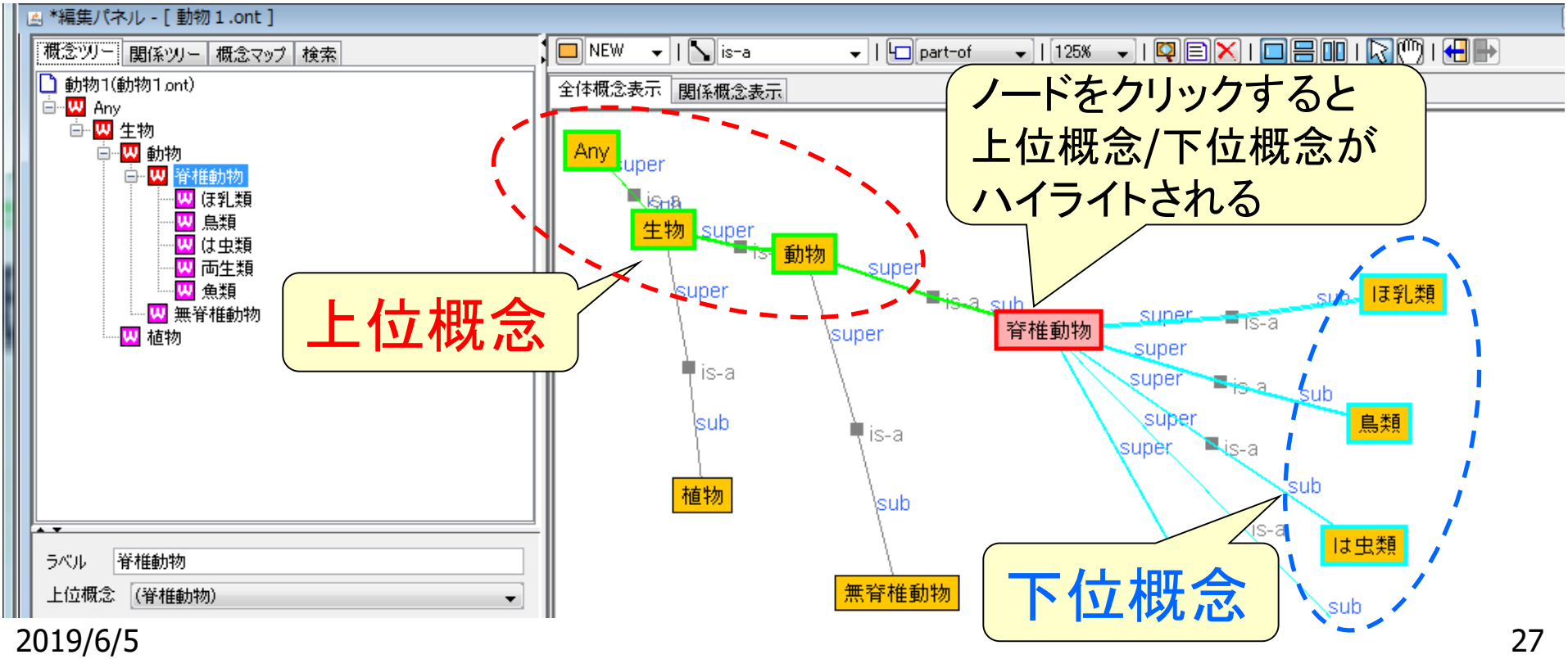
- **概念とis-a関係**
  - 概念（クラス）：ノードで表される
  - is-a関係：**is-aリンク**で表される
    - 画面左には，is-a関係がTree階層で表現される



オントロジーの表現①

- **is-a関係**
  - 概念間の一般-特殊関係を表す
    - 動物　*is-a*　生物　（動物は生物 *である*）
      →動物は，生物を***特殊化した概念***（下位概念）
      →生物は，動物を***一般化した概念***（上位概念）

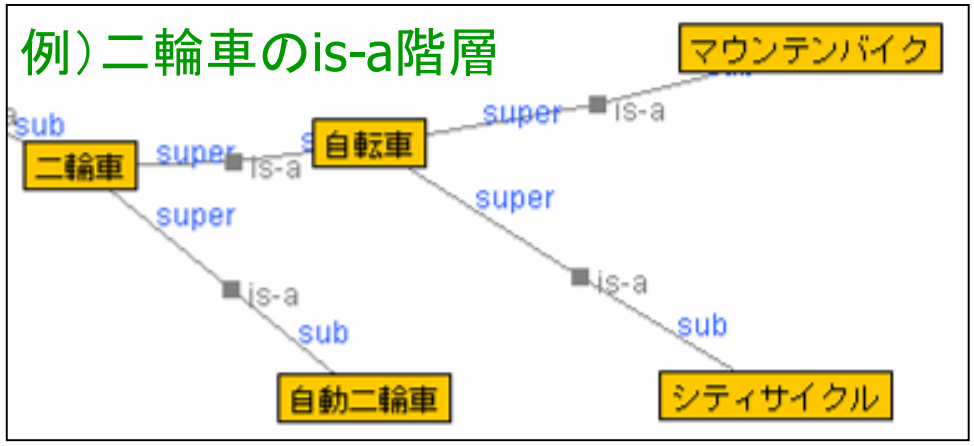
「動物1.ont 」を開いてis-a関係を確認する

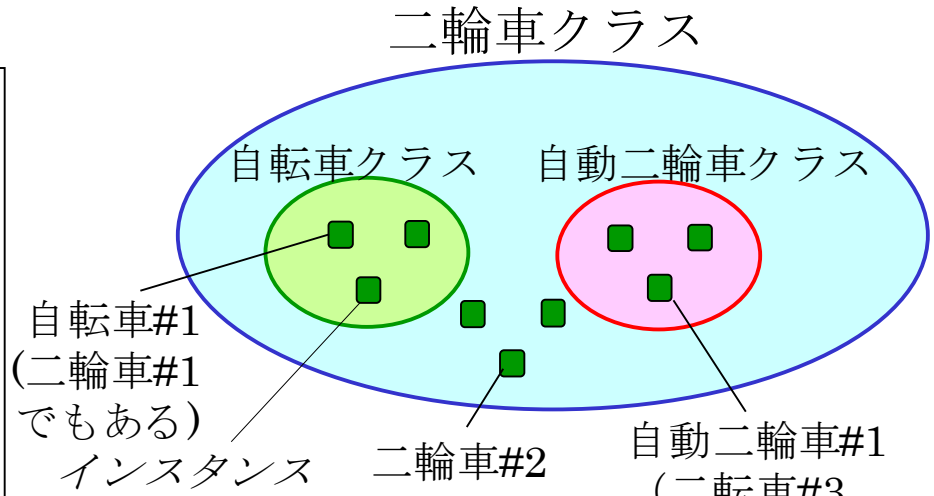


補足説明：is-a関係を用いた階層化の意味

- **上位-下位（is-a）関係（sub-class-of，a-kind-of関係）**
  - **「下位概念（クラス）は上位概念（クラス）の一種である」**ことを表す．
    - 例）自転車 *is-a* 二輪車，人間 *is-a* ほ乳類
  - 下位概念のインスタンス集合は上位概念のインスタンス集合の部分集合となる．
    - 例）全ての人間（のインスタンス）は，ほ乳類（のインスタンス）である．
  - 上位概念の性質は，下位概念に継承される．**＜性質の継承＞**
- **is-a関係に基づく階層**
  - **「世界にあるものはどのように分類されるか」**を表し，オントロジーの主要な構成要素となる．

『オントロジー構築入門』2.2





概念の分類視点の明確化

- 例1）概念のis-a階層のみ

車両
- ー二輪車
  - ー自動二輪
  - ー自転車
- ー三輪車
- ー
  - …

- 例2）概念の定義を追加

車両
- ー二輪車
  - →車輪の数 ＝２
  - ー自動二輪
    - →動力源 ＝エンジン
  - ー自転車
    - →動力源 ＝人
- ー三輪車
  - →車輪の数 ＝３
- ー
  - …

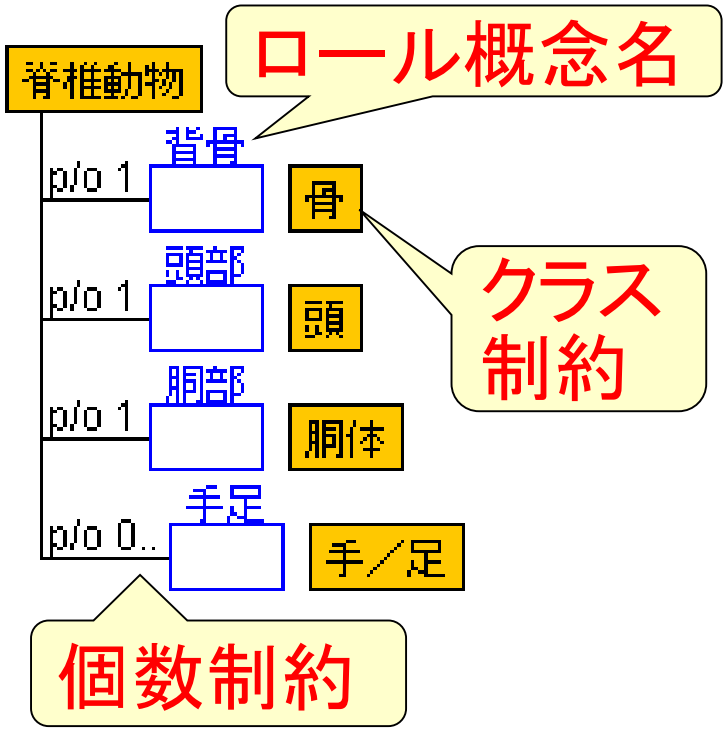
各概念の意味の違いは暗黙的

各概念の意味の違いが明示化される

オントロジーの表現②

サンプル：
動物2.ont

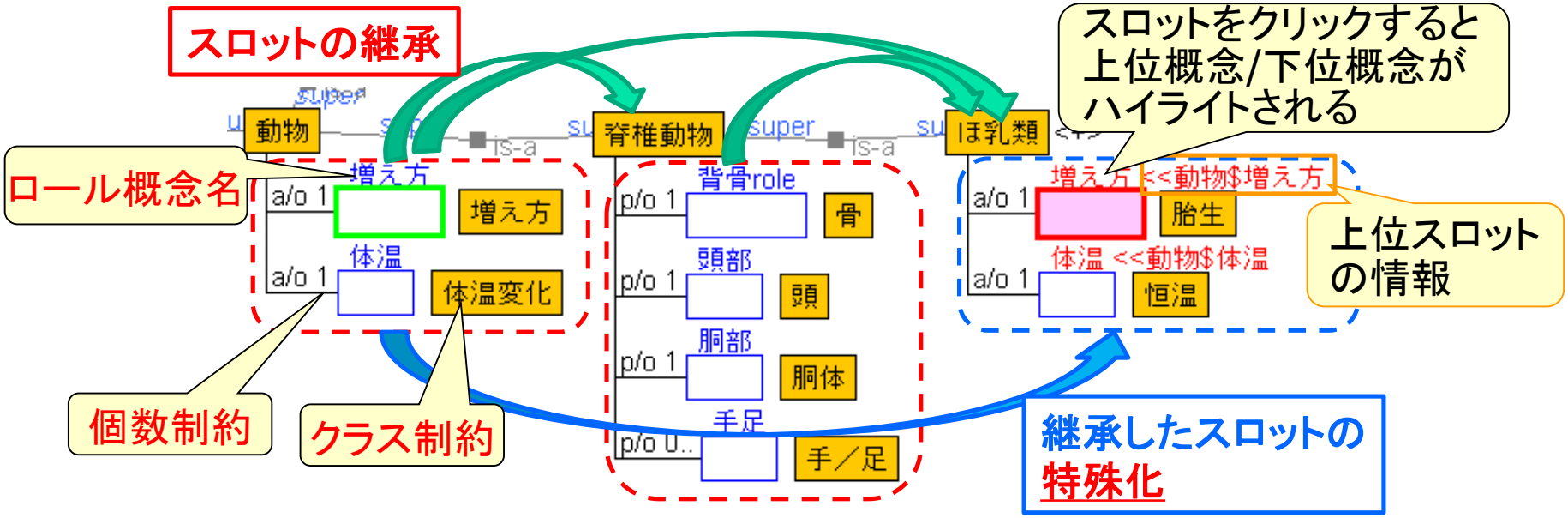
- **part-of，attribute-of関係**
  - スロットで表現される
  - **p/o**：part-of　**a/o**：attribute-of
- **スロット定義の意味**
  - 概念の定義を計算機が理解可能な形で明示化する
- **スロットの表現法**
  - ロール概念名：部分や属性の名前
  - クラス制約：部分や属性となる概念の種類を規定
    ※他の箇所で定義している概念を参照する
  - 個数制約：部分や属性の数を規定
    - **n..m** → **n**以上**m**以下ということを表す



is-a関係の性質

「動物2.ont」で
スロットの継承・特殊化を確認する

- 上位概念から下位概念へ性質が**継承**される
  - 「法造」では**スロットが継承**される
    ※継承したスロットは**ブラウジングペイン上では表示されない**
  - *例）“動物”から“脊椎動物”に継承されているスロットは？*
- 継承した性質（スロット）が下位概念で**特殊化**されることがある
  - *例）“増え方”スロットの特殊化*
  - 「法造」では**継承・特殊化したスロットは「赤色」で表示**される
  - スロットをクリックすると，上位/下位のスロットがハイライトされる



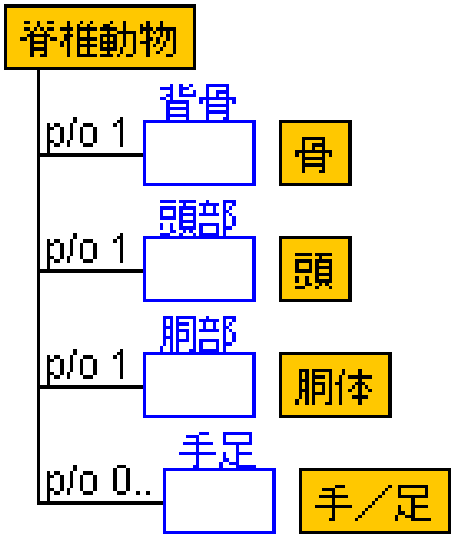
オントロジーの表現③

- **定義ペインにおける表現**（概念選択時）
  - 選択した概念（ノード）の定義内容を表示する.
    - ラベル：概念名
    - 上位概念：クリック操作で上位概念をリスト表示
    - スロット：定義されたスロット一覧
    - 継承スロット：上位概念から継承したスロット一覧
    - 内容説明：概念の説明を自然言語で表現
    - 公理：より詳細な公理記述（使用改訂中）

スロット選択時の定義ペインについては後ほど解説

表示する継承スロットと連動

	種類	数	ロール概念	クラス制約	値	
NW	p/o	1	背骨	骨		
NW	p/o	1	頭部	頭		
NW	p/o	1	胴部	胴体		
NW	p/o	0..	手足	手／足		



その他の閲覧機能

- **ブラウジングペインを用いた閲覧**
  - 全体概念／関係概念の切り替え［次回説明］
  - 概念表示／RDF(S)表示の切り替え
  - 拡大・縮小表示
  - 選択／スクロールの切り替え
  - クラス制約のクリックによるジャンプ機能
  - 上位・下位・参照先へのジャンプ機能
  - 上位概念・下位概念のハイライト表示
  - 別ウィンドウ表示（※継承したスロットも表示可能）
- **ナビゲーションペインを用いた閲覧**
  - 概念ツリー：概念のis-a階層をTree状に表示する.
  - 概念マップ：オントロジーの全体をマップ表示する.
  - 検索：キーワードにより概念を検索する.

Mini Quiz:
サンプルオントロジーの閲覧

- 「動物2.ont」オントロジーを閲覧し下記の問いに答えよ

(1)「コウノトリ」の上位概念を列挙せよ.

→

(2)「脊椎動物」の下位概念はいくつあるか？

→

(3)「鳥類」と「は虫類」のどのような定義の違いで表されているか？

→

(4)「人間」が持つスロットをすべて列挙せよ(継承したものも考慮するが，特殊化したものは重複して数えない).

→

(5)「恒温」・「変温」を参照している概念は，それぞれいくつか？(継承したスロットは考慮しなくて良い)

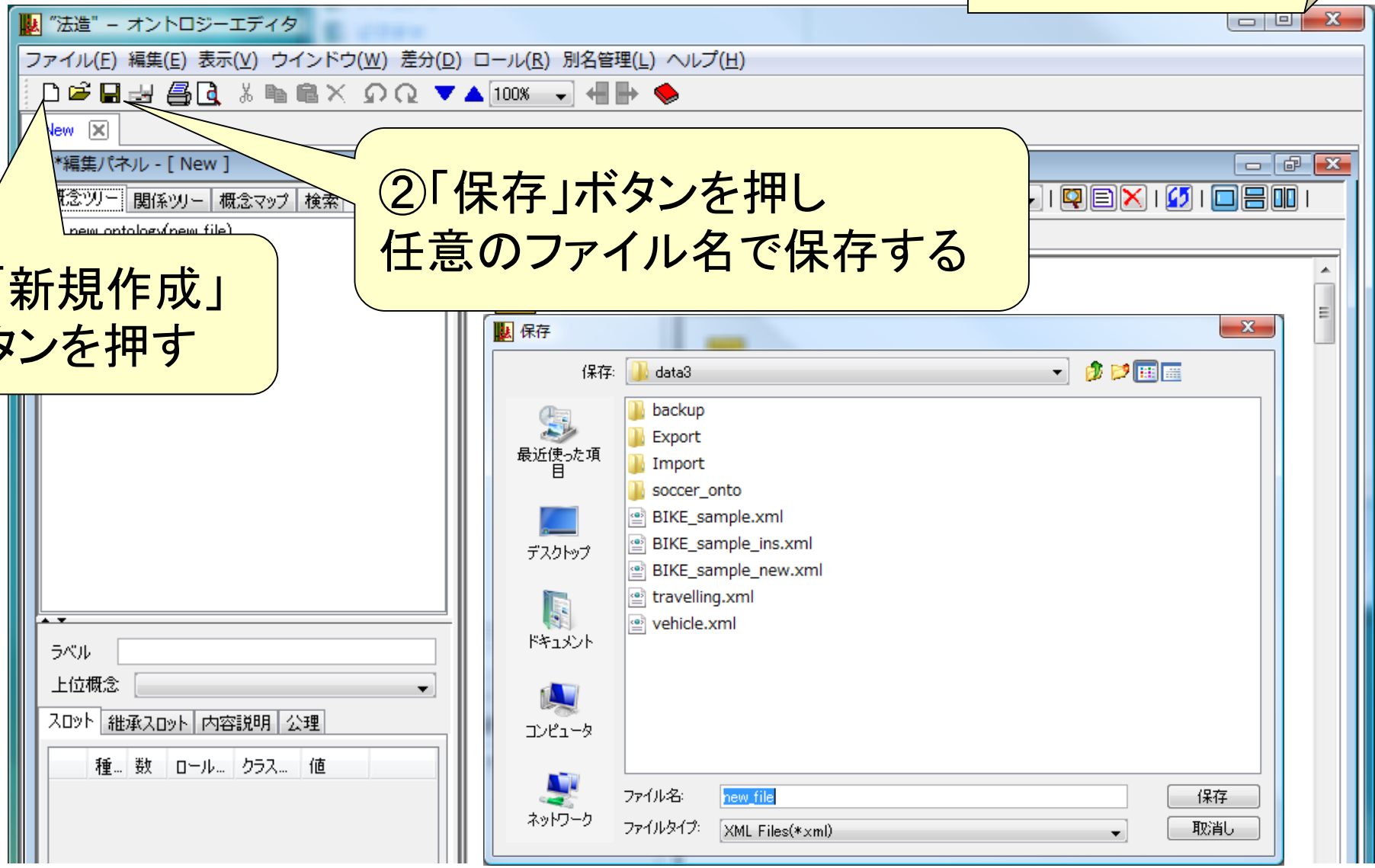
→

おまけ:(時間が余れば答える)このオントロジーでは，いくつの概念，およびスロット)が定義されているか？　ヒント:検索機能を利用する

→

新規オントロジーの作成

サンプル：
揚げ物.xml



①「新規作成」
ボタンを押す

②「保存」ボタンを押し
任意のファイル名で保存する

共通する編集操作

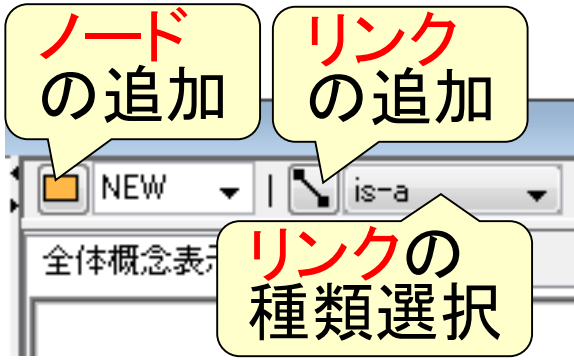
- **編集操作の流れ**
  - ツールバーのボタンを押すと，ブラウジングペイン上にノード/リンク/スロットが追加（削除）される
  - ブラウジングペイン上で編集するノード/リンク/スロットを選択する
  - 定義ペインで定義内容を編集する

ツールバーの概要

- **備考**
  - 編集操作はブラウジングペインで選択中のノード/リンク/スロットに対して行われる
  - 必要に応じてツールバーの選択リストを切り替える
  - ブラウジングペインでの複数選択は「SHIFT＋クリック」
  - ブラウジングペイン上の右クリックメニューでも同じ操作が可能

基本操作(1)
概念とis-a階層の作成

- **概念(ノード)の作成**
  - [概念を追加]で新規概念の追加
    - ボタン，メニューバー，ポップアップメニューで同じ操作が可能.
- **概念名(ラベル)の変更**
  - ノードを選択して，**定義ペイン**で変更
    - ノードを選択して，右クリック→[ラベルを変更]でも変更可能
- **is-a階層の構築**
  - 2つのノードを選択(*SIFTキーを押しながらクリック*)し，[リンクを追加]でis-a関係(Linkの種類で"is-a"を選んでおく)を追加.
  - 作成したis-a関係は**ナビゲーションペイン**で，Tree状に表示される.

