UNIVERSITÀ
DI TORINO

di.unito.it

DIPARTIMENTO
DI INFORMATICA

Editing di ontologie tramite il linguaggio di programmazione funzionale ℂDuce

Candidato:
Davide Camino

Relatrice:
Viviana Bono

Introduzione

Definizioni e presentazione degli strumenti usati

Introduzione

- Ontologie
- Editing
- ℂDuce

Introduzione

- Ontologie
- Editing
- ℂDuce

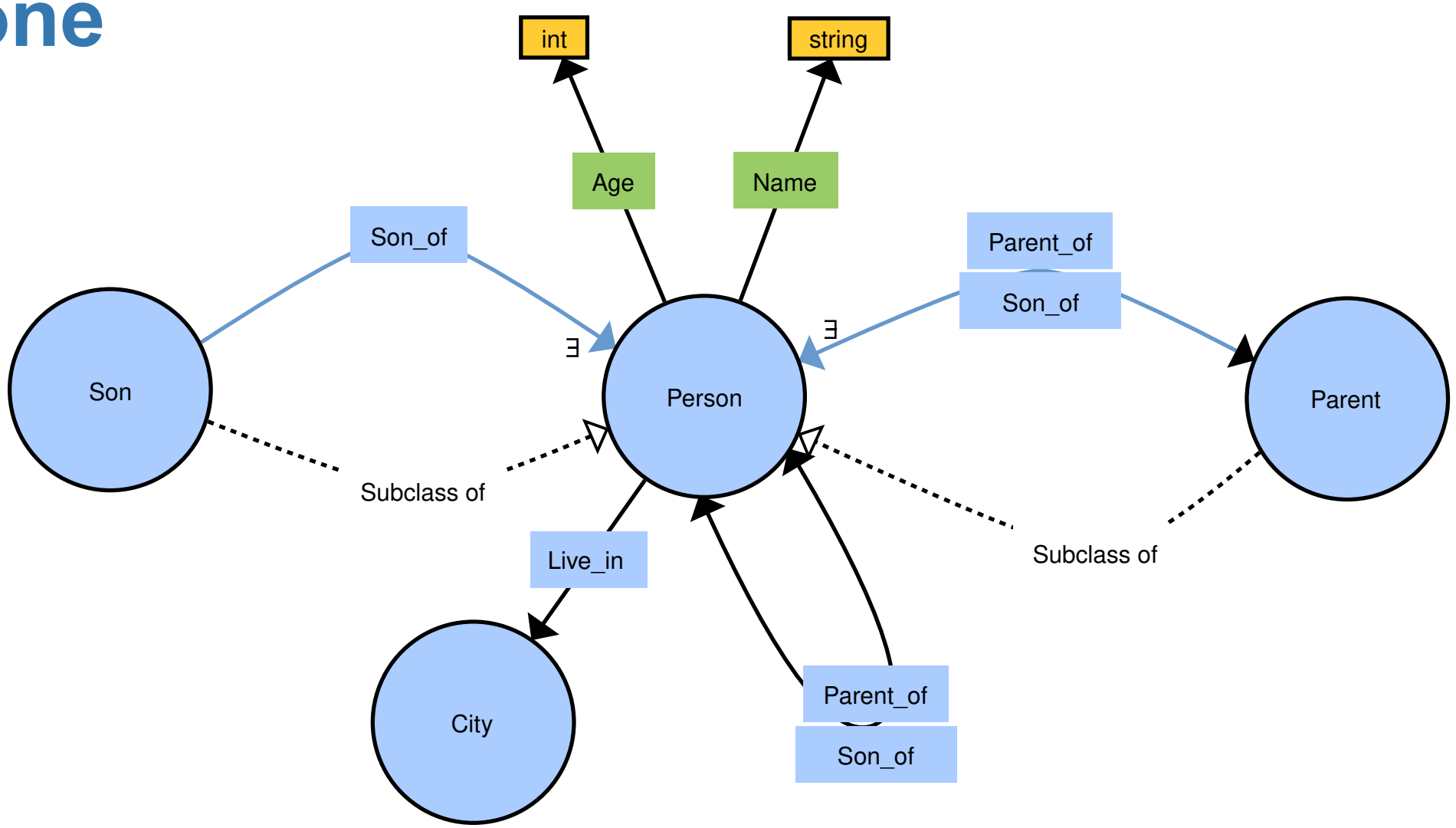
Introduzione

- Ontologie
- Editing
- ℂDuce

- Basi di conoscenza (KB)
- Classi e individui
- Gerarchia di classi ed ereditarietà multipla
- Relazioni tra classi
- Salvate come documenti XML con tag OWL