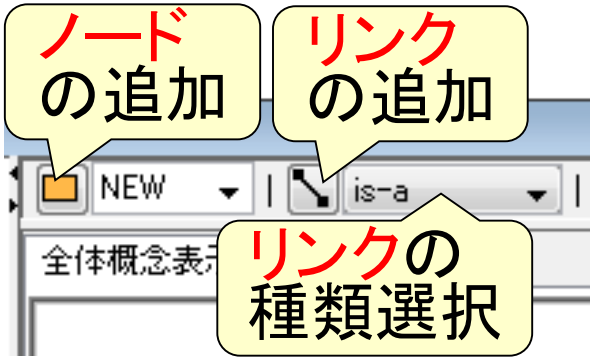


操作練習：例に習いのis-a階層を作る
(※時間が余れば，前の例以外の概念も追加してみる)

補足説明:
is-a階層の作成時の注意点

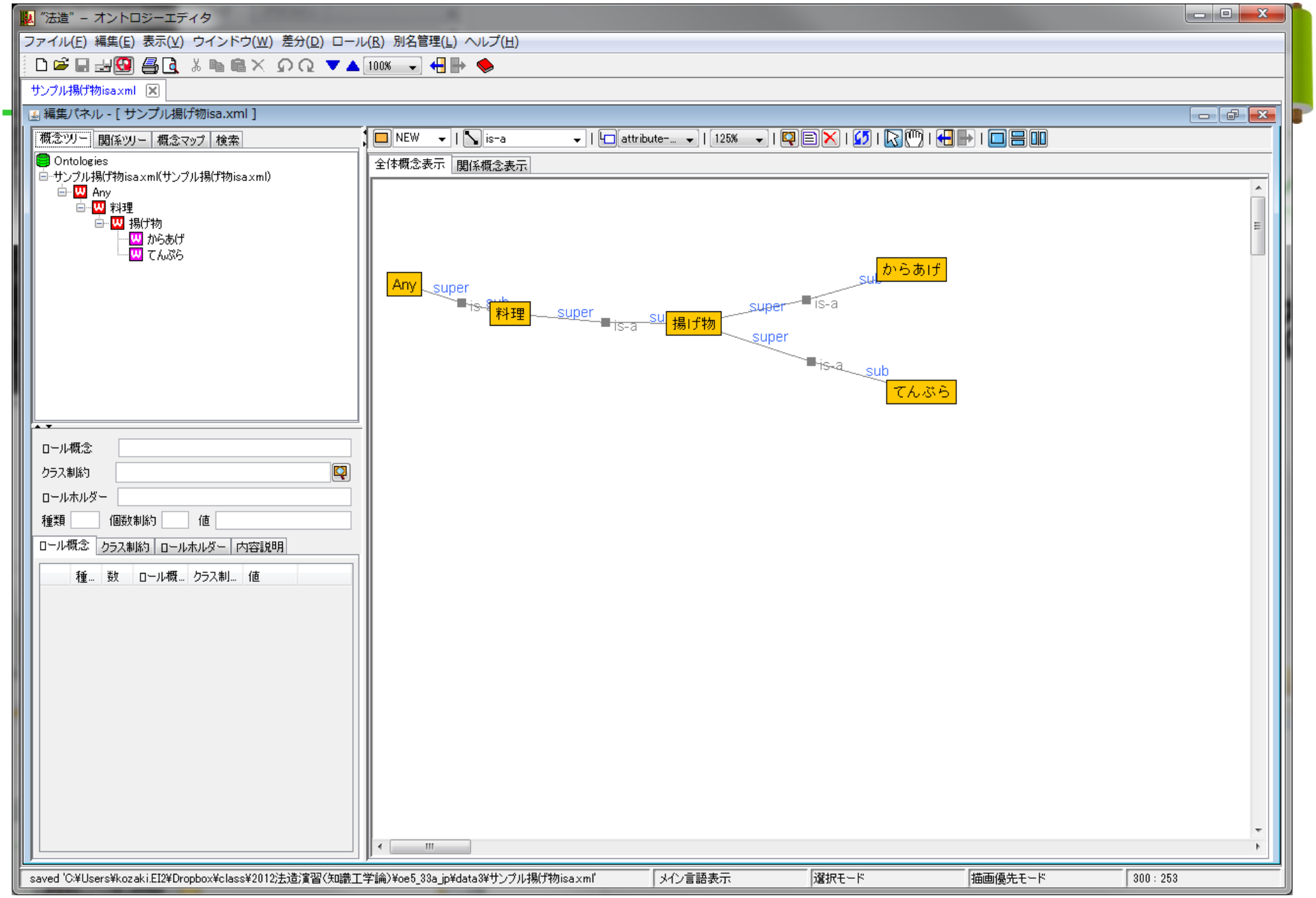
- **下位概念の追加**
  - 既存のノード(概念)を選択した状態で，[概念を追加]すると，新しい概念は，その下位概念となる(自動的にis-a関係が引かれる).

- **リンクの種類選択**
  - リンクの種類で「is-a以外」が選択されていると，is-aリンクが追加できないので注意する.



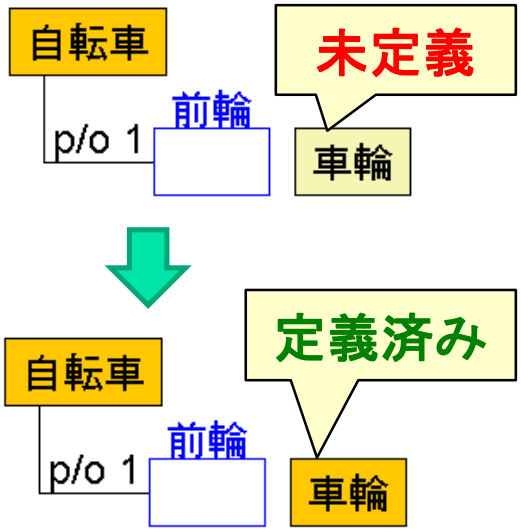
- **多重継承の扱い**
  - 法造では原則として，多重継承(同じ概念が上位概念を2つ以上持つ)ことを認めていない(オントロジー的に様々な弊害があるため).
    - →多重継承となるようにis-aリンクを追加すると，古い方のis-aリンクが自動的に削除される.
  - 多重継承を用いたい時には，以下のいずれかの方法を取る.
    - ロール概念を用いると多重継承を使わずに正しい概念定義ができる(後ほど解説)
    - 2つめ以降のis-a関係に「IS-A関係」(リンクの種類で選択)を用いる.

参考文献:太田 衛, 古崎 晃司, 溝口 理一郎:実践的なオントロジー開発に向けたオントロジー構築・利用環境「法造」の拡張 -理論編 - , 人工知能学会論文誌, Vol.26 No.2, pp.387-402, 2011


"法造" – オントロジーエディタ
ファイル(F) 編集(E) 表示(V) ウインドウ(W) 差分(D) ロール(R) 別名管理(L) ヘルプ(H)
サンプル揚げ物isa.xml
編集パネル - [サンプル揚げ物isa.xml]
概念ツリー 関係ツリー 概念マップ 検索
Ontologies
サンプル揚げ物isa.xml(サンプル揚げ物isa.xml)
Any
料理
揚げ物
からあげ
てんぷら
NEW
is-a
attribute-...
125%
全体概念表示 関係概念表示
Any
super
is-a
sub
料理
揚げ物
からあげ
てんぷら
ロール概念
クラス制約
ロールホルダー
種類
個数制約
値
ロール概念 クラス制約 ロールホルダー 内容説明
種... 数 ロール概... クラス制... 値
saved 'C:¥Users¥kozaki.EI2¥Dropbox¥class¥2012法造演習(知識工学論)¥oe5_33a_jp¥data3¥サンプル揚げ物isa.xml'
メイン言語表示
選択モード
描画優先モード
300 : 253

基本操作(2)
スロットの作成

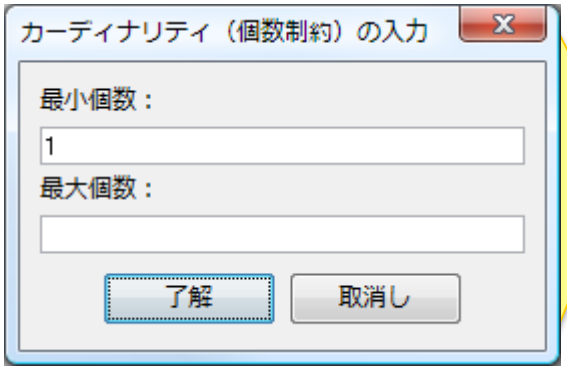
- **part-of/attribute-ofの作成(スロットの作成)**
  - ノードを選択して[スロットを追加]で新規スロットの作成
    - ボタン，メニューバー，ポップアップメニューで同じ操作が可能.
    - part-of／attribute-of　はリストで選択.
- **スロットの定義記述**
  - スロットを選択し，定義ペインで編集
    - **ロール概念**：部分や属性の名前(スロット名)記入
    - **クラス制約**：部分や属性となる概念の種類を規定
  - クラス制約で参照する概念
    - 未定義の概念は「薄い黄色」で表示されるので，
      別途，概念を定義する.(*※右クリックメニューを使うと簡単に定義可能*)
    - **[クラス制約を選択]** を用いると定義済みの概念から選択可能.

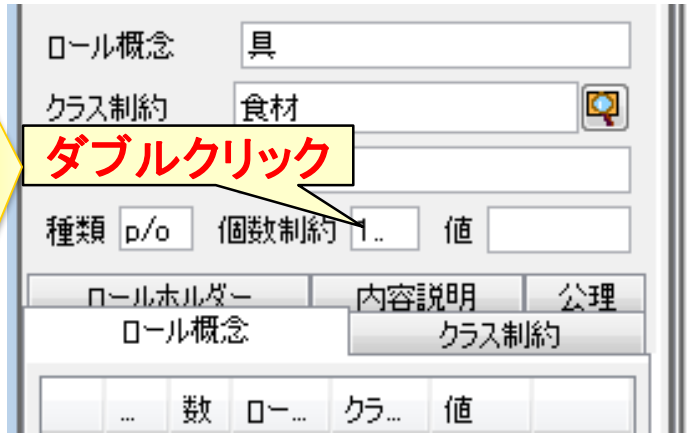


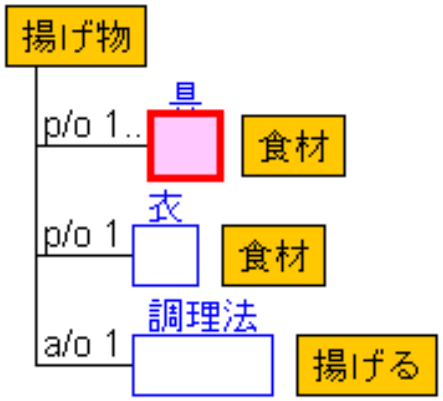
操作練習：スロットの作成.

基本操作(3)
スロット定義の詳細

- **part-of, attribute-of関係(スロットで表現)**
  - p/o(part-of):部分となる概念を表す
  - a/o(attribute-of):その概念の属性(性質)を表す
- **スロットで定義する内容**
  - ロール概念:部分や属性の名前(※詳細は第2回で説明)
  - ロールホルダー:午後に説明
  - クラス制約:部分や属性となる概念の種類を規定
    - 「Any」は,**すべての概念の上位概念**を表す
    - **Data typeの利用**:Integer(整数),Float(浮動点少数),String(文字列),Boolean(真偽値),decimal(小数),date(日時),time(時間),※他も追加予定
  - 個数制約(カーディナリティ):部分や属性の数を規定
    - **n..m** → **n**以上**m**以下ということを表す　※欄をダブりクリックすると入力補助







補足説明:
スロットの作成

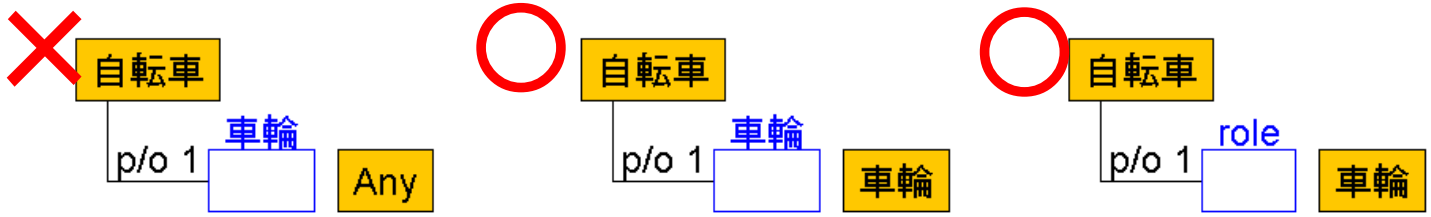
- **part-of/attribute-ofの使い分け**
  - p/o(part-of):部分(部品)となる概念を表す
  - a/o(attribute-of):その概念の属性(性質)を表す

  ※部分(部品)と見なせる場合にはp/o,それ以外はa/oを用いればよい.
    - 法造におけるa/oは,「厳密な意味の属性」だけでなく,広くpart-of以外の関係を表すのに用いる.

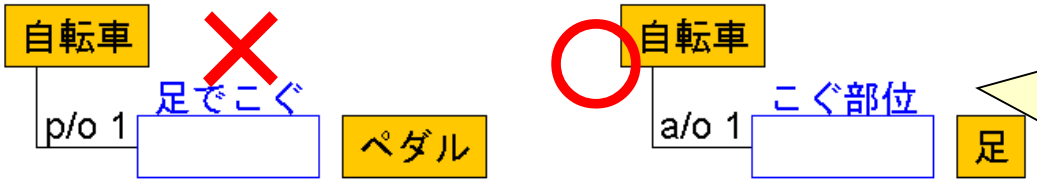
  ※p/oとa/oの使い分けのみでは表せない違いは「ロール概念」で表す=部分や属性(関係性)の名前を付ける.
- **クラス制約のOR条件**
  - クラス制約には複数の概念のOR条件を用いることができる.
    - **[クラス制約を選択]**  の際に,「Ctrl+クリック」で複数の概念を選択
    - 定義パネルの「クラス制約」欄に,**“|(全角)”**で区切って複数の概念を入力

補足説明:
スロット定義の注意事項

- **クラス制約は必ず定義する**
  - ロール概念名とクラス制約が同じになる場合，ロール概念名は省略しても良い．



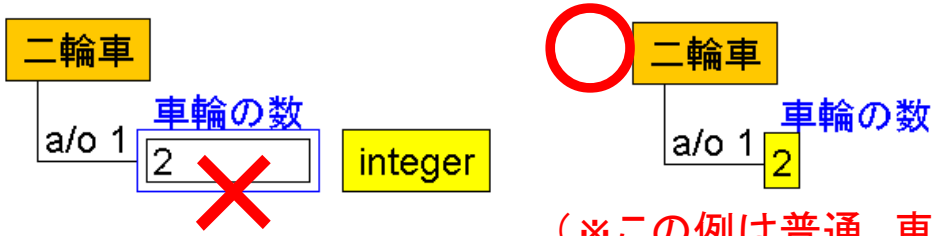
- **スロットのロール概念の欄に説明文を書かない**
  - 自然言語による説明（コメント）を書きたいときは，内容説明の欄に書く．



同等の意味をスロットを工夫して表すこともできる（ペダルは未定義）

- **デフォルト値（インスタンス）の規定**
  - 「ロールホルダー」の欄に書かずに「値」の欄に書く

→黄色のノードで示される

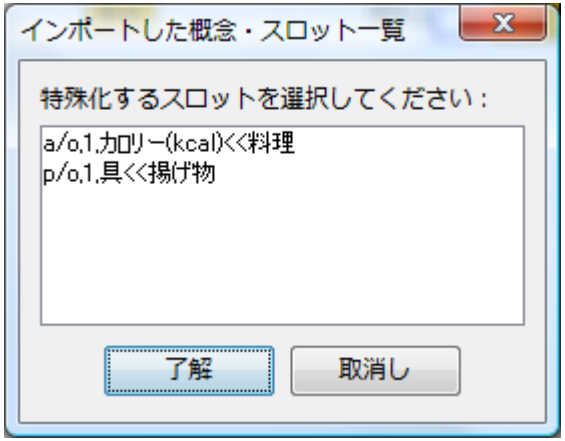


（※この例は普通，車輪の個数制約で定義する）

基本操作(4)
下位概念でのスロットの継承・特殊化

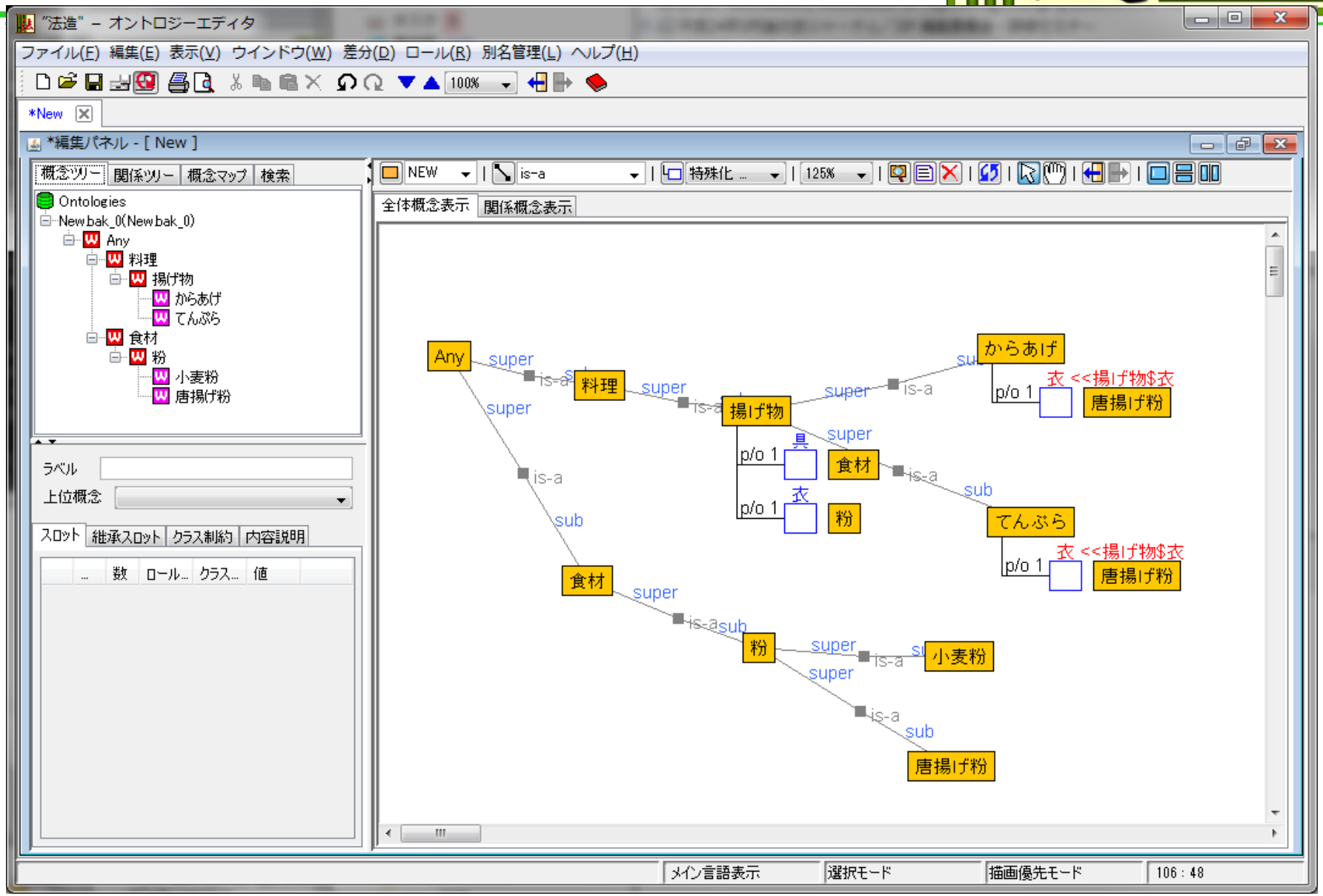
- **継承されているスロットの確認**
  - 定義ペインで[上位概念]のリストから，上位概念を選択すると継承されたスロットが[継承スロット]タブに表示される.

- **スロットの特殊化操作**
  - ノード(スロット)を選択し右クリックメニューから[スロットを追加]→[特殊化]を実行
    - 表示されるスロットの一覧から特殊化したいスロットをマウスでクリックする



- **特殊化されたスロットの定義内容の編集**
  - 新しく作成されたスロットの定義内容を必要に応じて書き直す.
    - 個数制約の特殊化
    - クラス制約の特殊化　．．．など

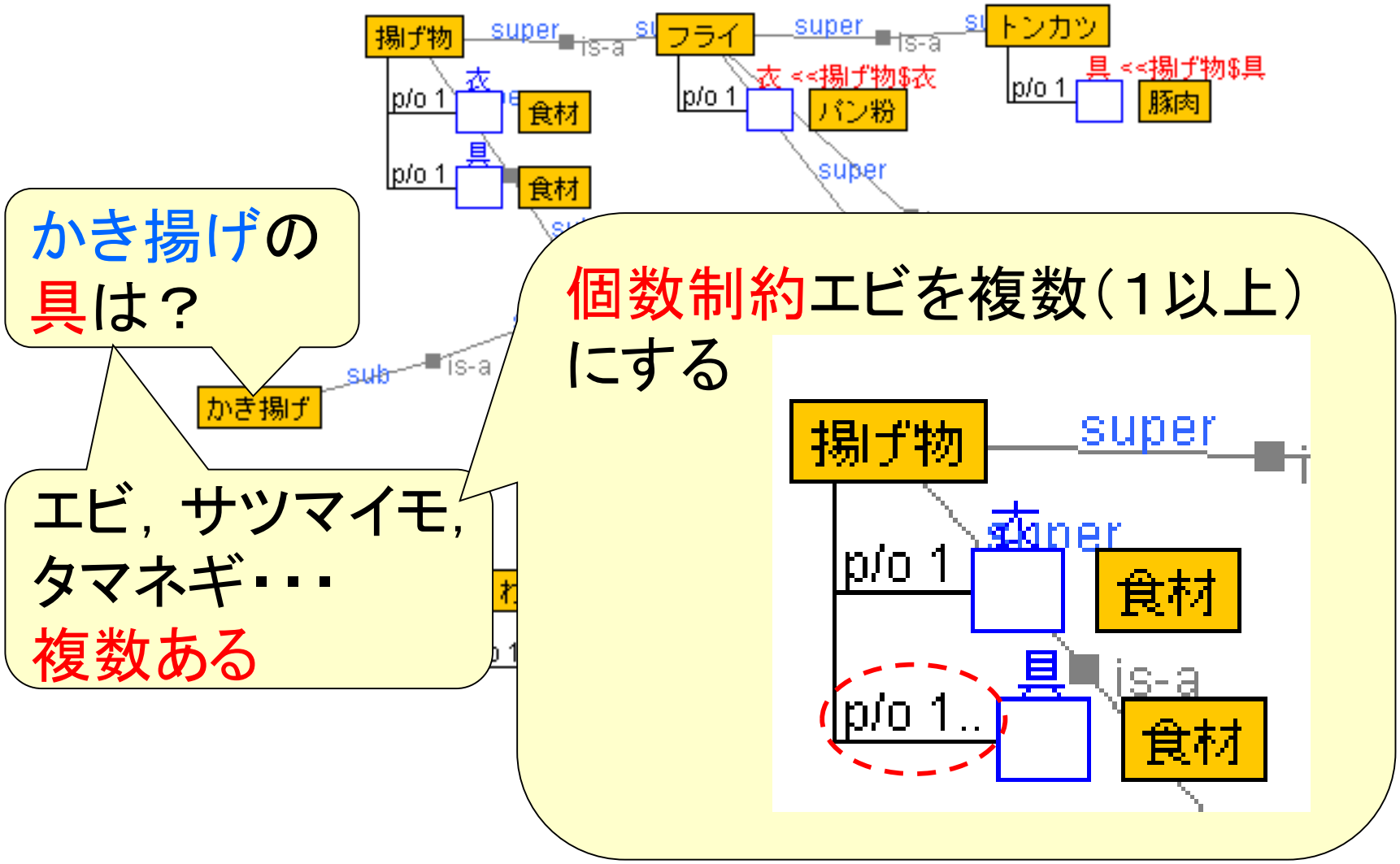
操作練習：スロットの継承・特殊化を利用し，より詳細な定義する.


"法造" – オントロジーエディタ
ファイル(F) 編集(E) 表示(V) ウインドウ(W) 差分(D) ロール(R) 別名管理(L) ヘルプ(H)
*New
*編集パネル - [New]
概念ツリー
関係ツリー
概念マップ
検索
Ontologies
New bak_0(New bak_0)
Any
料理
揚げ物
からあげ
てんぷら
食材
粉
小麦粉
唐揚げ粉
ラベル
上位概念
スロット
継承スロット
クラス制約
内容説明
NEW
is-a
特殊化 ...
125%
全体概念表示
関係概念表示
Any
料理
揚げ物
からあげ
唐揚げ粉
食材
粉
小麦粉
てんぷら
唐揚げ粉
唐揚げ粉
super
is-a
sub
p/o 1
衣 <<揚げ物$衣
メイン言語表示
選択モード
描画優先モード
106 : 48

補足説明:
継承・特殊化の注意事項

- **特殊化操作が不要な場合**
  - スロットが継承されているだけ(定義内容が特殊化されていない)場合は特殊化操作は不要.
    - 継承したスロットは，定義ペインなどで確認できる.
- **継承・特殊化の制限事項**
  - 上位概念の定義に矛盾するような特殊化は行えない.
    - 個数制約:上位概念で定義された数の範囲内でのみ可能.
    - クラス制約:上位概念のクラス制約で指定した概念の下位概念のみ可能
      - [クラス制約の選択]コマンドを使うと，特殊化できる概念のみ表示される.
    - 現状の法造では，制限事項の一部についてのみ，警告や操作の制限を実装している.
    - 制限の違反がないかは「整合性チェック」機能で確認可能.

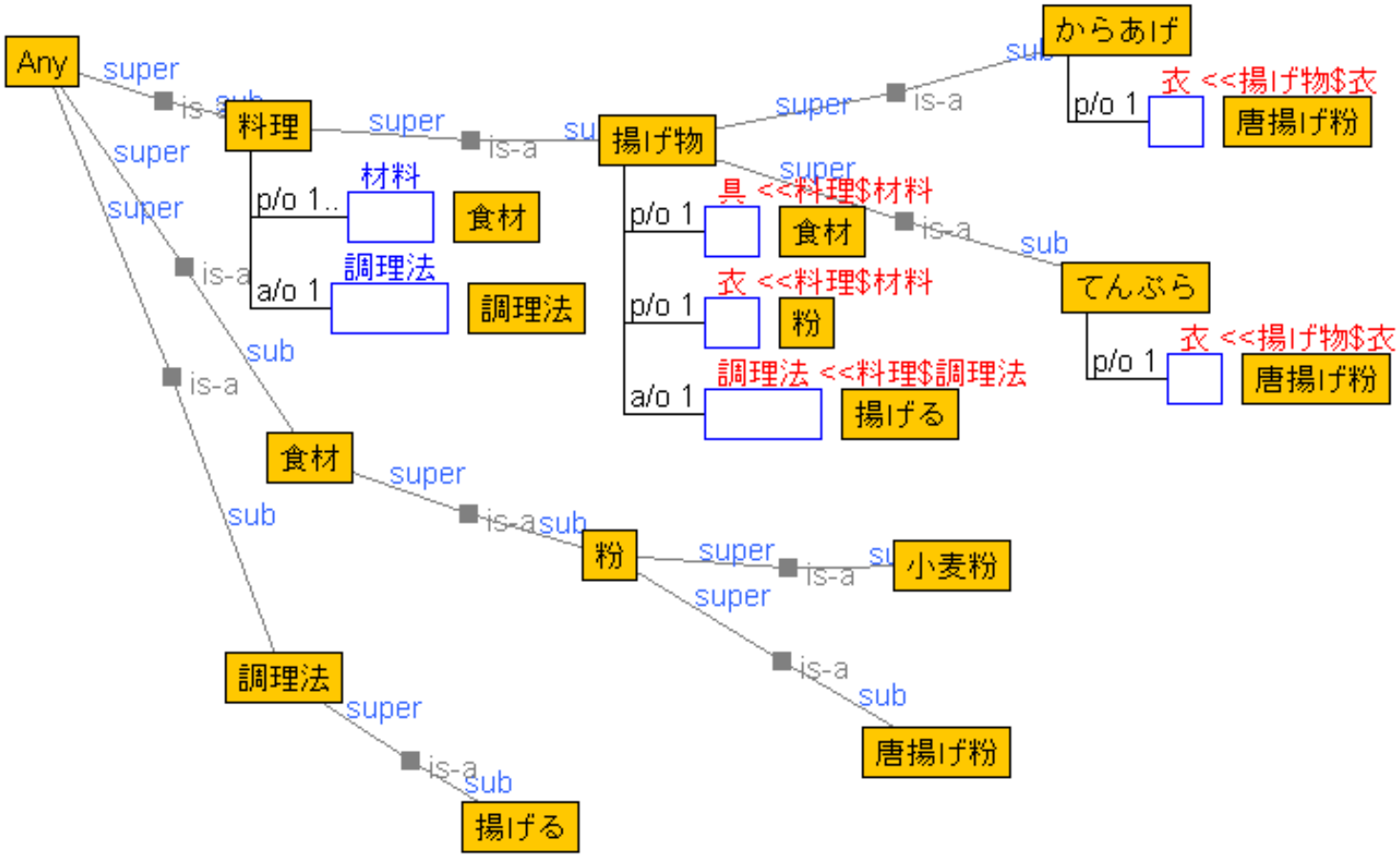
以前のレポートより


揚げ物
super
is-a
フライ
super
is-a
トンカツ
衣
p/o 1
食材
具
p/o 1
食材
衣 <<揚げ物$衣
p/o 1
パン粉
super
具 <<揚げ物$具
p/o 1
豚肉
かき揚げの具は？
エビ，サツマイモ，タマネギ・・・複数ある
sub
is-a
かき揚げ
個数制約エビを複数（1以上）にする
揚げ物
super
衣
super
p/o 1
食材
具
is-a
p/o 1..
食材

補足説明:
その他の継承操作

- 継承・特殊化情報の修正
  - 既に定義済みのスロットを後から継承したことに修正したい場合
    - **警告ダイアログの利用:**
      - 上位概念と同じロール概念を持つスロットを下位概念で定義すると**「特殊化したスロットとして定義する」**ことを確認するダイアログが表示される.
    - **「上位スロットの選択」コマンドの利用:** 推奨する方法
      - スロットを選択して右クリックで「上位スロットの選択」を実行
      - 表示されるダイアログから上位とするスロットを選択する
    - **「リンクの追加」コマンドの利用 :**
      - Shiftキーを押しながら上位概念のスロット，下位概念のスロットの順に2つ選択し，**「is-aリンクの追加」操作**を実行する.
  - 特殊化済みのスロット(赤で表示)を，特殊化していないスロットに変更する場合
    - スロットを選択して右クリックメニュー「継承情報の削除」を実行

サンプル：揚げ物.xml


Any
super
is-a
sub
料理
材料
p/o 1..
食材
調理法
a/o 1
揚げ物
具 <<料理$材料
衣 <<料理$材料
調理法 <<料理$調理法
揚げる
からあげ
衣 <<揚げ物$衣
唐揚げ粉
てんぷら
粉
小麦粉

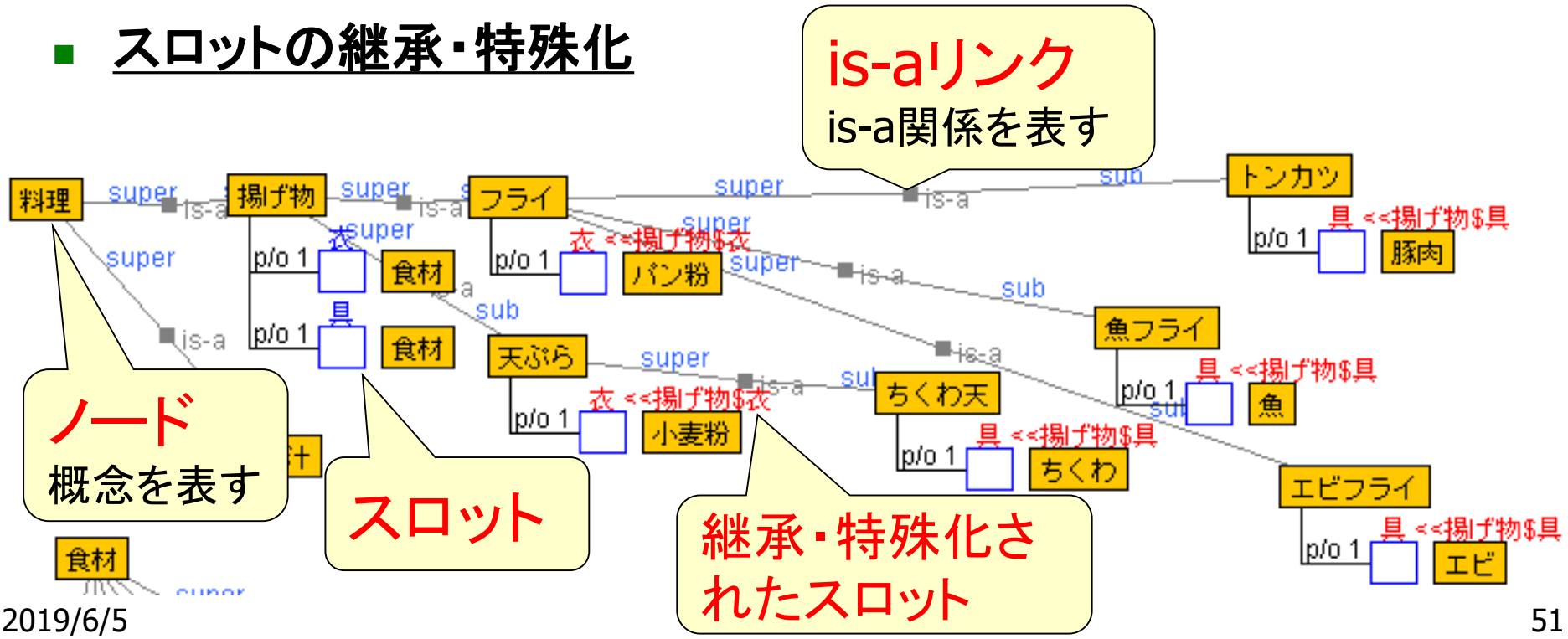
スロット定義の基本的な考え方

- **スロット定義による概念間の共通性・違いの明示**
  - 複数の(下位)概念に共通な性質は上位概念のスロットとして定義する
  - 継承したスロットを下位概念で特殊化することにより,
    - 上位概念と下位概念の違い
    - 兄弟概念(同じ上位概念を持つ概念)間の違い　を明示する
  - 継承・特殊化されたスロット(赤いスロット)が多いオントロジーが好ましい
- **スロット定義とis-a階層**
  - 各概念のスロットを適切に定義することで,is-a階層の分類視点を明示することができる
  - クラス制約で参照する概念は別のis-a階層として定義する.

is-a階層とスロット定義の組み合わせで概念が体系化される

まとめ：オントロジー構築の基礎

- **概念とis-a関係**
  - 概念：ノード・is-a関係：**is-aリンク**で表される
- **スロットの定義**
  - 概念の定義違いを明示できる
  - クラス制約：他の場所で定義された概念を参照する
- **スロットの継承・特殊化**



（再掲）オントロジー構築の基本的な考え方

- **オントロジー構築の基本姿勢**
  - 「何が本質か？」を追求する
    - 例）**人間 vs 教師，もの vs プロセス**（川や滝はどちら？）
    - この基本姿勢が，**データスキーマ**や**シソーラス**などとの違い
  - 「対象世界をどのように捉えたか（概念化したか）を明らかにすることにより，諸概念の共通性と相違点を明確にする」
- **具体的な構築指針**
  - 概念間の意味的相違点（**分類視点**）の明確化
    - 概念の「違い」を理解する（分かる＝分ける）
    - 概念間の違いを明確にすることが，is-a階層構築の基本指針
  - 概念の共通性・本質属性
    - 対象とする概念群に**共通する性質**や**本質的な性質**（**本質属性**）を捉える

共通性・相違性を捉えた概念化例
例)「料理を作る」という行為

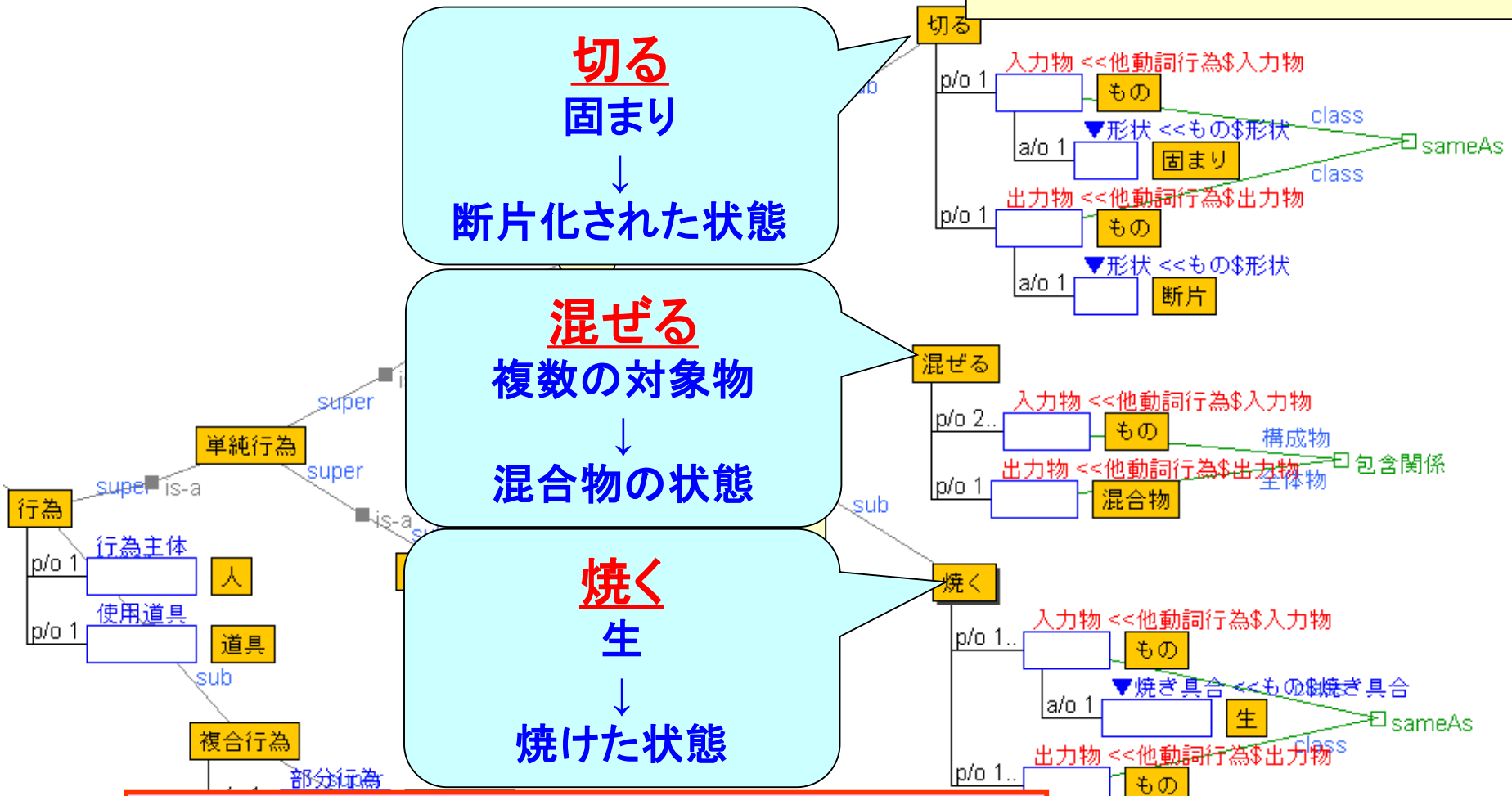
- **「お好み焼き」を作る手順**
  - キャベツ，ちくわなどの材料を「切る」
  - 切った材料に水，小麦粉，卵などを「混ぜる」
  - 混ぜた材料(お好み焼きのもと)を「焼く」

- **「切る」，「混ぜる」，「焼く」といった概念に共通する性質・本質的な捉え方は？**
  - 使う道具，手の動かし方などは，全て異なる
  - 料理を作る際に必要な基本動作(行為)という意味では共通？．．．
  - **対象物に「変化を与える」という点で共通**

    →行為の前後で**「対象物がどのように変化するか」**で捉える(概念化する)ことができる．

「行為」の定義例

サンプル：
(2) 料理する.xml



この概念化は，料理に関わらず，
あらゆる（他動詞）行為に適用可能

構築ガイドライン:is-a階層の推奨基準
is-a階層構築の基礎的指針

- **オントロジーにおけるis-a階層の特徴**
  - 対象世界における概念の性質をできるだけ反映したものであることがのぞまれる.

  →オントロジーにおけるis-a階層は**「単なる語彙分類階層（taxonomy）」ではない.**
- **is-a階層構築（クラス分類）の推奨基準**
  - **基本的な考え方**
    - 上位概念から**「どのような性質が継承されるか？」**を意識する.
    - 同じ上位概念を持つ**下位概念同士の「違い」を明確**にする.
  - **クラス分類基準の同一性**
    - is-a階層を構築する（下位概念を定義する）時の基準（視点）は一定にする.
  - **インスタンス集合のパーティション性**
    - 下位概念間のインスタンス集合の重なりを避ける.
  - **本質的な属性による分類**（何を本質とするかは恣意的）
  - **分類の上下関係**
    - 分類観点に従属性があればそれに従う.
    - 一般性の高い性質は上位概念で定義する.
    - 多重継承の利用はできるだけ避ける.