Introduzione

- Ontologie
- Editing
- ℂDuce

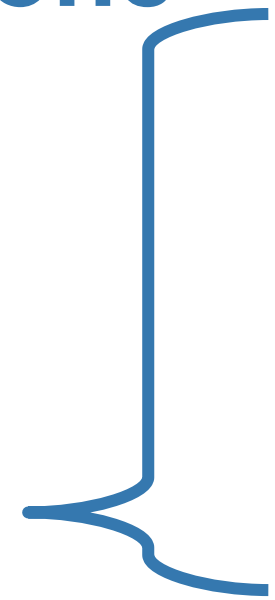


Introduzione

- Ontologie
- Editing
- ℂDuce

Introduzione

- Ontologie
- Editing
- ℂDuce



- Refactor
- Traduzione partendo da altre basi di conoscenza
- Merge

Introduzione

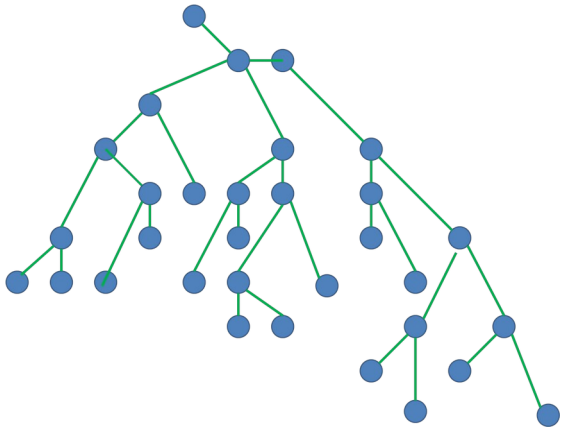
- Ontologie
- Editing
- ℂDuce

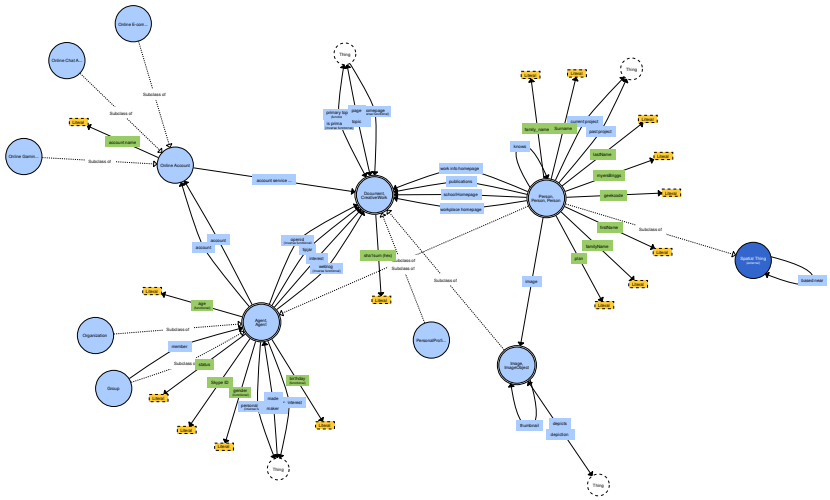
- Refactor
- Traduzione partendo da altre basi di conoscenza
- Merge

Introduzione

- Ontologie
- Editing
- ℂDuce

- Refactor
- Traduzione partendo da altre basi di conoscenza
- Merge

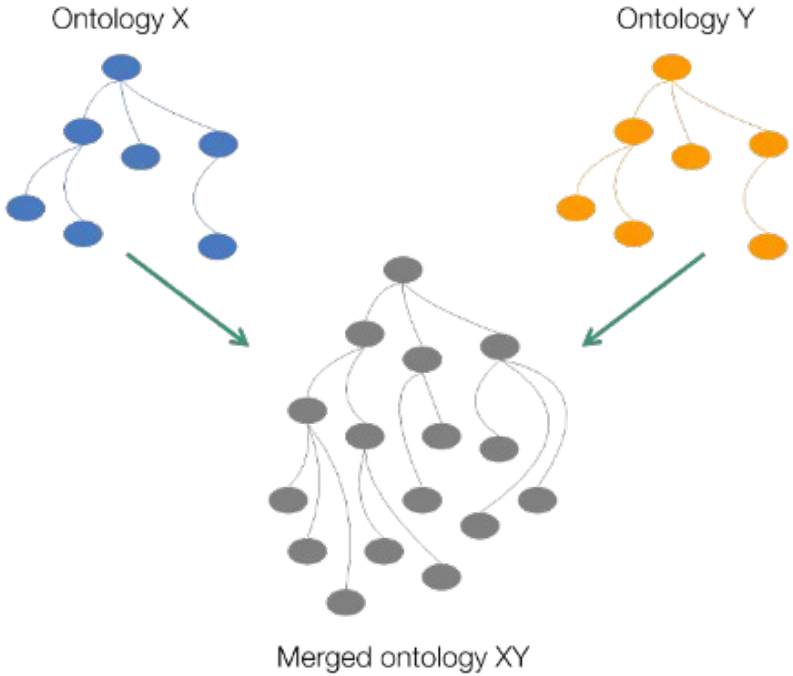




Introduzione

- Ontologie
- Editing
- ℂDuce

- Refactor
- Traduzione partendo da altre basi di conoscenza
- Merge



Introduzione

- Ontologie
- Editing
- ℂDuce

Introduzione

- Ontologie
- Editing
- ℂDuce

- Sviluppato specificatamente per manipolare documenti XML
- Utilizzabile in modo interattivo o tramite script
- Controllo statico dei tipi

Introduzione

- Ontologie
- Editing
- ℂDuce

Funzioni

- Pattern Matching
- Overloading
- Funzioni di ordine superiore
- Applicazioni parziali

```
1  fun f (t1 → s1;  ... ; tn → sn)
2      | p1 → e1
3      ...
4      | pm → em
```

Introduzione

- Ontologie
- Editing
- ℂDuce

Manipolazione di liste

- map

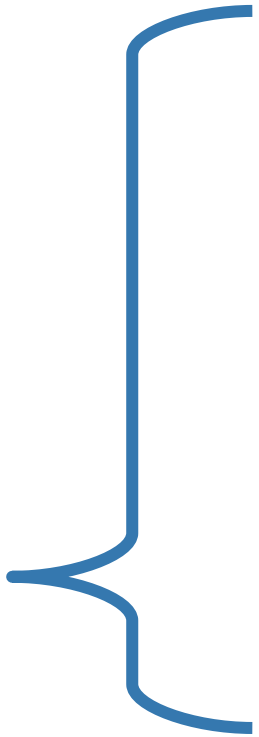
```
1  map e with
2      | p1 → e1
3      ...
4      | pn → en
```

- transform

```
1  transofrm e with
2      | p1 → e1
3      ...
4      | pn → en
```

Introduzione

- Ontologie
- Editing
- ℂDuce



Query e proiezioni

```
1  select e from
2     p1 in e1,
3     p2 in e2,
4          :
5     pn in en
6  where c
```

Introduzione: perchè innovativo?

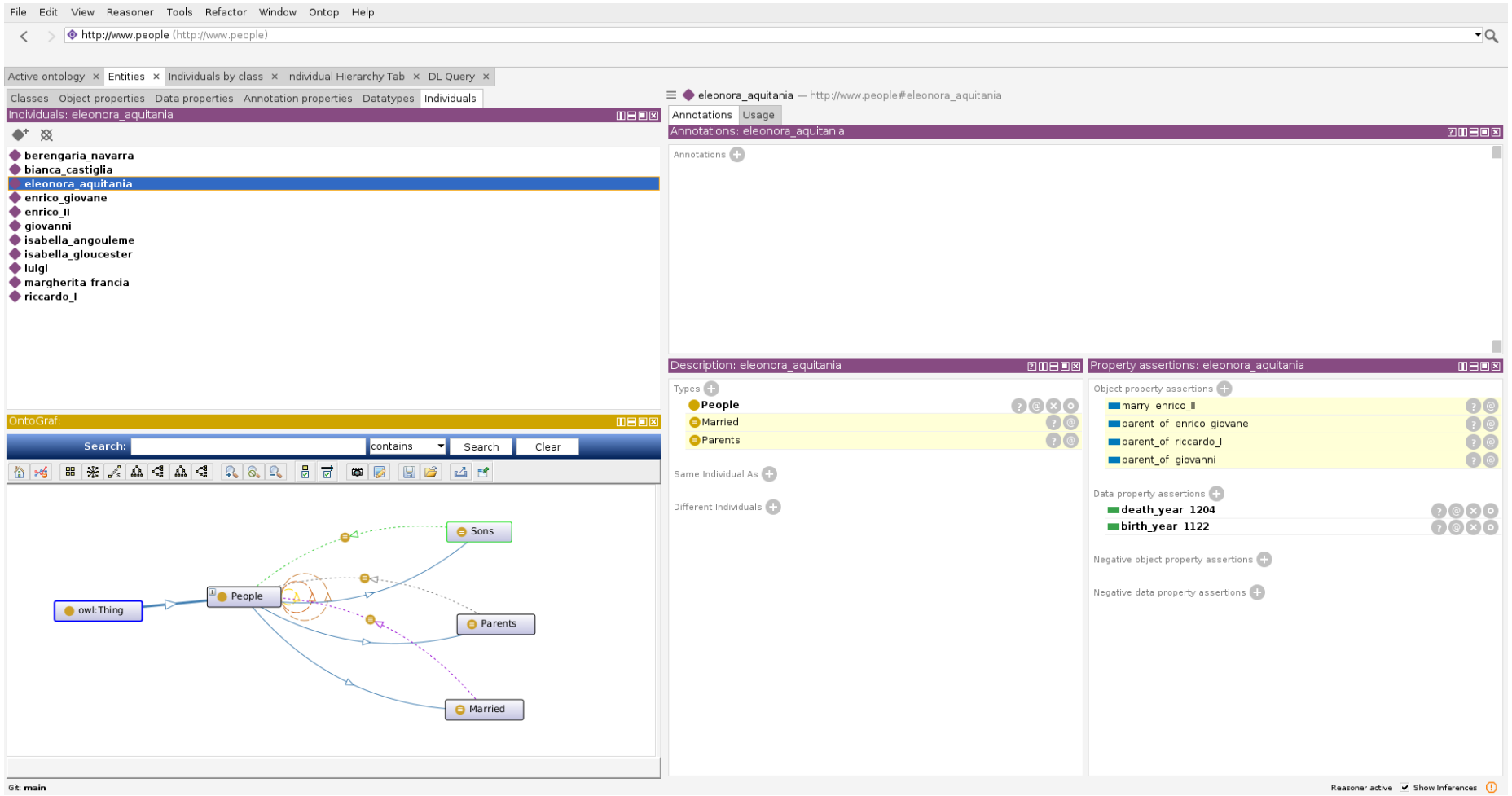
- Protégé
- Neon Toolkit
- Fluent Editor
- Semantic Turkey

Introduzione: perchè innovativo?

- Protégé
- Neon Toolkit
- Fluent Editor
- Semantic Turkey

Introduzione: perchè innovativo?

- Protégé



Traduzione da un’altra base di conoscenza

Trasformare un tesauro in un ontologia in modo da aumentarne l’espressività

Cos’è un tesauro?

- Dizionario tassonomico
- Forniscono sinonimi, contrari e traduzioni
- Relazione “`broader term`”
- Tag “SKOS”

Cos’è un tesauro?

Europeana Fashion Thesaurus

europeana
fashion

europeana

Cos’è un tesauro?

fashion objects

- ++ fashion objects
    - -- costume accessories
        - -- costume accessories worn
            - amulet
            - -- accessories worn on arms or hands
                - armband
                - -- handwear
                    - mittens (handwear)
                    - mitts (fingerless handwear)
                    - glove
                    - muff
                    - sports glove
                    - gauntlet
            - -- accessories worn on the legs or feet
                - chaps
                - -- footwear