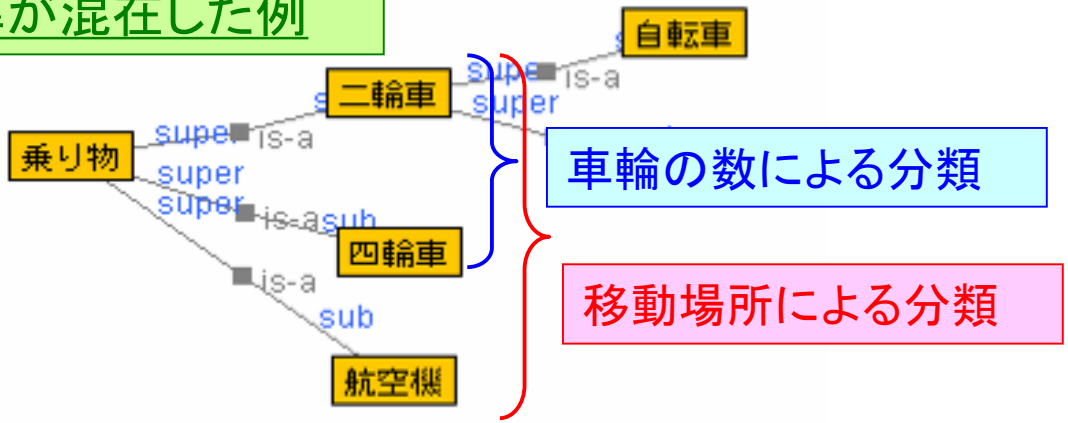
法造では，多重継承を用いる場合にはIS-A関係を利用するただし，*ロール概念を用いると多重継承を使わずに適切な概念定義ができる場合が多い*.

概念の分類視点の明確化
is-a階層構築の基礎的指針

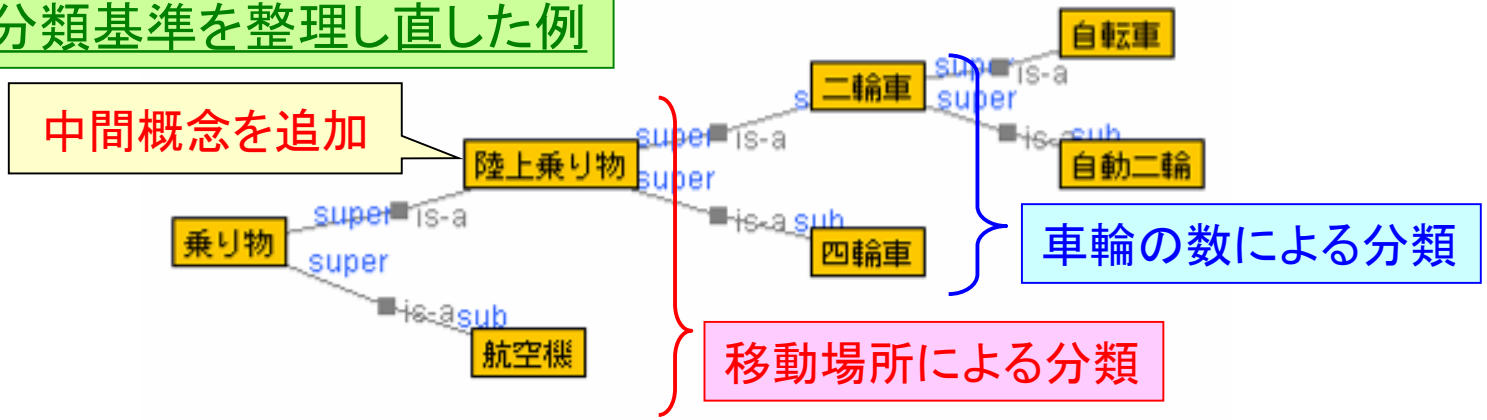
- **クラス分類基準の同一性**

  is-a階層を構築する（下位概念を定義する）時の基準（視点）は一定にする.

分類基準が混在した例



分類基準を整理し直した例

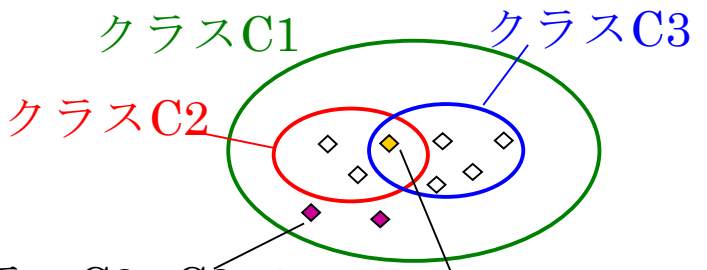


概念の分類視点の明確化
is-a階層構築の基礎的指針

- インスタンス集合のパーティション性

下位概念間のインスタンス集合の重なりを避ける.

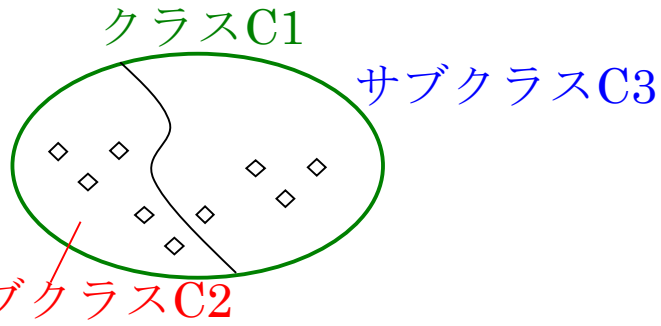
パーティションではない場合



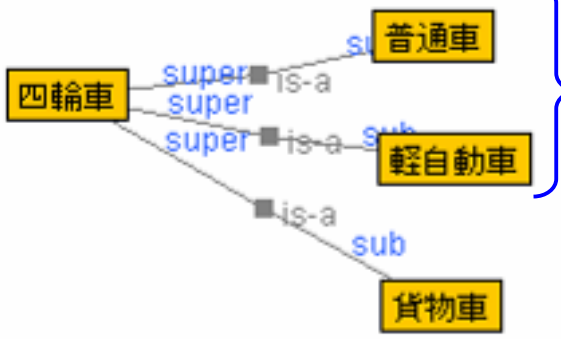
サブクラスC2, C3 のどちらにも属さないC1のインスタンス

サブクラスC2, C3の両方に属するインスタンス

パーティションである場合



パーティションではないis-a階層の例



排気量による分類

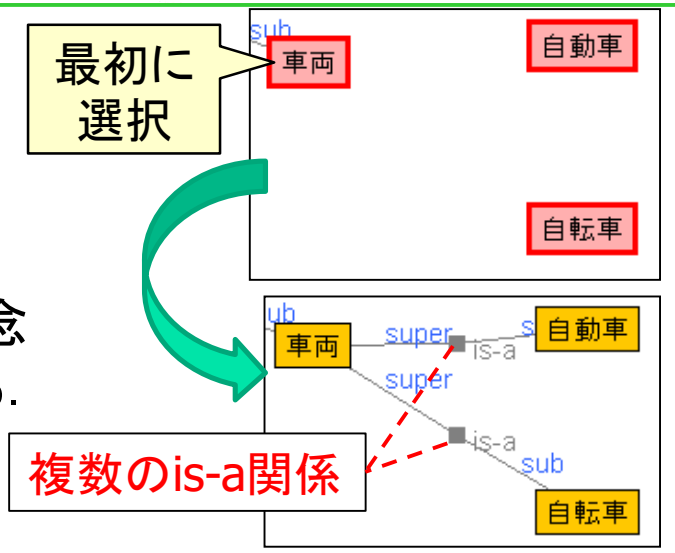
用途による分類

修正例）
- 「貨物車」を「普通貨物車」と「軽貨物車」に分けて，それぞれ「普通車」，「貨物車」の下位概念とする.
- 「貨物車」をロール概念として定義する

補足説明:
is-a階層の構築・修正の操作

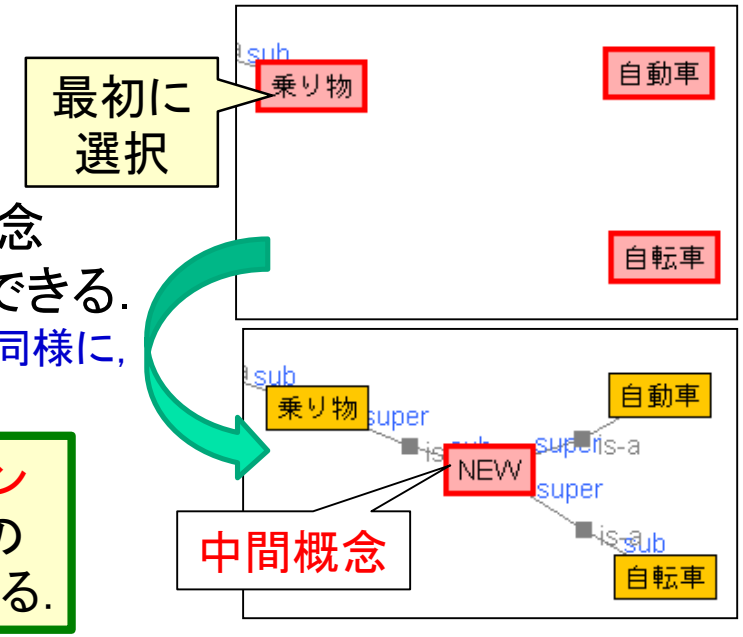
- **複数のis-a関係の同時追加**
  - 3つ以上のノード(概念)を選択した状態で
    [is-aリンクの追加]を実行すると,
    - 最初に選択したノードを上位概念
    - 残りすべての選択中のノードを下位概念

    とした「複数のis-a関係」を同時に追加できる.



- **中間概念の挿入**
  - 2つ以上のノード(概念)を選択した状態で
    [ノードの追加]を実行すると,
    - 最初に選択したノードを上位概念,
    - 残りすべての選択中のノードを下位概念
  - とした「中間概念」を新規ノードとして追加できる.

※既にis-a関係を持つ場合, 通常のis-a関係の追加と同様に,
多重継承となるis-a関係は自動的に削除される.



これらの操作を組み合わせることで, トップダウン(上位から下位), ボトムアップ(海外から上位)の双方向からis-a階層を構築・修正することができる.

オントロジーの構築方法②
～より詳細な概念定義～

- スロットの詳細定義
  - ロール概念，ロールホルダーの定義
  - ロール概念の性質
  - ロールに基づく概念化
- スロット間の関係に関する制約

概念の基本的な性質の違い

- 例）下記の概念を性質の違いで2つに分けると？
  - ほ乳類
  - 人間

  基本概念（ロールでない普通の概念）
  ＝コンテキストに依存せずに単独に定義できる概念

  - 教師
  - 歩行者
  - 殺人者
  - 母親

  ロール
  ＝コンテキスト（他者）に依存する概念

教師＝「学校」に依存→学校を辞めると教師でなくなる
殺人者＝「殺人行為」に依存（※やめれないロール）

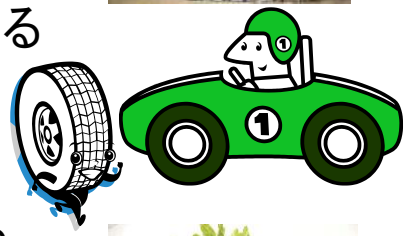
ロール(概念)と基本概念

- **ロール**
  - 特定のコンテキスト(他の概念や関係，状況など)に依存して定義される概念
    - 例:(学校における)教師，(夫婦関係における)夫，
      (自転車の)前輪，(唐揚げの)具
- **基本概念**(ロールでない普通の概念)
  - コンテキストに依存せずに単独に定義できる概念
    - 例:人間，木，石，車輪，...
- **法造におけるロールの定義**
  - そのコンテキストにおける役割(ロール概念)，その役割を担う概念(プレイヤー)，役割を担った状態の概念を表すロールホルダーから定義される.
  - 今回は，基本的な記述方法についてのみ解説する.

ロール概念とは

ロール概念：
ある“もの”や“人”が特定の状況で「果たす役割」を概念化したもの

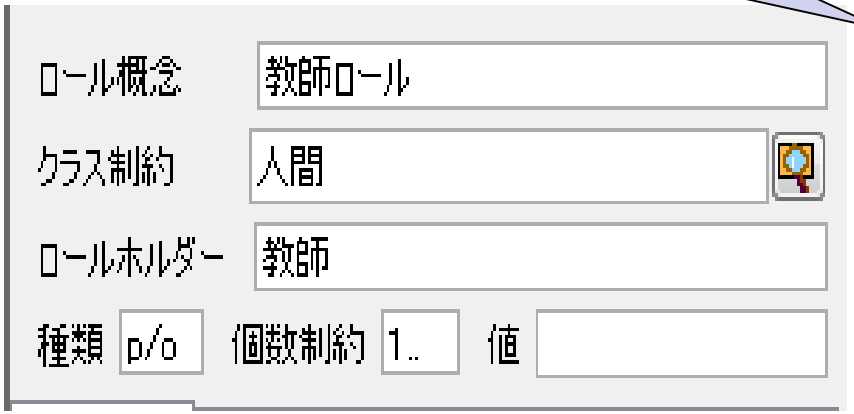
コンテキスト	ロール概念	プレイヤー
夫婦関係		
夫婦 における	夫(妻)の役割 は 夫・妻ロール	人間 が担っている
学校		
学校 における	教師の役割 は 教師ロール	人間が担っている
自動車		
自動車 における	前輪(後輪)の役割 は 前輪(後輪)ロール	車輪 が担っている
鳥の唐揚げ		
鳥の唐揚げ における	具の役割 は 具ロール	鶏肉 が担っている



ロール概念・クラス制約・ロールホルダー

ロール概念が依存しているコンテキスト（概念や関係）

ロール概念
ある“もの”が特定のコンテクストのもとで果たす役割を捉えて概念化したもの



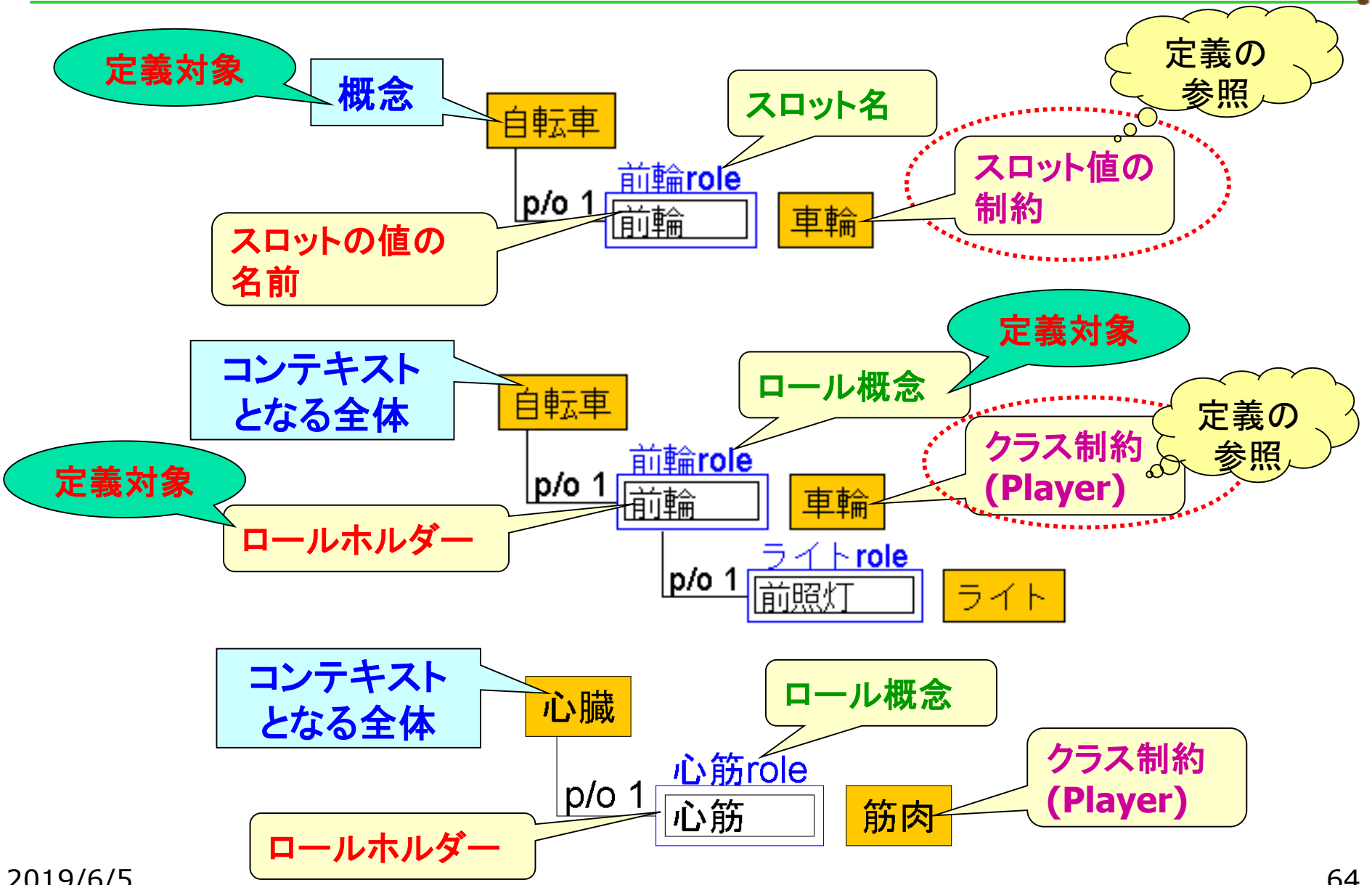
ロールホルダー
ロール概念で定義した役割を担っている概念

クラス制約（潜在的プレイヤー）
ロール概念で定義した役割を担うことができる概念

コンテキスト　プレイヤー　ロール概念　ロールホルダー

学校において、人間が教員ロールを担ったとき教員ロールホルダーと呼ばれる

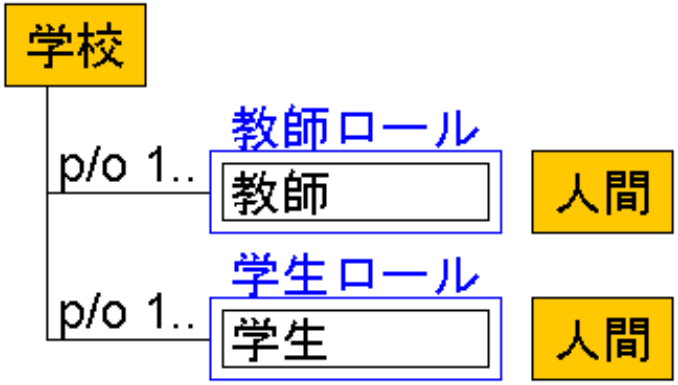
ロール概念の記述

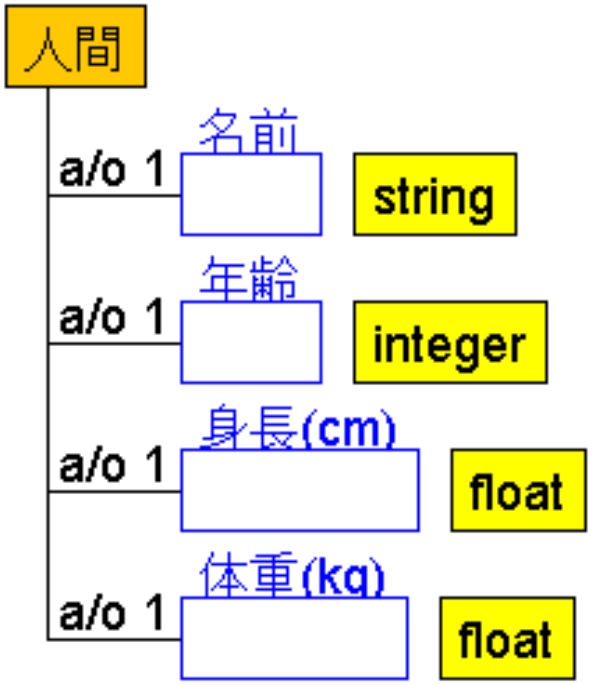

定義対象
概念
自転車
スロット名
定義の参照
前輪role
p/o 1
前輪
車輪
スロット値の制約
スロットの値の名前
コンテキストとなる全体
自転車
定義対象
ロール概念
定義の参照
前輪role
p/o 1
前輪
車輪
クラス制約(Player)
定義対象
ロールホルダー
ライトrole
p/o 1
前照灯
ライト
コンテキストとなる全体
心臓
ロール概念
心筋role
p/o 1
心筋
筋肉
クラス制約(Player)
ロールホルダー

練習操作：ロール概念の定義

サンプル：
学校.xml

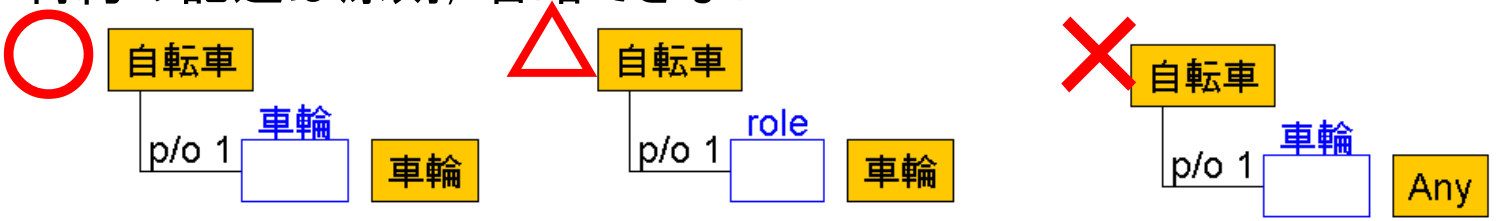
- オントロジー「人間.xml」を開き，先の例にならい，「学校」をコンテキストとした
  - 教師ロール・教師ロールホルダー
  - 学生ロール・学生ロールホルダー　を定義する.



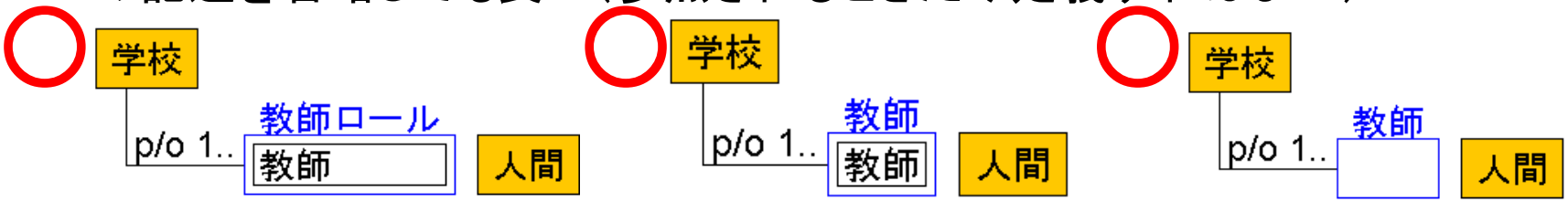


補足説明:
ロール概念定義の注意事項

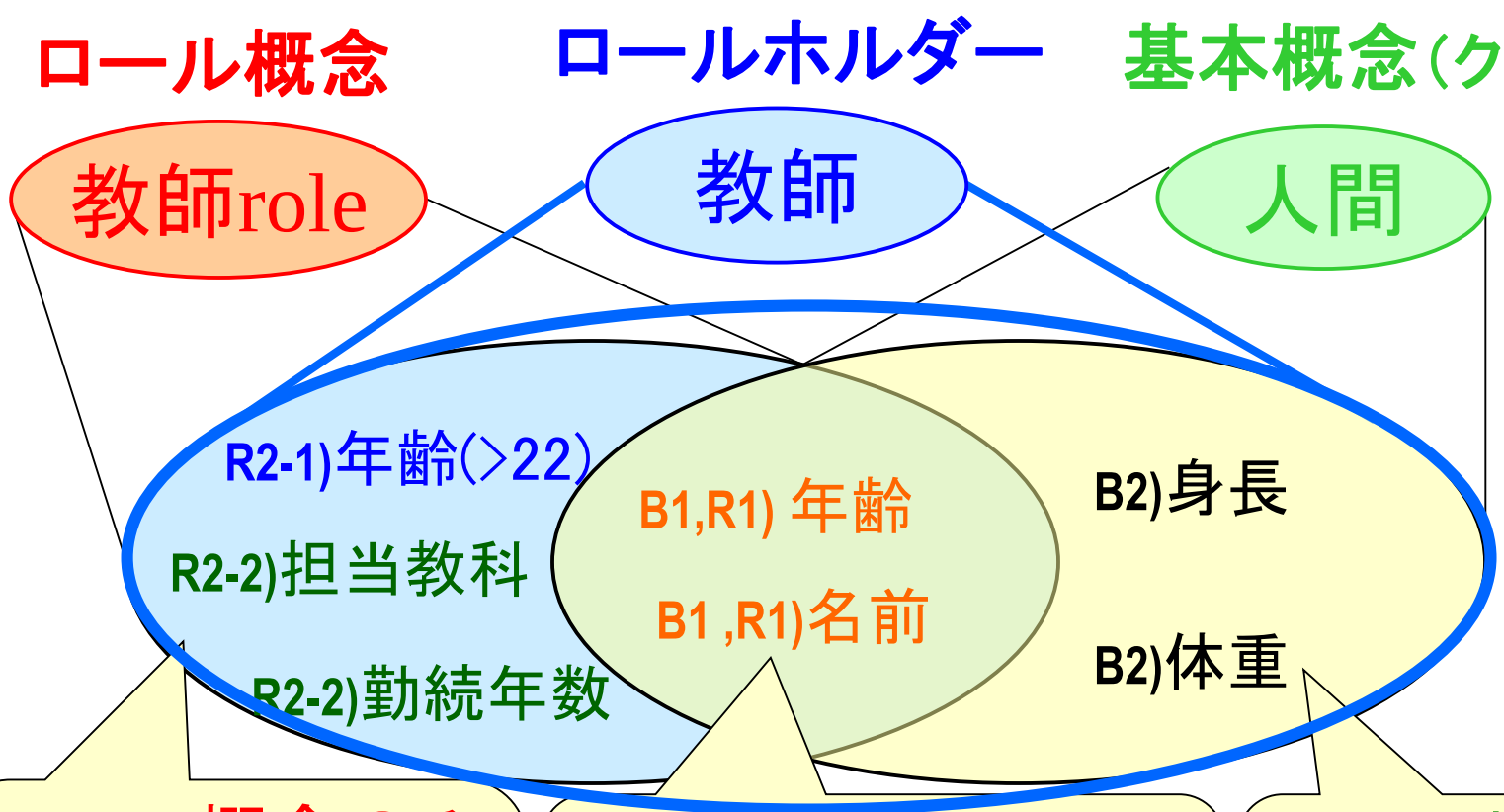
- **ロール概念とクラス制約**
  - 適切なロール概念名が思いつかない場合，ロール概念名とクラス制約を同じにする（推奨）か，ロール概念名の記述は省略しても良い．
  - クラス制約の記述は原則，省略できない．



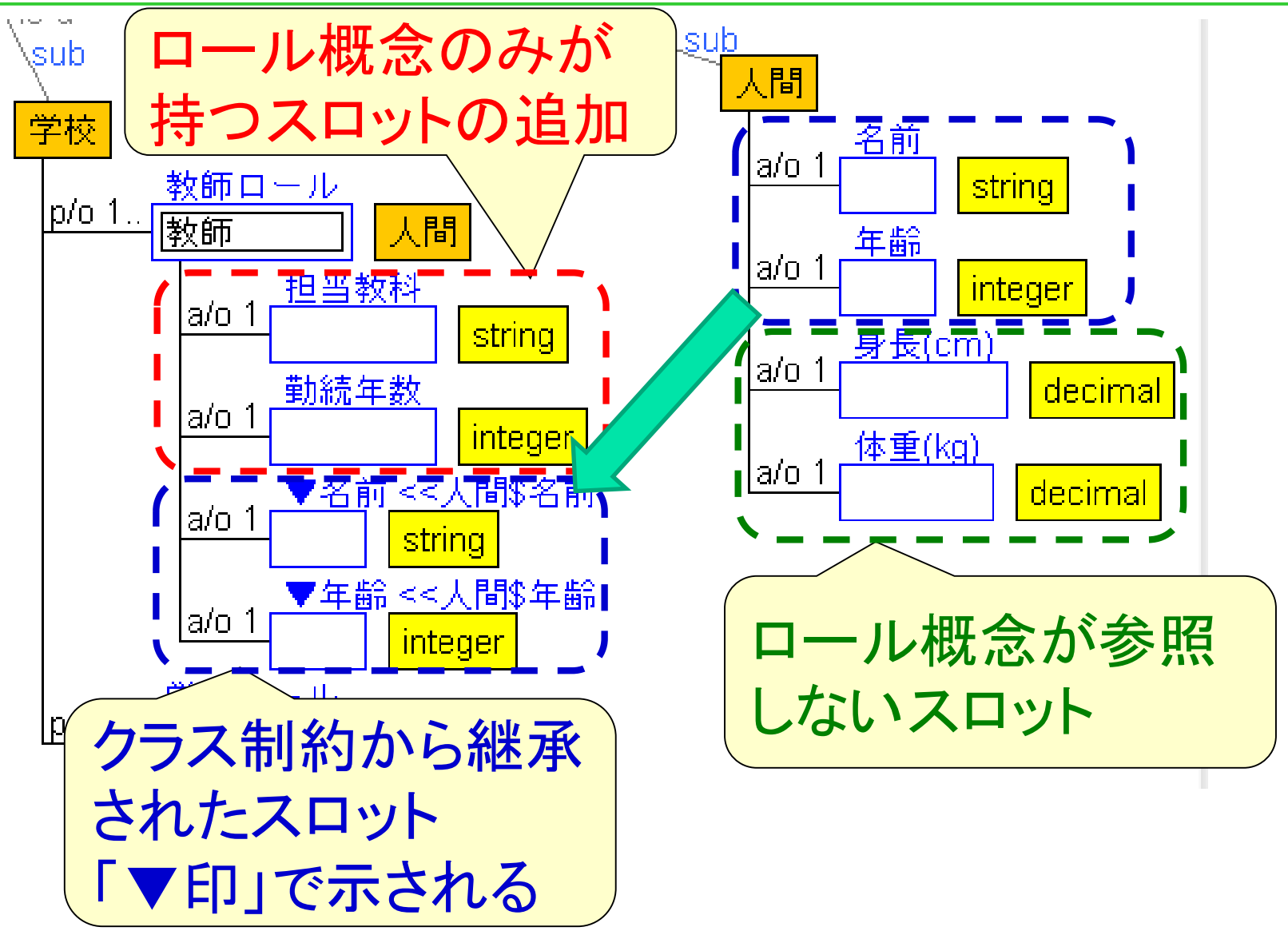
- **ロール概念とロールホルダー**
  - ロール概念とロールホルダーは同名にしてもよい．（その場合でも，法造の内部では別のidで区別して扱われる）
    - 明示的に区別したい場合は，ロール概念名を「〇〇ロール」とする．
  - ロールホルダーが他の箇所で参照されない場合は，ロールホルダー名の記述を省略しても良い（参照されるときだけ定義すればよい）．



ロール概念定義の詳細


ロール概念
ロールホルダー
基本概念（クラス制約）
教師role
教師
人間
R2-1)年齢(>22)
R2-2)担当教科
R2-2)勤続年数
B1,R1) 年齢
B1 ,R1)名前
B2)身長
B2)体重
ロール概念のみが持つスロット（ロール概念で追加された）
クラス制約から継承されたスロット
ロール概念が参照しないスロット
合わせたもの

例）ロール概念の定義・クラス制約からの継承

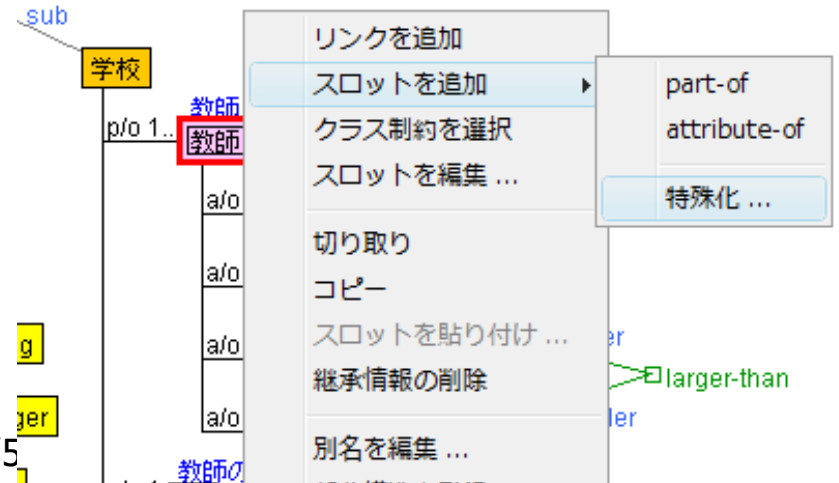


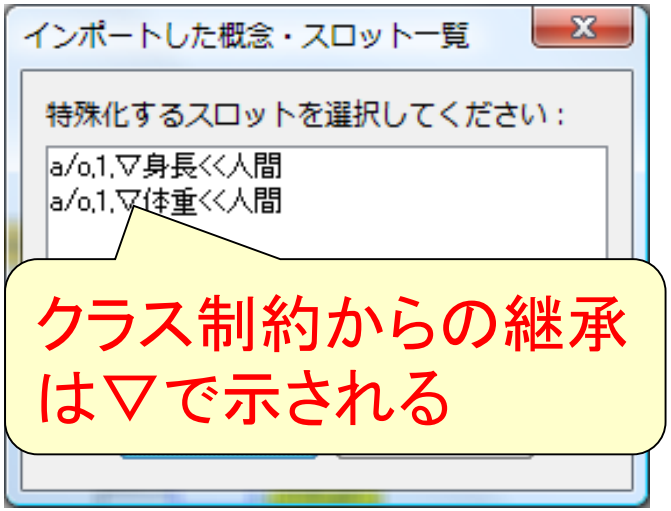
ロール概念のより厳密な定義

- **ロール概念のみが持つスロットの追加**
  - スロットを選択した状態で[**スロットの追加**]を実行することで　ロール概念のみが持つスロットが定義できる

  例）教師ロールの「担当教科」,「勤続年数」
- **クラス制約からロール概念へのスロットの継承・特殊化（クラス制約からの継承）**
  - スロットを選択し，右クリックメニューの「スロットの追加」→「特殊化」を実行する．　例）教師ロールの「名前」,「年齢」
  - クラス制約から継承したスロットの定義を編集する

    例）教師ロールの「年齢」

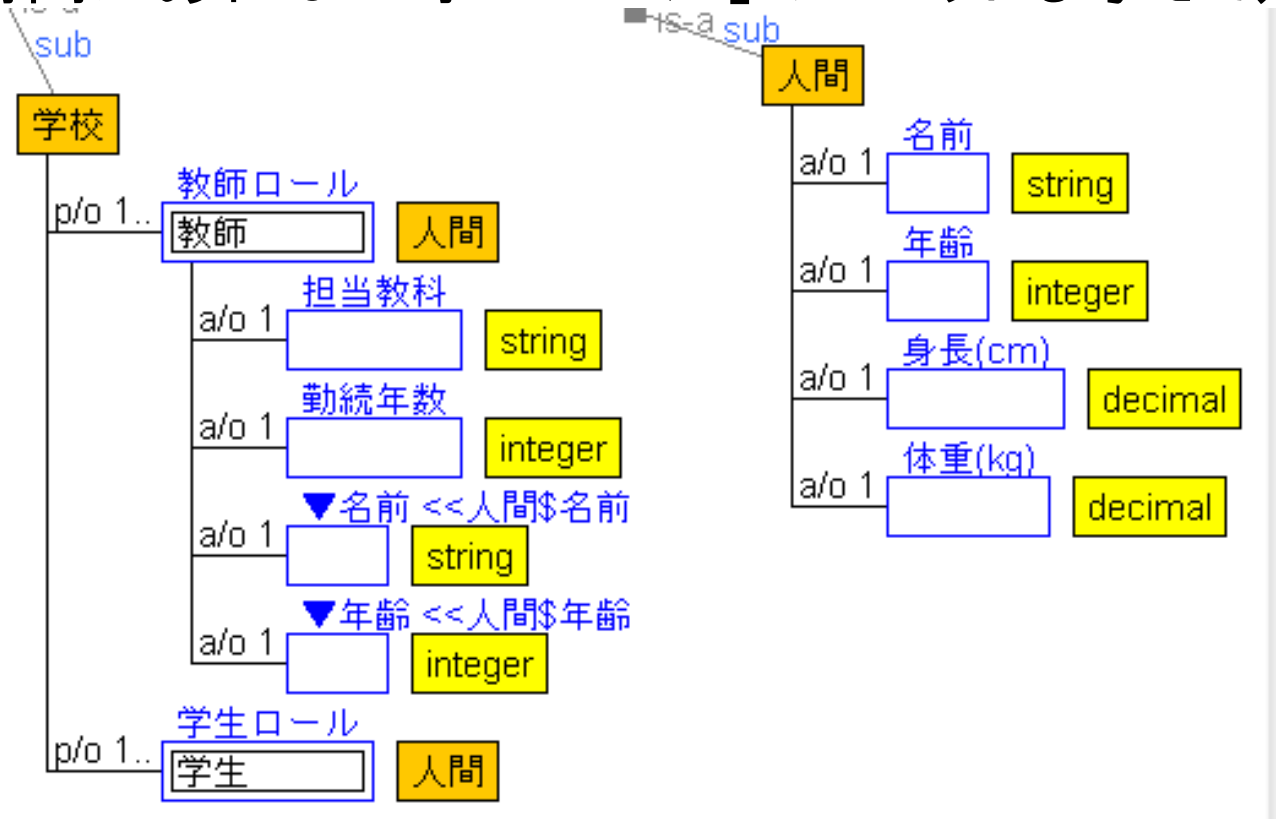




練習操作

サンプル：
学校＋ロール.xml

- **教師ロールの詳細定義**
  - 担当教科，勤続年数スロットを追加
  - クラス制約からの継承で「年齢」「名前」スロットを追加
  - 時間があれば 「学生ロール」のスロットも考えて定義する



オントロジーの構築方法②
～より詳細な概念定義～

- スロットの詳細定義
  - ロール概念，ロールホルダーの定義
  - ロール概念の性質
  - ロールに基づく概念化
- スロット間の関係に関する制約

ロール概念の主な性質

- **コンテキスト依存性**
  - 1つインスタンスがコンテキストに応じて，複数のロールを同時に担うことができる
    - コンテキストが変わればロールが変わる

    例）太郎が学校では「教師」，夫婦の間では「夫」
  - コンテキストが消滅すればロールが消滅する
    - 例）学校が廃校になれば，「教師」はなくなる．
- **プレイヤー（クラス制約）依存性**
  - プレイヤーが消滅するとロールホルダーは消滅する
    - 例）太郎が死ねば，（太郎が担っていた）「教師」はなくなる．
- **定義内容の重なり**
  - ロール概念からのみ決まる定義と，クラス制約（プレイヤー）にのみ依存して決まる定義がある

ロール概念の性質

- **ロール概念のコンテキスト依存性**

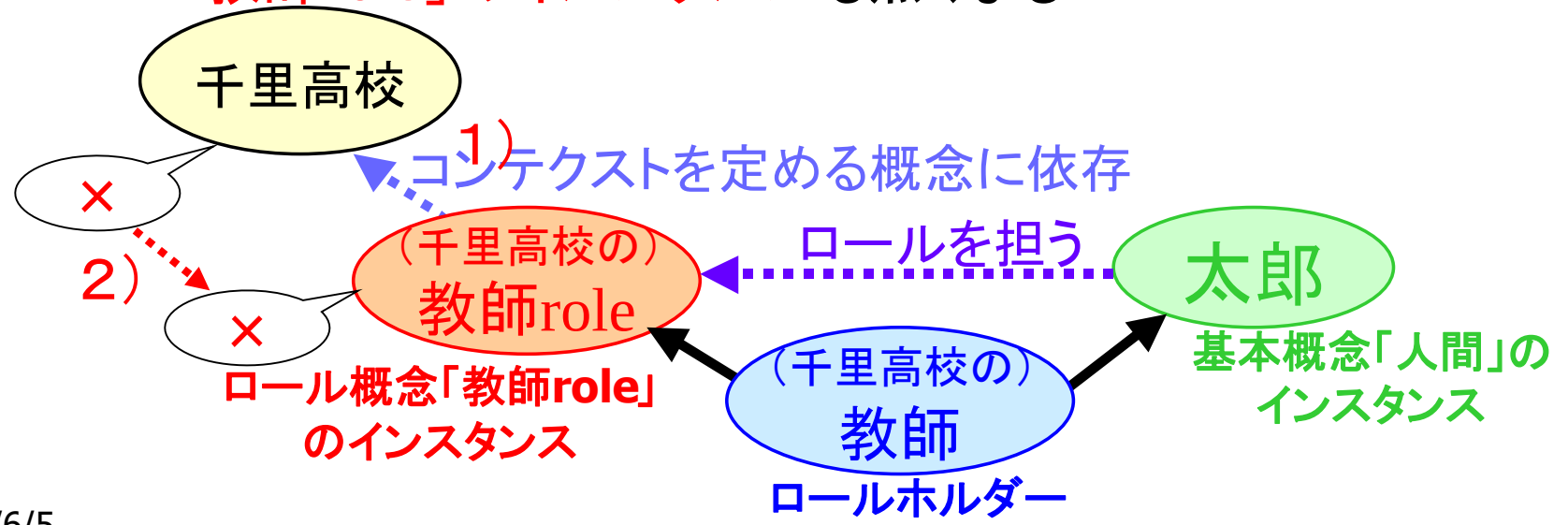
  1）ロール概念のインスタンスは，特定のコンテキストに依存して具体化される

  例）「教師role」のインスタンスは「学校」のインスタンス　によって定まるコンテクストに依存して具体化される.

  2）コンテクストが無くなると，ロール概念のインスタンスも消滅する

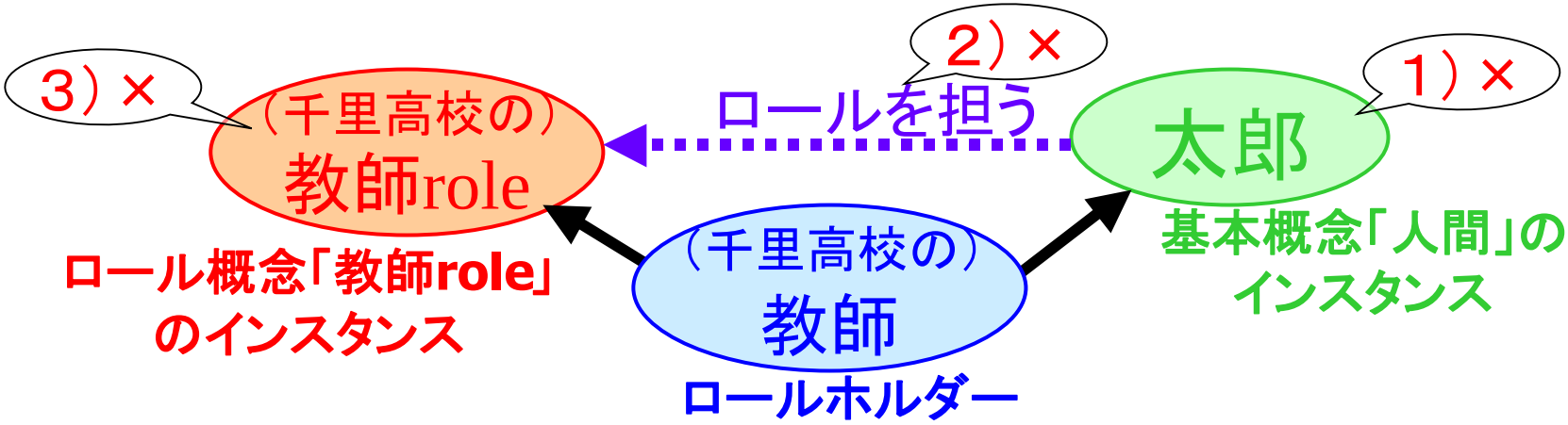
  例）「学校」が廃校になると，それに伴い

  「教師role」のインスタンスも無くなる.