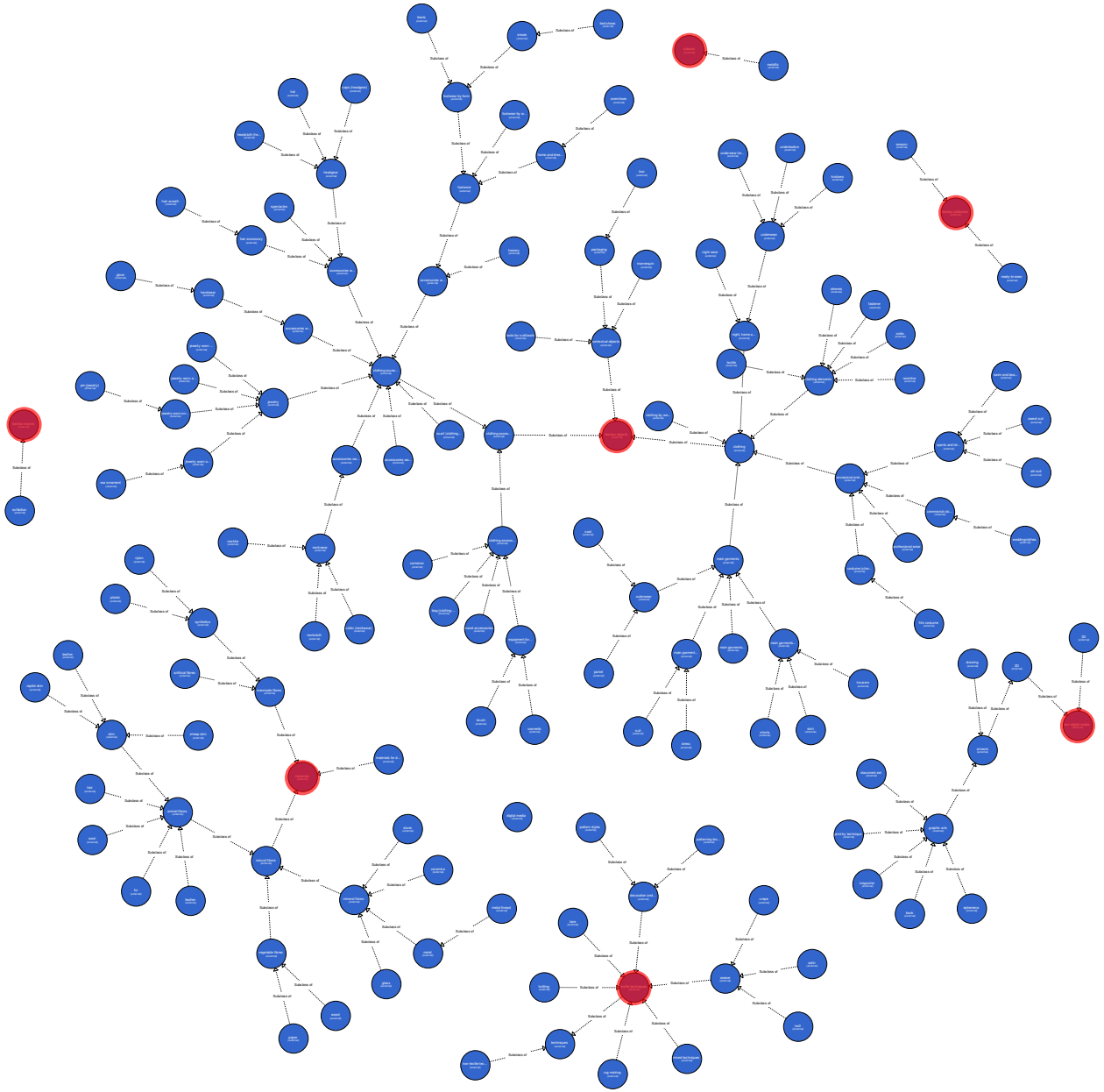
Struttura dell'ontologia

```
type Ontology = <rdf:RDF xml:base=String> [ Thing * ]
type Thing = Ont | Class | Individual

type Ont = <owl:Ontology rdf:about=String> []

type Class = <owl:Class rdf:about=String> [ ClassAtt * ]
type ClassAtt = SubClass | Label | Note | Dictionary
type SubClass = <rdfs:subClassOf rdf:resource=String> []
type Label = <rdfs:label xml:lang=String> String
type Note = <skos:scopeNote xml:lang=String> String
type Dictionary = <skos:exactMatch rdf:resource=String> []

type Individual = <owl:NamedIndividual rdf:about=String>
                    [ IndAtt * ]
type IndAtt = IndClass | IndProp
type IndClass = <rdf:type rdf:resource=String> []
type IndProp = <_ rdf:resource=String> [];;
```