ロールホルダーの性質

- **ロールホルダー**が消滅する3種類のパターン
  1）「太郎」が死ぬと（＝人間のインスタンスが消滅）
    - 「教師」は消滅するが，「教師role」は空きポストとして残る
  2）「太郎」が教師を辞めると（＝ロールを担うことを止める）
    - 「教師」は消滅するが，「教師role」は空きポストとして残る
    - 「人間」のインスタンス（太郎）はそのまま存続
  3）「教師role」が無くなると（＝ロール概念のインスタンスの消滅）
    - 「教師」は消滅するが，「人間」（太郎）はそのまま存続



ロール概念を用いる意義

- **概念のコンテキスト依存性を明確にする**
  - コンテキスト(他の概念)が無いと定義できない概念と，コンテキストに独立して定義できる概念を区別する.
    - 例）教師，学生，夫vs.人間　　前輪，後輪，駆動輪vs.車輪

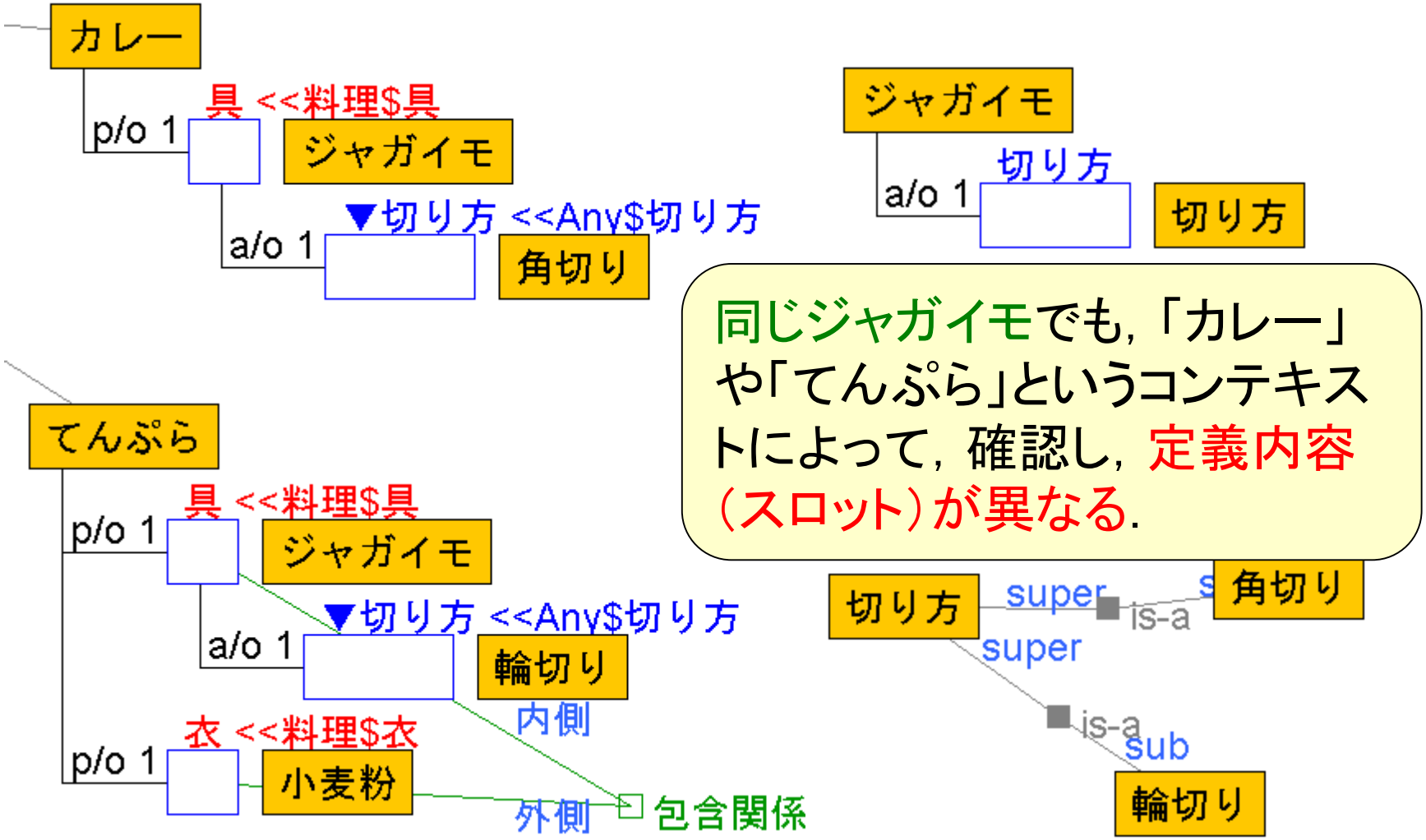
人間は，コンテキストによって，教師，学生，夫…など別のロール(役割)を果たす.

車輪は，コンテキストによって，前輪，後輪，駆動輪…など別のロール(役割)を果たす.

  - クラス制約が同じでも，コンテキスト(＝どの概念のスロットか)によってロール概念の定義が追加される.
    - コンテキストに独立して共通な性質は，基本概念の定義
    - コンテキストに依存して決まる性質は，ロール概念の定義）

    と，適切に使い分けることが重要

料理におけるロールの例



オントロジーの構築方法②
～より詳細な概念定義～

- スロットの詳細定義
  - ロール概念，ロールホルダーの定義
  - ロール概念の性質
  - ロールに基づく概念化
- スロット間の関係に関する制約

全体と部分の相互依存性

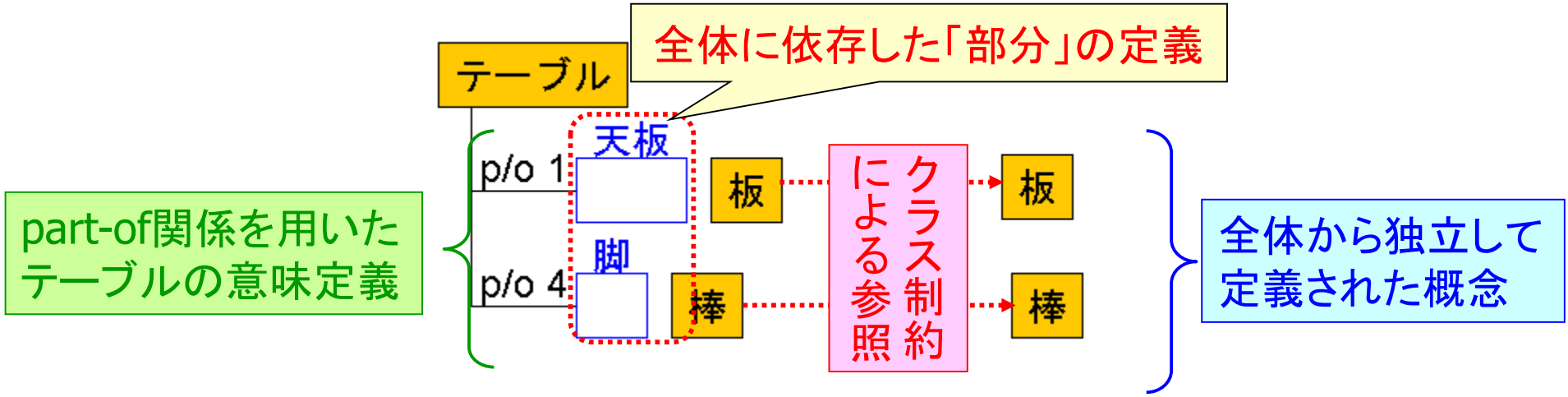
- **全体の部分依存性**
  - 全体となる概念は，部分（部品）の集合として定義され
    例）「テーブル」は，「天板」と4本の「脚」という**部品から構成される**
- **部分の全体依存性**
  - 部分となる概念（部品）は，全体から定義される
    - 例）「天板」や「足」は，「テーブル」**（という全体）の部品である**.
- **全体から独立した部分の定義**
  - 部分となる概念は，他の箇所で定義された概念を参照して定義される
    - 例）ある「板」や「棒」が「テーブル」（全体）となるときは「天板」や「脚」となる.
    - →「板」や「棒」が単独で存在するときは，「天板」や「脚」ではない.

ロール概念を用いることで，これらの違いを適切に概念化できる

ロールに基づく全体-部分の概念化

- **全体の部分依存性**
  - part-ofスロット（関係）を用いた概念の意味定義
- **部分の全体依存性**
  - part-ofスロット（関係）の定義
- **全体から独立した部分の定義**
  - クラス制約による参照

全体をコンテキストとしたロール概念としての扱い

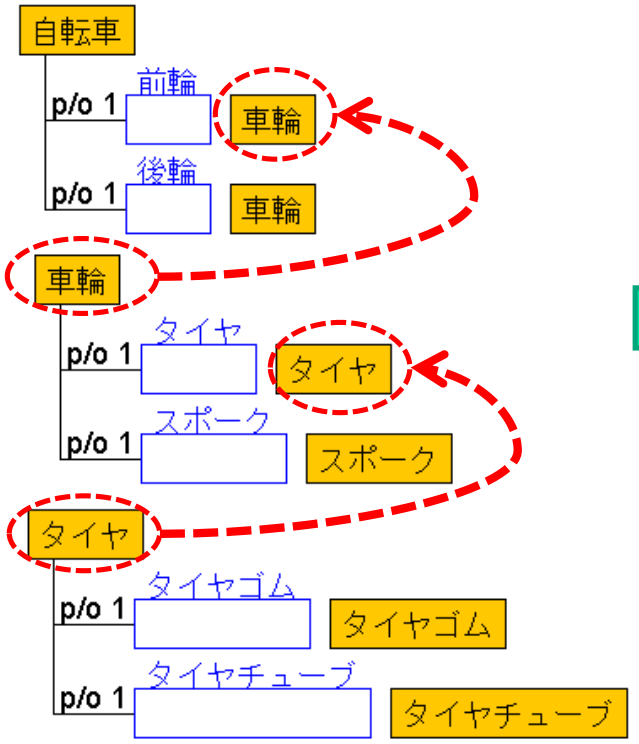


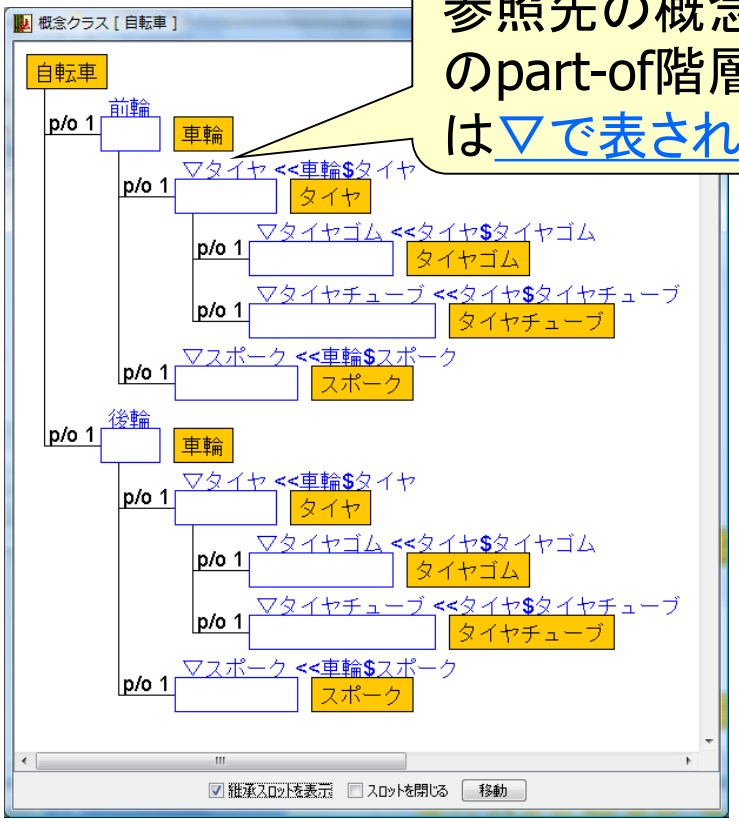
part-of階層の定義（1）

- **part-of階層**

『オントロジー構築入門』4.3.1

  - part-of（p/o）スロットがクラス制約で参照している概念を順に辿ることで，**part-of関係に基づく階層**が形成される．
    - part-of階層を見たいときは，概念を選択して「別ウィンドウに表示」を実行し，「継承スロットを表示」にチェックを入れる．

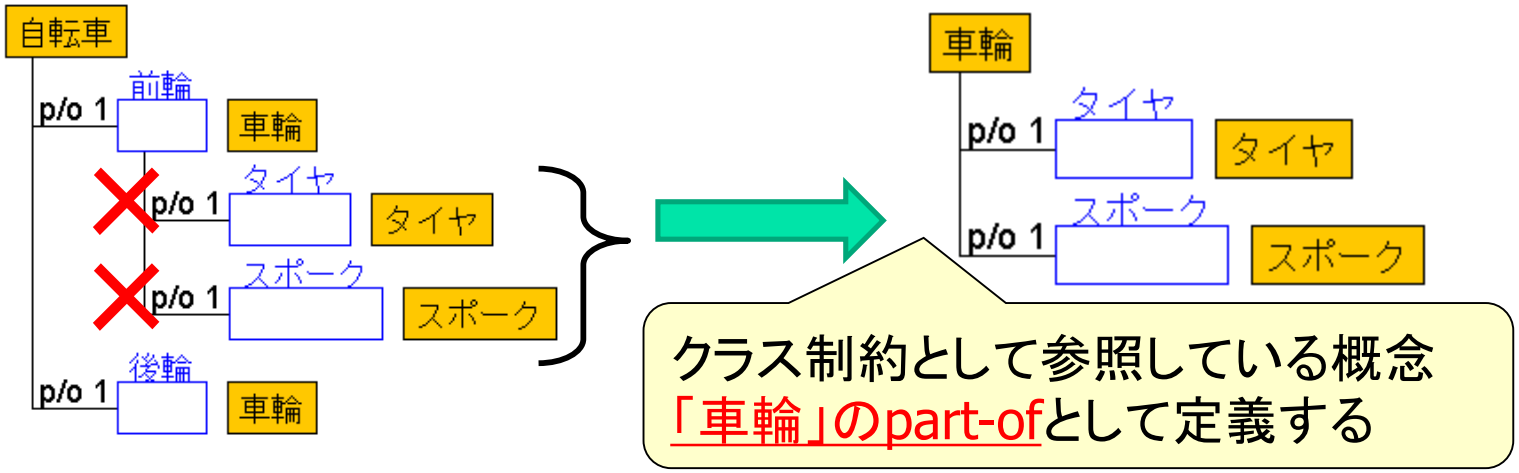




参照先の概念のpart-of階層は▽で表される

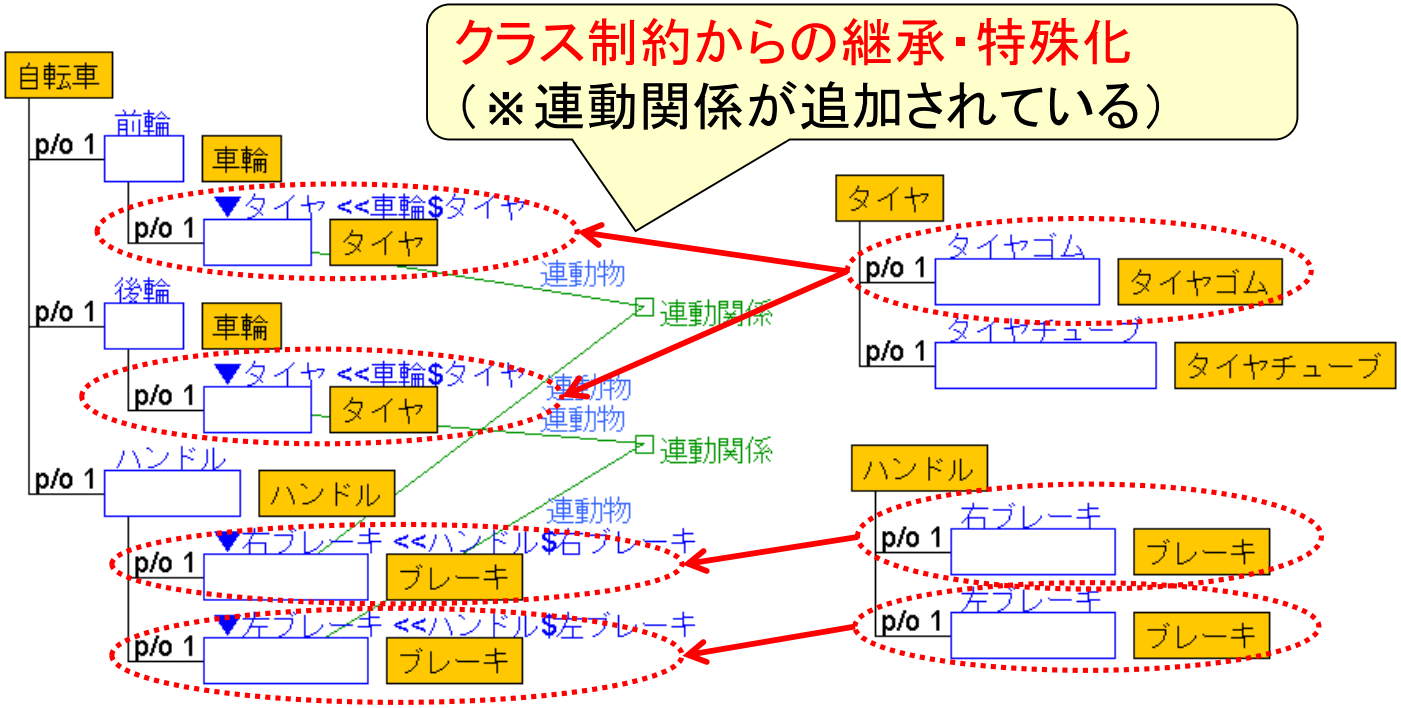
part-of階層の定義（2）

- **スロットが持つスロットの追加**
  - スロットにもノードと同様にスロットを追加できる
    - 新しいスロットを追加
    - クラス制約から継承したスロットの特殊化
  - これは，全体（コンテキスト）に依存した部分や属性（ロール概念）の詳細定義の一部となる．（応用編で解説）
  - クラス制約で参照している概念で定義すべきスロットを，スロットが持つスロットとして定義しないよう注意する．



part-of階層の定義（3）

- **クラス制約で参照している概念が持つスロットの継承・特殊化の操作**（クラス制約からの継承・特殊化）
  - スロットを選択し，ツールバーのスロットの種類で「特殊化」を選択した状態で [スロットを追加]を実行する.
  - クラス制約から継承・特殊化したスロットは「▼」で示される



共通性・コンテキスト依存性の扱い①
－上位概念からの継承・特殊化－

共通の概念構造を上位概念で定義

下位概念

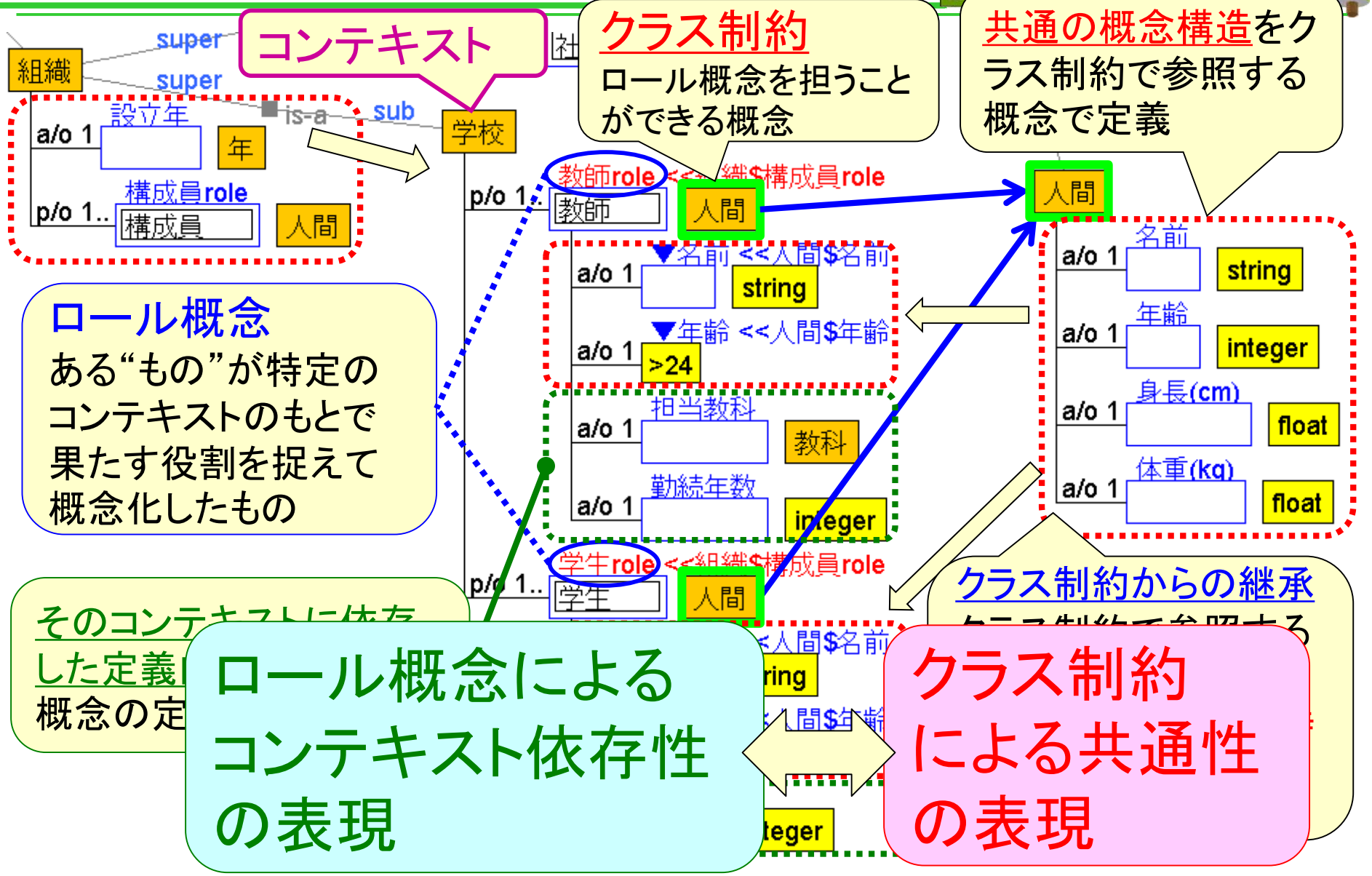
継承された定義内容の一部は，下位概念で特殊化される．

上位概念からの継承
上位概念の定義内容は下位概念に継承される．

上位概念による共通性の表現

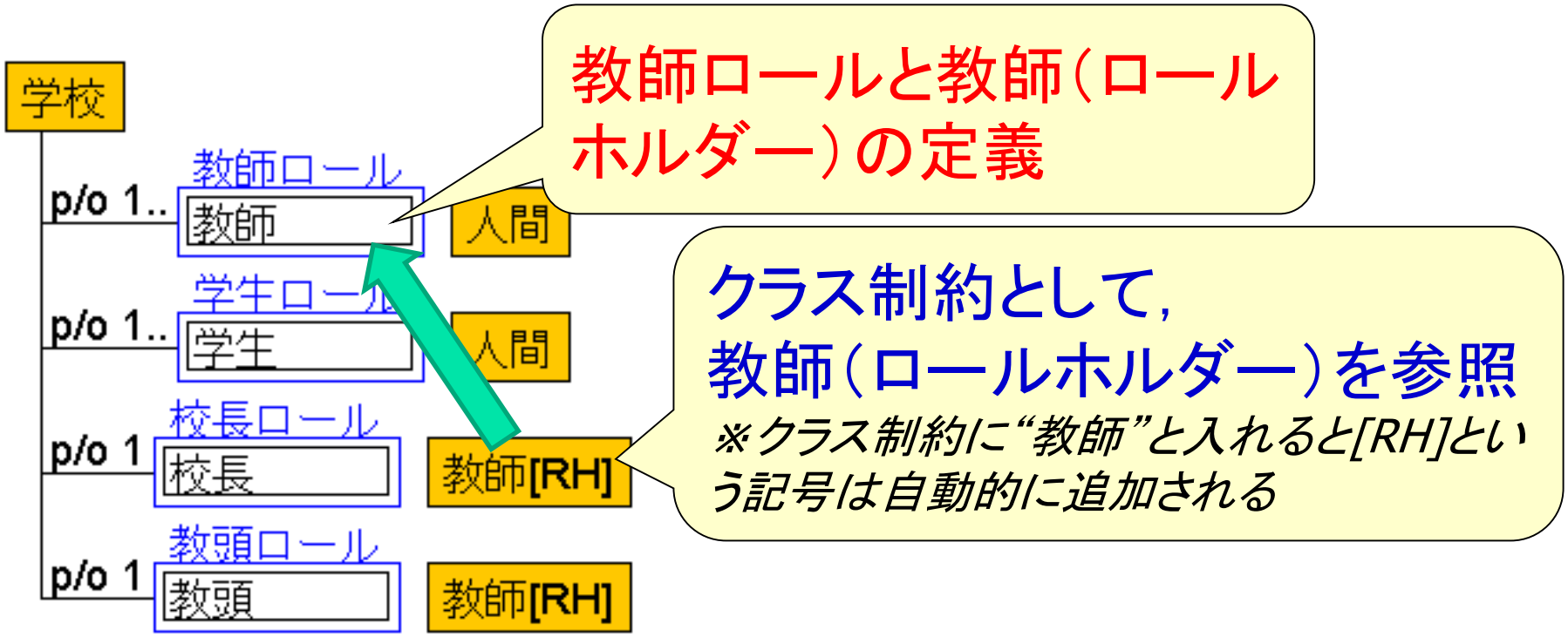
下位概念による特殊性の表現（コンテキスト依存性）

共通性・コンテキスト依存性の扱い②
−クラス制約からの継承・特殊化−



ロール概念の利用例 クラス制約となるロールホルダー

- ロールホルダーをクラス制約で参照し，別のロール概念を定義することができる．
  - 例）学校の校長ロールを担うのは教師（ロールホルダー）



操作練習：教師をクラス制約とする校長・教頭を定義する．

まとめ
ロール概念　vs　ロールホルダー

- *自然言語上の名前は区別されないことがほとんど*
- **ロール概念**
  - 他の概念への依存性
    - コンテキストに依存して存在
    - プレイヤーがなくても存在可能
  - 定義内容
    - ロール概念のみが持つ定義内容の追加が可能　例）勤続年数
    - クラス制約からの継承を介してプレイヤーの定義内容を参照可能

      ※ロール概念で参照されない定義内容もある

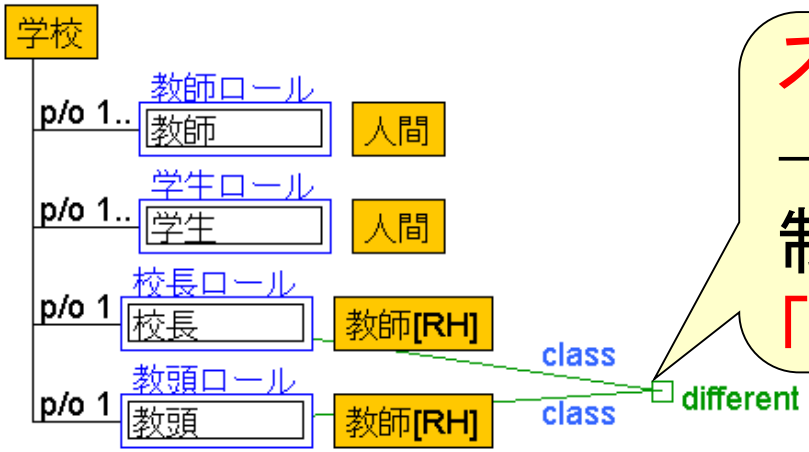
      →これがないとき，ロール概念とロールホルダーの定義が同一になる
- **ロールホルダー**
  - 他の概念への依存性
    - ロール概念（＋コンテキスト）＋プレイヤーに依存して存在
  - 定義内容
    - プレイヤー（クラス制約）の定義＋ロール概念の定義で決まる
    - 独自の定義内容の追加が不可

オントロジーの構築方法②
～より詳細な概念定義～

- スロットの詳細定義
  :ロール概念，ロールホルダーの定義
- スロット間の関係に関する制約

スロット間の関係の利用

- **スロット間の関係を利用することで，より詳細な定義ができる．**
  - 例）学校において，「校長と副校長は異なる人間がなる」



スロット間の関係
→スロットに入る値に関する制約を与える
「関係に関する制約」と呼ぶ

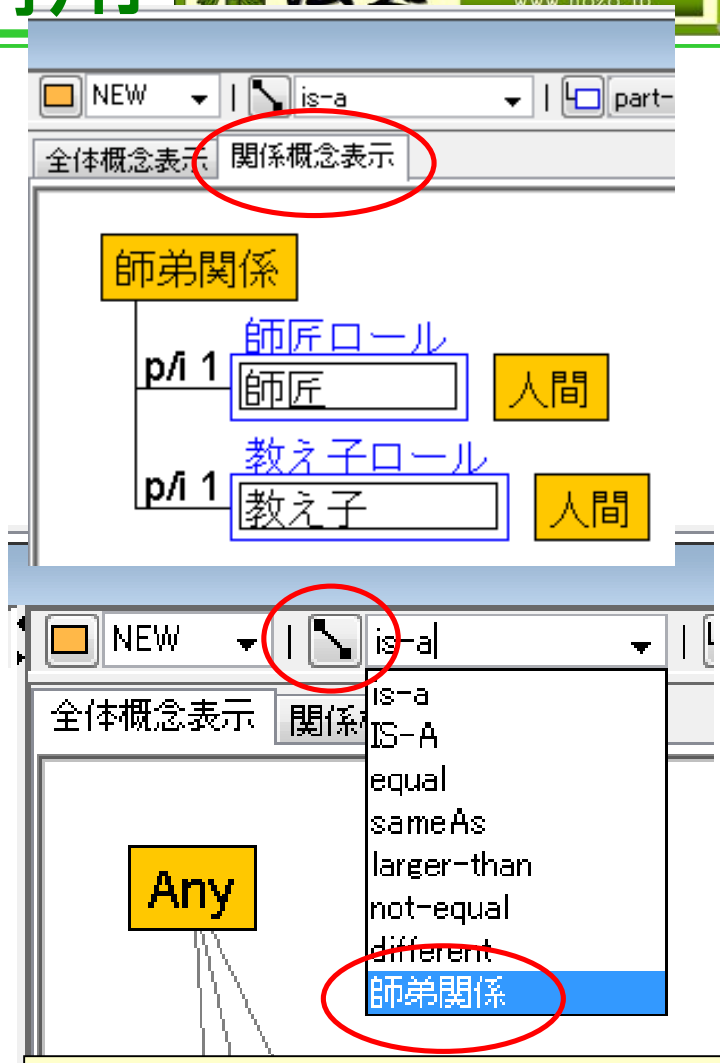
- **法造であらかじめ定義している関係**
  - equal関係：数字（number）が等しい
  - not-equal関係：数字（number）が異なる
  - larger-than関係：数字（number）の大小
  - sameAs関係：同じインスタンスが入る
  - different関係：異なるインスタンス入る

編集操作
1)SIFTキーを押しながらスロットを複数選択する．
2)追加したい関係をリストで選び[**リンクの追加**]ボタンを押す．
3)ダイアログで**追加内容を確認**してOKボタンを押す．

操作練習：校長・教頭の間にdifferent関係追加する．

その他の関係の定義と利用

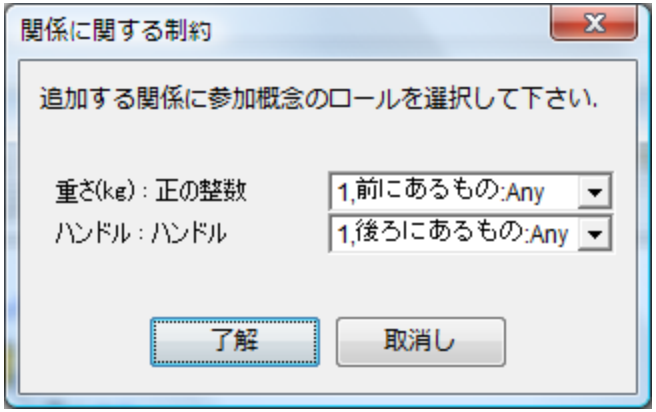
- **その他の関係の定義**
  - 新しい関係を定義することができる
  - 「関係概念」タブを選択
    →概念と同様にノードを用いて定義
    →定義した関係は「リンクの種類」に追加される
- **定義した関係の利用**
  - 定義した関係概念は，スロットで示される概念の間に必ずその関係が成立するという関係に関する制約として利用できる．
    - SIFTキーを押しながらスロットを複数選択し，追加したい関係を選び，[リンクの追加]ボタンを押す．
    - ダイアログで追加内容を確認してOKボタンを押す．



操作練習：教師・学生の間に師弟関係を追加する．

関係概念利用の注意点

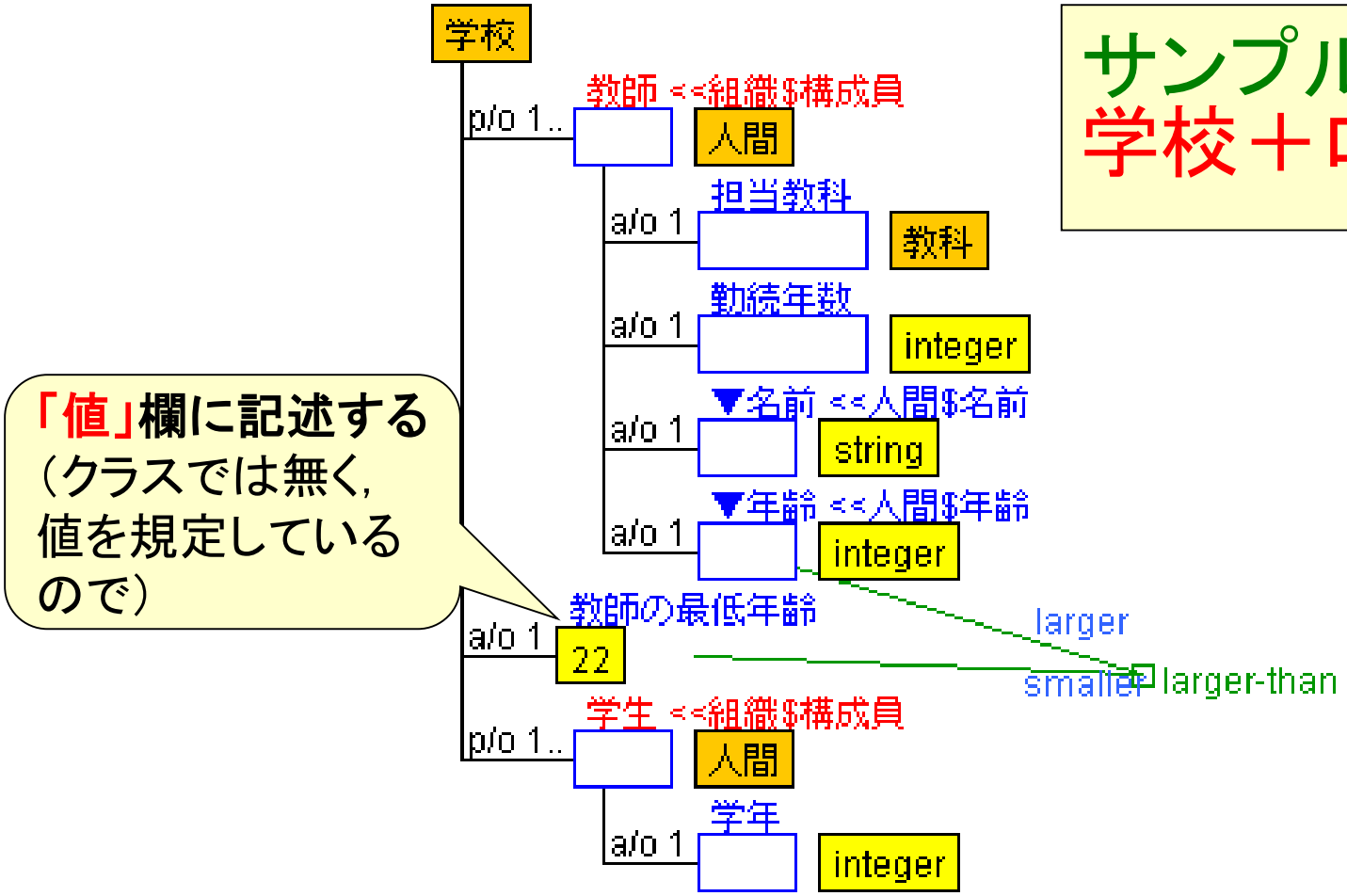
- **関係利用時の制約事項**
  - **個数制約**：関係に参加する概念（スロット）の数を規定
  - **クラス制約**：関係に参加する概念のクラスを規定
    - スロット間に関係を引くときは，**各スロットのクラス制約が，関係概念で定義したクラス制約を満たす**必要がある．



- **ノード（概念）間の関係**
  - **ノード（概念）間の関係を直接引くことはできない**．
    - 概念間に成り立ちうる関係（例：「人間」と「もの」の間の「所有関係」）は，**関係概念の定義**として表される．
      - ノード間の関係はモデルエディタで**インスタンス間の関係**として引く．
    - 内容によっては，スロットの**ロール概念を用いて記述**する．

例）関係を使った詳細な制約

- 例）「larger-than関係」を定義し教師roleの「年齢＞22」をより正確に定義する

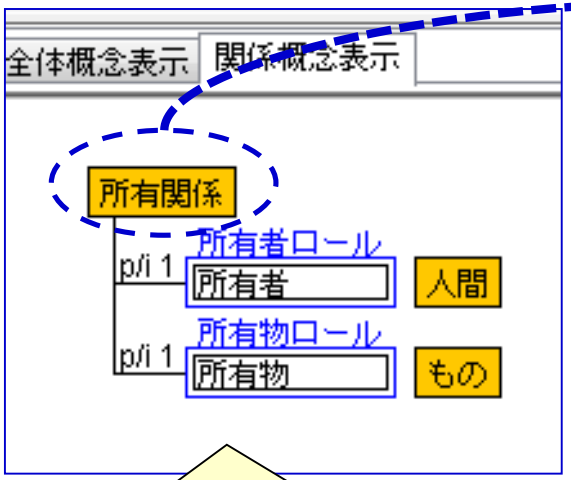


サンプル：
学校＋ロール.xml

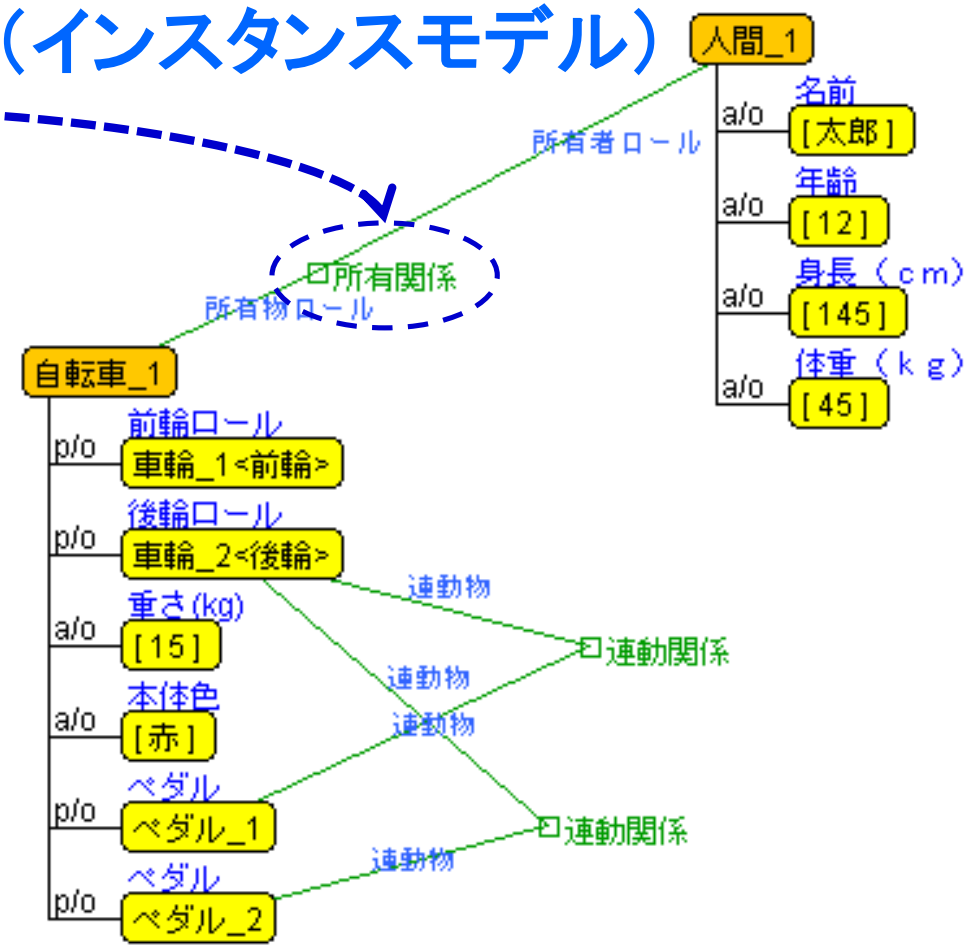
補足説明:
その他の関係の種類

- **特定のインスタンス間で「成り立ちうる」関係**
  - すべてのインスタンス間で成り立つとは限らない関係
    - 例)「人間」と「もの」の間の「所有関係」
    - 「関係概念」として定義，「インスタンス」ではリンクで記述
- **すべてのインスタンス間で成り立つ関係**
  - 特定の状況(コンテキスト)の下で成り立つ関係
    - 例)自転車の部品間の「連動関係」や「接続関係」
    - 「スロット間のリンク」として記述
  - コンテキストに関係なく成り立つ関係
    - 例)「3つ星」と「2つ星」より「ランクの上下関係」
    - 「ノード間のリンク」(法造では未サポート)に対応
    - 適切なコンテキストを概念として導入することで表現可能