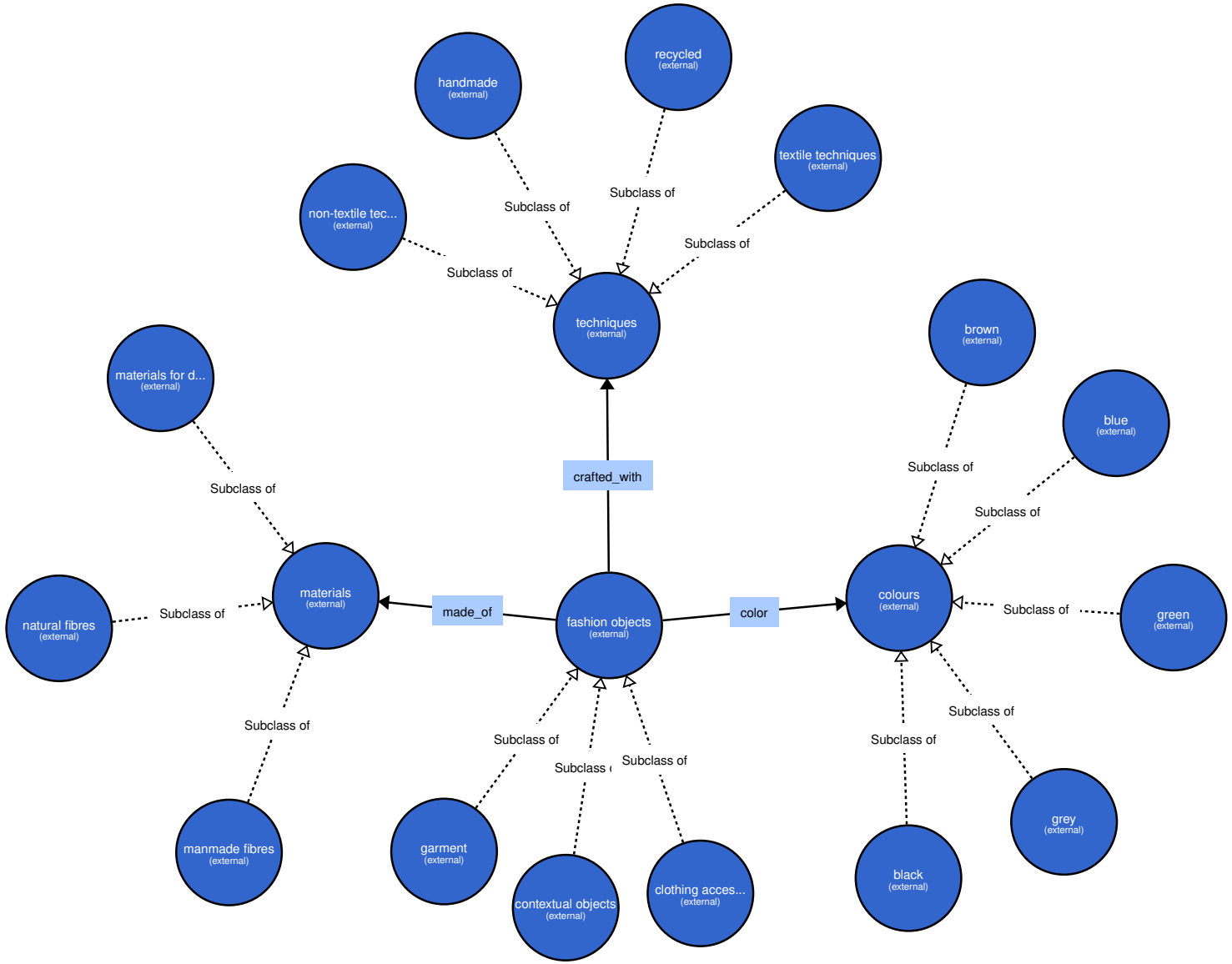
Traduzione

```
1  let fun thesaurusToOntology (Thesaurus → Ontology)
2    <rdf:RDF>[ (concepts :: Desc) * ] →
3      let newClasses = map concepts with
         <rdf:Description rdf:about=ab> [ (descAttr :: DescAtt)* ] →
4      let classAttr = transform descAttr with
5        | x & ScopeNote → [x]
6        | x & ExactMatch → [x]
7        | <skos:prefLabel xml:lang=l> lab
             → [<rdfs:label xml:lang=l> lab]
8        | <skos:altLabel xml:lang=l> lab
             → [<rdfs:label xml:lang=l> lab]
9        | <skos:broader rdf:resource=res> []
             → [<rdfs:subClassOf rdf:resource=res> []]
10           in
11           <owl:Class rdf:about=ab> classAttr
12     in
13     <rdf:RDF xml:base="http://www.semanticweb.org/OntEur">
         ( [ <owl:Ontology rdf:about="OntEur"> [ ] ] @ newClasses );;
```

Risultato



Risultato


handmade
(external)
recycled
(external)
textile techniques
(external)
non-textile tec...
(external)
Subclass of
Subclass of
Subclass of
Subclass of
techniques
(external)
materials for d...
(external)
crafted_with
brown
(external)
blue
(external)
Subclass of
Subclass of
Subclass of
natural fibres
(external)
Subclass of
materials
(external)
made_of
fashion objects
(external)
color
colours
(external)
Subclass of
green
(external)
Subclass of
Subclass of
Subclass of
Subclass of
Subclass of
manmade fibres
(external)
garment
(external)
contextual objects
(external)
clothing acces...
(external)
black
(external)
grey
(external)

Merge di Ontologie

Unione ontologie semplici per creare una descrizione di un dominio più complesso

Obbiettivo del merge

Rappresentazione della conoscenza

- Accorpare classi equivalenti
- Definire nuove relazioni
- Importare classi e individui già definiti
- Mostrare funzioni più sofisticate
- Derivare proprietà senza reasoner esterni