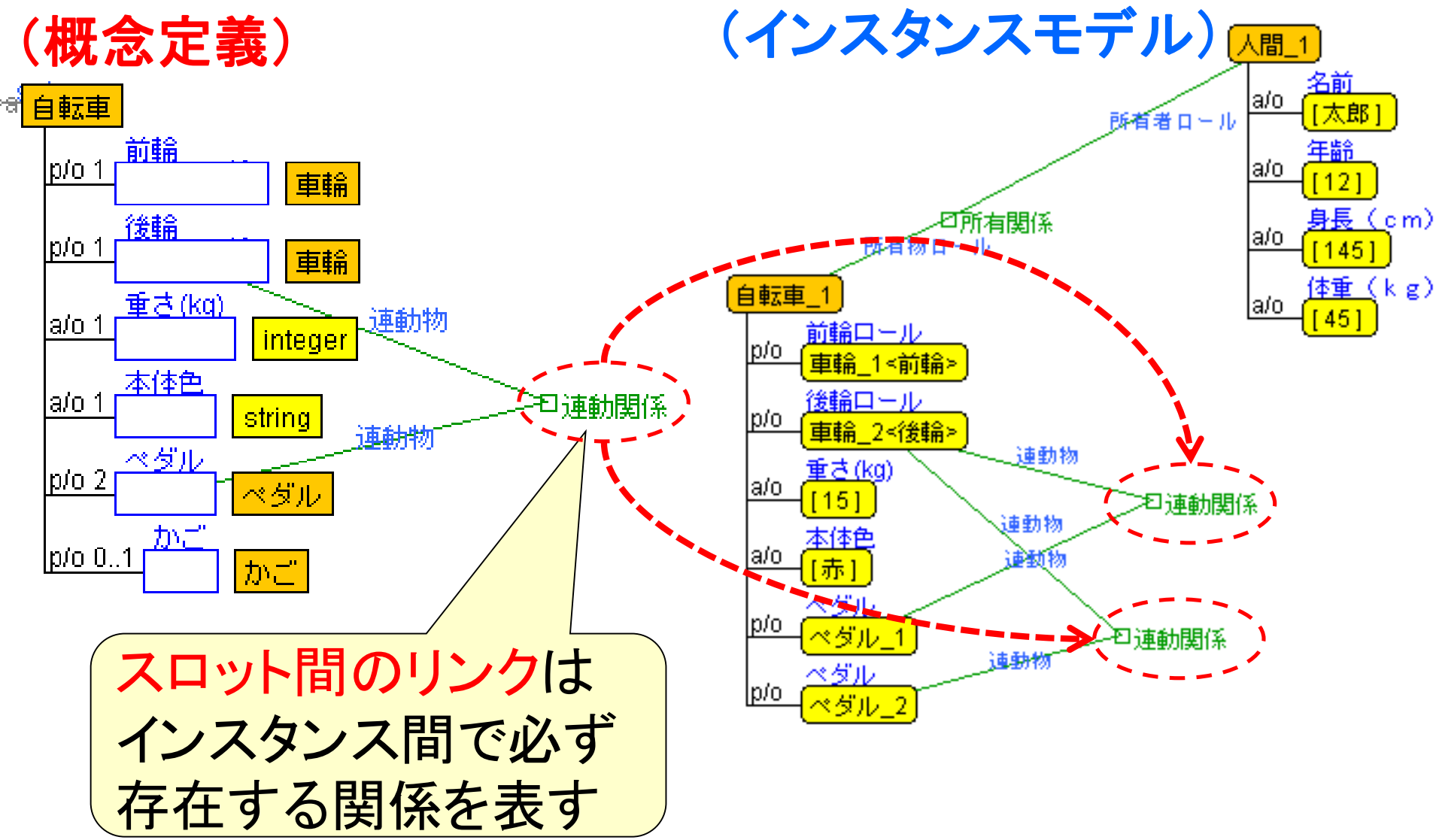
関係概念の定義



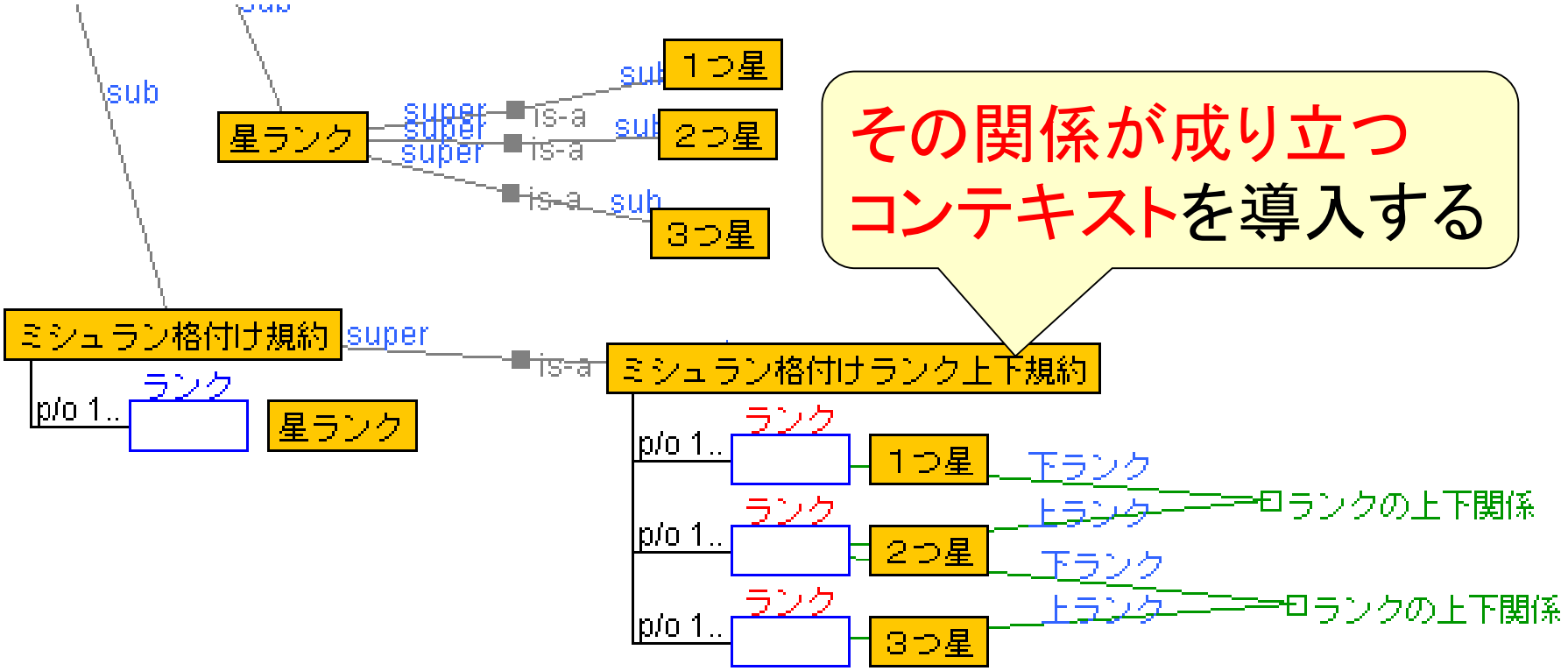
関係概念の定義はインスタンス間で成り立ちうる関係を表す



特定のコンテキストで必ず成り立つ関係（スロット間の関係）



概念間に常に成り立つ関係の表現



コンテキストの導入無しで常に成り立つ関係の別な記述法を導入するかについては検討中

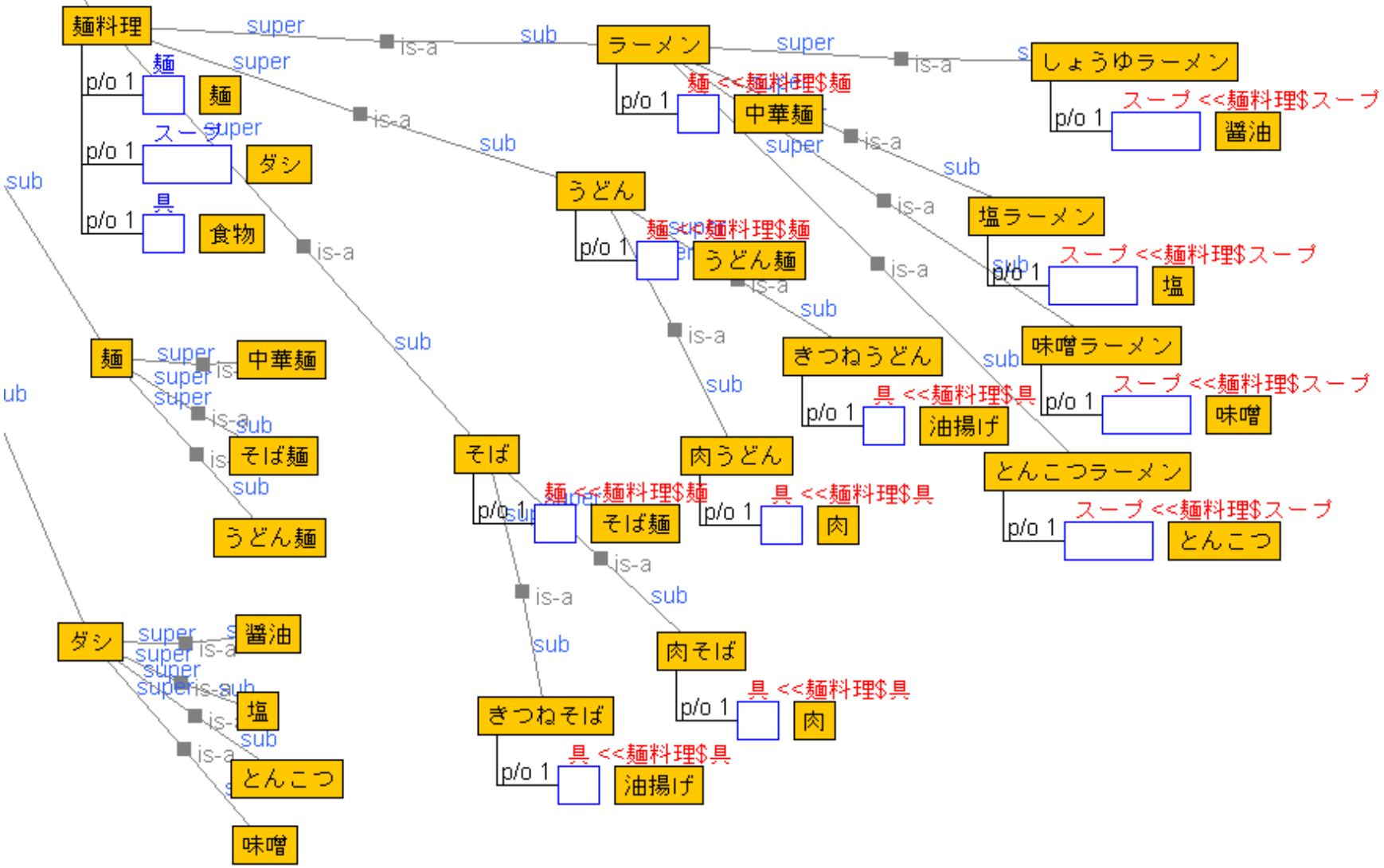
オントロジー構築・概念化の事例
～①一般的な概念化の事例～

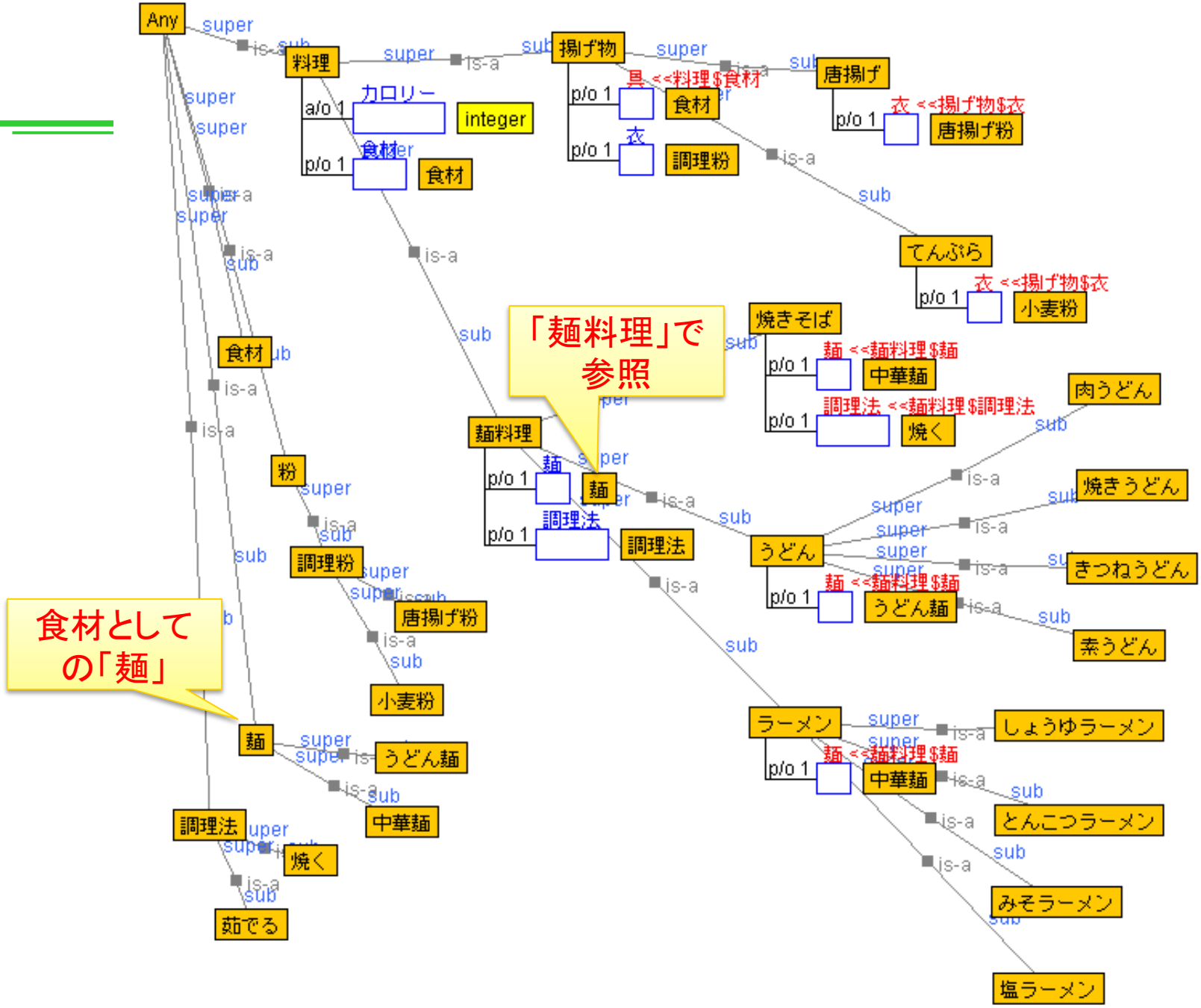
- ロール概念を利用した概念化
  - 麺料理
  - 外国人
- 動的な状況の概念化
  - ジャンケン

オントロジー構築・概念化の例 ～「麺料理」オントロジーの構築～

- **課題内容**
  - 今日の講義で解説した操作方法を使って，麺料理のオントロジーを構築せよ．
  - 麺料理の分類・検索ができる程度の概念を定義する
    - 単なる「麺の種類（うどん，そば，ラーメン，…）」だけでなく，「きつねうどん」，「豚骨ラーメン」，…など，「麺を使った料理」を定義すること．
  - 以下の点に心がけること
    - 適切なis－a階層を構築する
    - 概念の違いをスロットを用いて適切に定義する
    - スロットの継承・特殊化を適切に使用する

演習結果の例


麺料理
super
is-a
sub
ラーメン
しょうゆラーメン
麺
p/o 1
麺
スープ
ダシ
具
食物
うどん
うどん麺
塩ラーメン
塩
きつねうどん
油揚げ
味噌ラーメン
味噌
とんこつラーメン
とんこつ
中華麺
そば麺
そば
肉うどん
肉
肉そば
きつねそば
醤油
塩
麺 <<麺料理$麺
具 <<麺料理$具
スープ <<麺料理$スープ


Any
料理
揚げ物
唐揚げ
カロリー
integer
食材
唐揚げ粉
調理粉
てんぷら
小麦粉
焼きそば
中華麺
焼く
肉うどん
焼きうどん
きつねうどん
素うどん
麺料理
麺
調理法
うどん
うどん麺
ラーメン
しょうゆラーメン
とんこつラーメン
みそラーメン
塩ラーメン
粉
調理粉
唐揚げ粉
小麦粉
麺
うどん麺
中華麺
調理法
焼く
茹でる
「麺料理」で参照
食材としての「麺」

演習のポイント

サンプル：
麺料理（仮）.xml

- 「麺料理」の概念定義
  - 下位概念の分類に必要なスロットを適切に定義できているか？
  - スロットのクラス制約で参照する概念の階層が，適切に定義できているか？
  - 下位概念の分類観点が統一されているか？
- 「麺料理」と「麺」の区別が出来ているか？
  - 「料理としての麺」と「食材としての麺」は異なる．
  - 「料理としての麺」は“「食材としての麺」を材料（部分）として持つ”
  - 自然言語で「ラーメン」や「うどん」と言ったときは，両者を区別していない．
  - 「とんこつラーメン」や「きつねうどん」は，「料理としての麺」を指している．
    - それらの上位概念としての「ラーメン」や「うどん」と，
    - それらの部分（材料）としての「ラーメン」や「うどん」

    を考えると，それぞれ別の概念であることが分かる．

    ロールと呼ぶ
  - このような，自然言語では混同されている概念を適切に区別することが，オントロジー構築で重要な考え方．

麺料理の部分（材料）としての麺

- **部分（材料）としての「ラーメン」や「うどん」のクラス制約**
  - 小麦粉，そば粉などの「粉」？
  - （食材としての）「中華麺」，「そば」，などの「麺」を別の概念として定義する？
- **考え方のポイント**
  - （食材としての）「中華麺」，「そば」，などの「麺」は，（料理としての）「ラーメン」や「そば」の部分にならなくても，独立して存在することが出来る．＝コンテキストに独立して存在可能
    - Cf．「揚げ物」の「衣」は，「揚げ物」の部分としてしか存在できない．

    （揚げ物をバラバラにして，衣を残した場合であっても，過去に一度，「揚げ物の部分」と存在することが必須）
  - 「中華麺」は，「ラーメン」以外の料理（例えば「焼きそば」）の材料（部分）となることがある．＝複数概念から参照される

→これらの条件を満たす場合は，別の概念として定義した方が良い．

オントロジー構築・概念化の例
「外国人」の概念化

- ロール概念を使って「外国人」を定義することを考える.
  - 「外国人を定義するコンテキスト」は数通り考えることが出来る.
  - 考えられるコンテキストを挙げ，それぞれ簡単に説明せよ.

解説～外国人～

サンプル：
外国人.xml

- **演習のポイント**
  - 何を外国人（ロール）を定義するコンテキストとするかが重要.
    - 国をコンテキストとして外国人を定義した場合，各国のインスタンス毎に，外国人を定義することになるので非現実的.
- **概念化例**
  - ある人に注目した際の外国人（Foreigner-H）
  - ある国に注目した際の外国人（Foreigner-C）
  - ある人の母国に注目した際の外国人（Foreigner-HC）
  - 2人の人間の関係から見た際の外国人（Foreigner-R）

解説～外国人～

- **演習のポイント**
  - それぞれのコンテキストにおいて，スロット間の関係用いて制約を記述している．
  - 各コンテキストの「to whom」「to country」といったスロットは，「誰（何）にとっての外国人か」を示す．
  - 各外国人ロールにも同じスロットがあるのは，外国人ロールホルダーを別の箇所でクラス制約として利用する際に，「to whom」や「to country」スロットを山椒失せきるようにするため．
    - 利用例は「Foreigner club in Japan」（日本における外国人クラブ）を参照

オントロジー構築・概念化の例
～ジャンケン～

- **課題内容**
  - 「ジャンケン」のオントロジーを作れ.
  - グー，チョキ，パーの強弱を表す.
  - ２人で行うジャンケンに限定してよい.

解説：ジャンケン

サンプル：
(4) じゃんけん.xml

- **演習のポイント**
  - 「ジャンケン」は手だけではなく，「足」や口で“ぐー”と言うなどの「発話」に依っても行えるので，グー，チョキ，パーの「ジャンケン手」をロール概念として定義する．
  - クラス制約として，手，口，発話など「どのような形」を出すかということを表す「人体の部分_機能」を定義し，それが「ジャンケンの手規約」というコンテキストの元で「ジャンケンの手」ロールを担うことを定義．
  - さらに「ジャンケンの強弱規約」で3種類のジャンケン手があることと，その強弱を表している．

解説：ジャンケン

- **演習のポイント**
  - ジャンケンに「勝敗がつく」場合と「あいこ」の場合があることを「ジャンケンシーン」として定義.
  - 実際に行われる「ジャンケンゲーム」は「ジャンケンシーン」が決着のつくまで繰り返し行われるということを再帰的に定義している.

※この演習では2人で行うジャンケンに限定している.

（3人以上となるとかなり複雑な定義が必要）