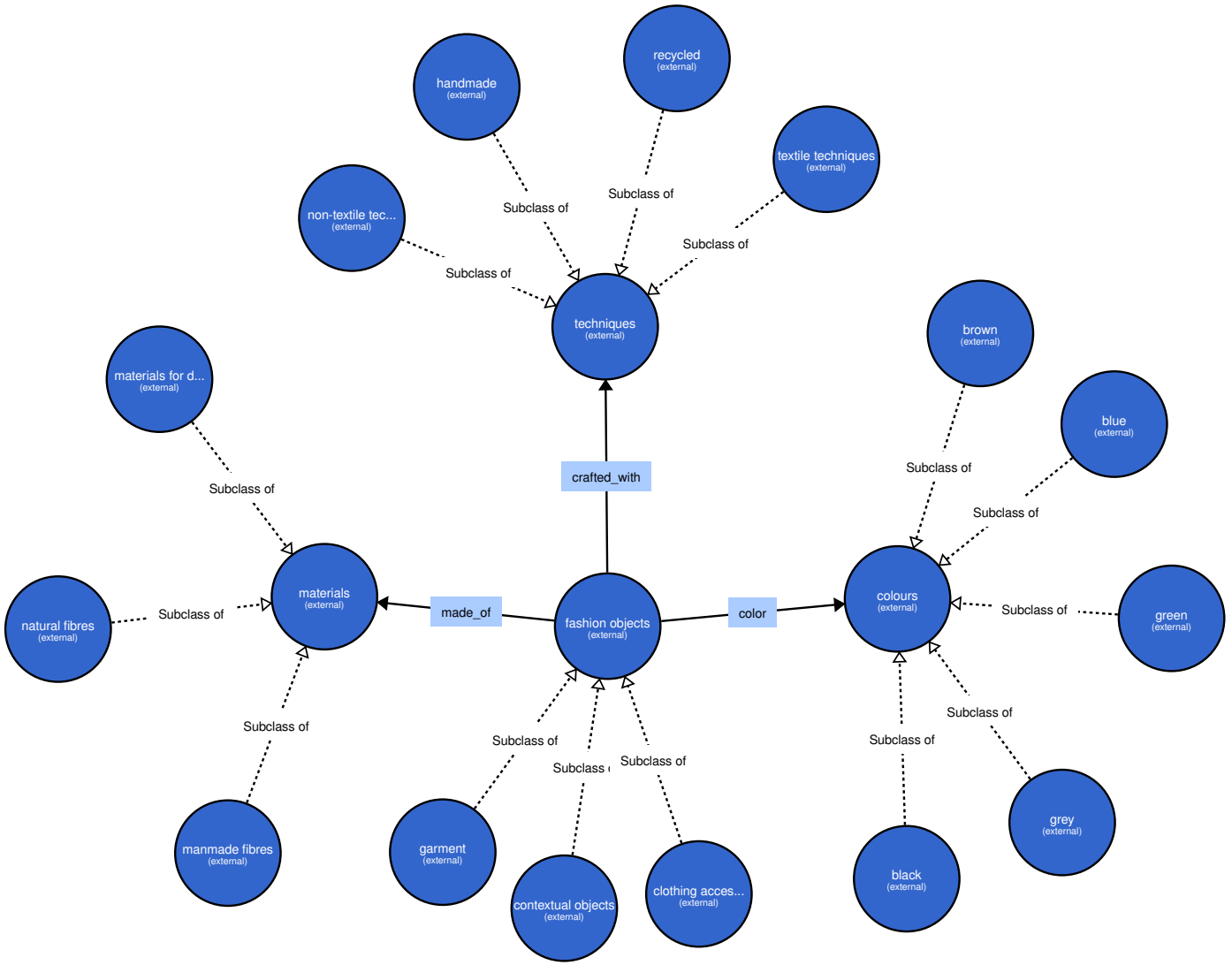
Obbiettivo del merge

- Accorpare classi equivalenti
- Definire nuove relazioni
- Importare classi e individui già definiti

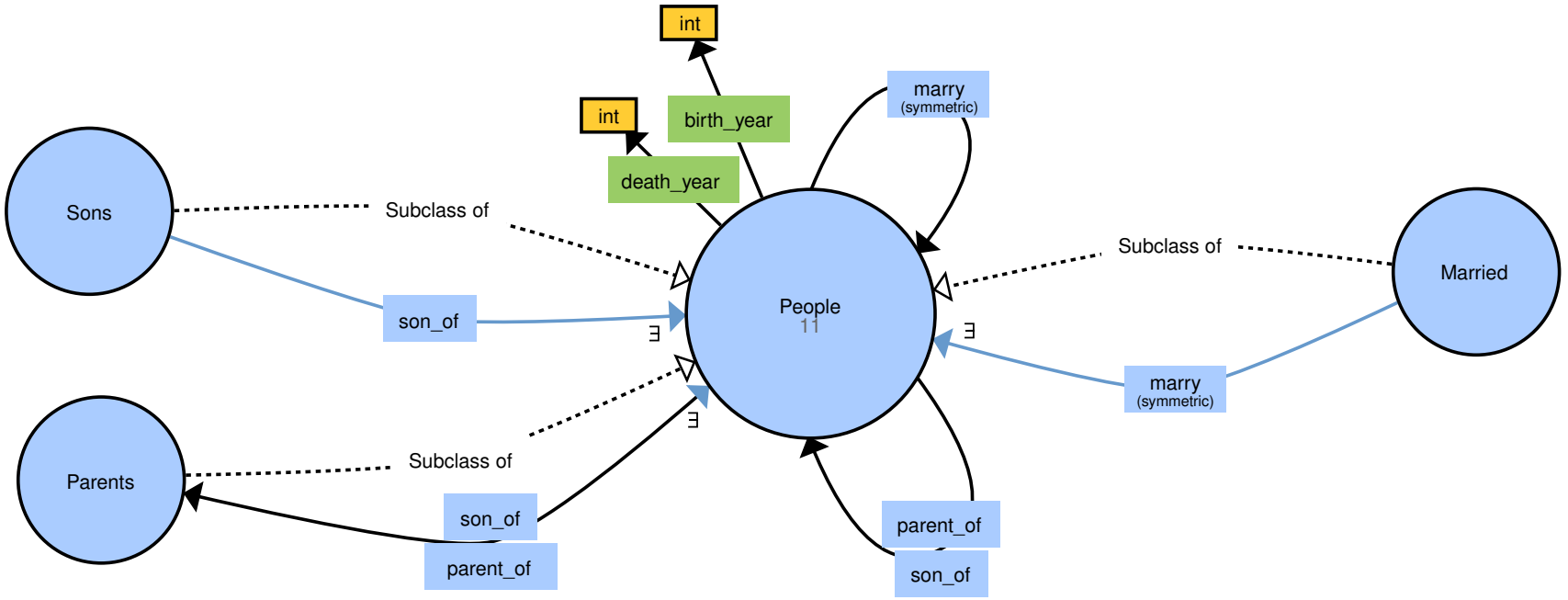
ℂDuce

- Sviluppare funzioni più sofisticate
- Derivare proprietà senza reasoner esterni

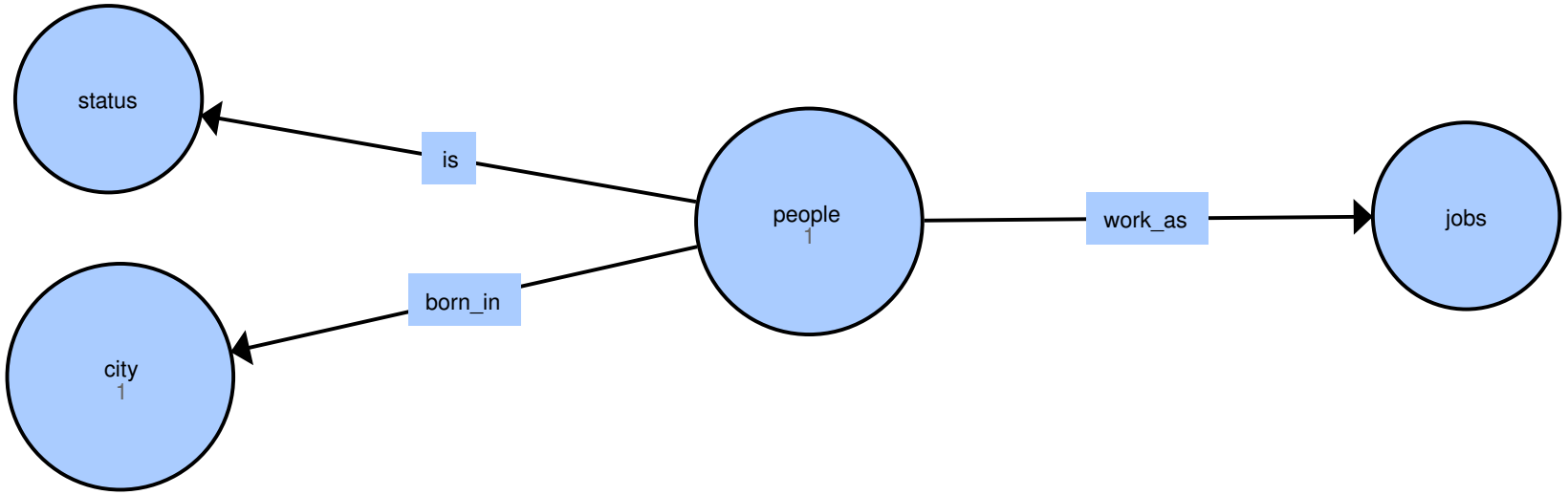
Ontologie di partenza



Ontologie di partenza



Ontologie di partenza



Obbiettivo del merge

- Importare tutte le persone vissute un un dato periodo storico
- Importare gli abiti e i materiali presenti in quel periodo
- Mantenere tutte le relazioni definite in precedenza
- Definirne di nuove per descrivere usi e costumi della società del periodo storico di interesse

Reasoning

```
let fun madeOf (Individual → Ontology → [Individual*])
  <owl:NamedIndividual ..> [ (ip::IndProp | _) *] → fun (Ontology → [Individual*])
    ont →
      transform ip with <ns1:made_of rdf:resource=str> [] →
        transform [ont]/Individual with x →
          match x with <owl:NamedIndividual rdf:about=a>[ _* ] → if a = str then [x] else [];;
```

```
let fun isArtificial (Individual → [Class*] → [Class*] → Ontology → Bool)
  ind → fun ([Class*] → [Class*] → Ontology → Bool)
    materials → fun ([Class*] → Ontology → Bool)
      artificialMats → fun (Ontology → Bool)
        ont → if (isInClasses ind materials ont)
          then (isInClasses ind artificialMats ont)
            else
              let matMade = madeOf ind ont in
                orList (map matMade with x → isInClasses x artificialMats ont);;
```

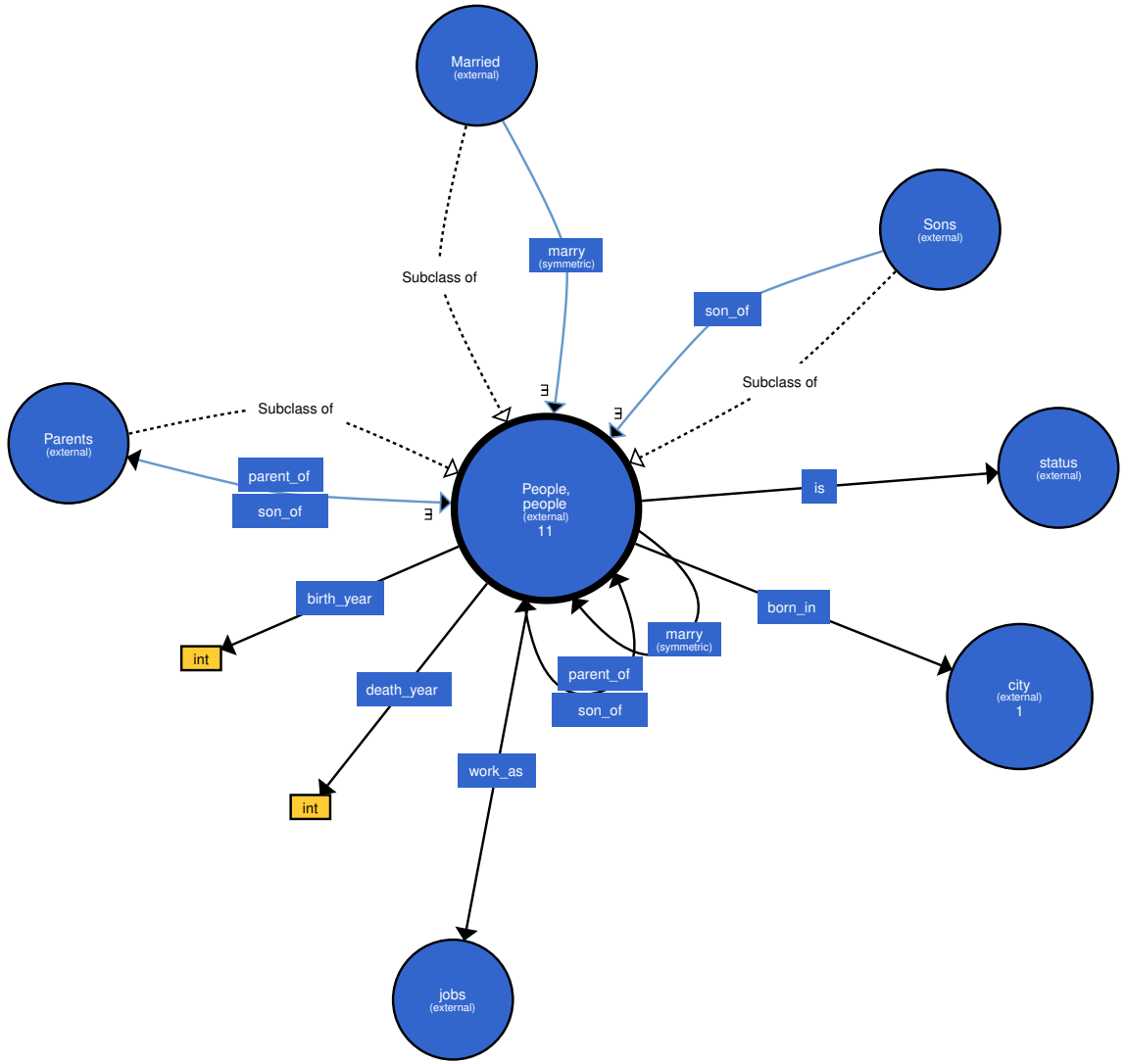
Ristrutturazione e assemblaggio

```
let newMaterials : [Class*] =
                   select x
                   from x in materials
                   where (not contains x artificialMats);;

let newFashionIndividual : [Individual*] =
                   select x
                   from x in [fashion]/Individual
                   where (not isArtificial x materials artificialMats fashion);;

let socPeople :? Class =
  match socPeople with
    <owl:Class rdf:about=str> [ (attr::ClassAtt)* ] →
      <owl:Class rdf:about=str>   (attr  @ [ <owl:equivalentClass
        rdf:resource="http://www.people#People"> [] ]) ;;

let newPeople : [Thing*] = [people]/(Thing \ Ont);;
```

Ristrutturazione e assemblaggio

```
let newThing : [Thing*] =
  [ newOnt ] @ newSociety @ [ socPeople ] @ newFashionIndividual @  newMaterials
  @ newPeople;;

let newOntology :? Ontology =
  <rdf:RDF xml:base="http://www.semanticweb.org/society_merged">  newThing ;;

dump_to_file_utf8 "society_merged.rdf" (print_xml_utf8 newOntology);;
```

Risultato



- Classe People unuione delle due definizioni
- Possibilità di definire relazioni tra persone e vestiario
- Selezionati solo vestiti realizzati con materiali presenti nel XII secolo

Conclusioni

Vantaggi e svantaggi dell’uso di ℂduce e possibili sviluppi futuri

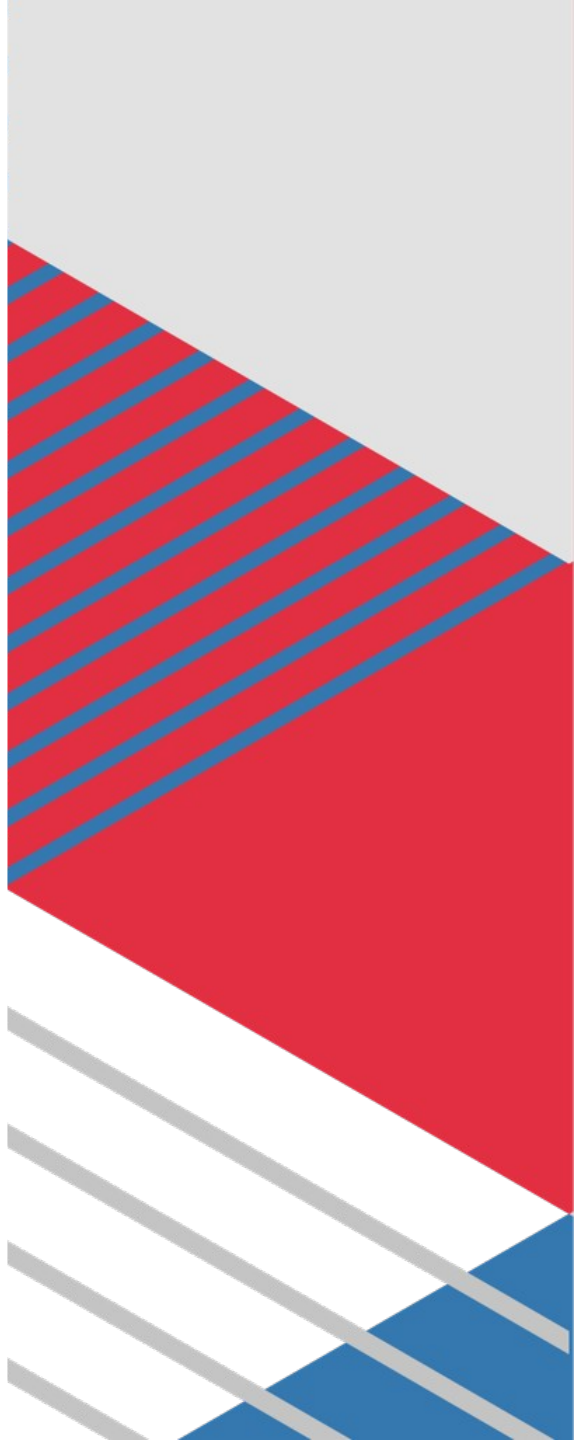
Svantaggi

- Progetto di intersse accademico
- Piccole dimensioni
- Aggiornamenti non costanti

Svantaggi

- Progetto di intersse accademico
- Piccole dimensioni
- Aggiornamenti non costanti

- Ottenere ℂDuce
- Compatibilità
- Formattazione dell’output
- Reperire informazioni



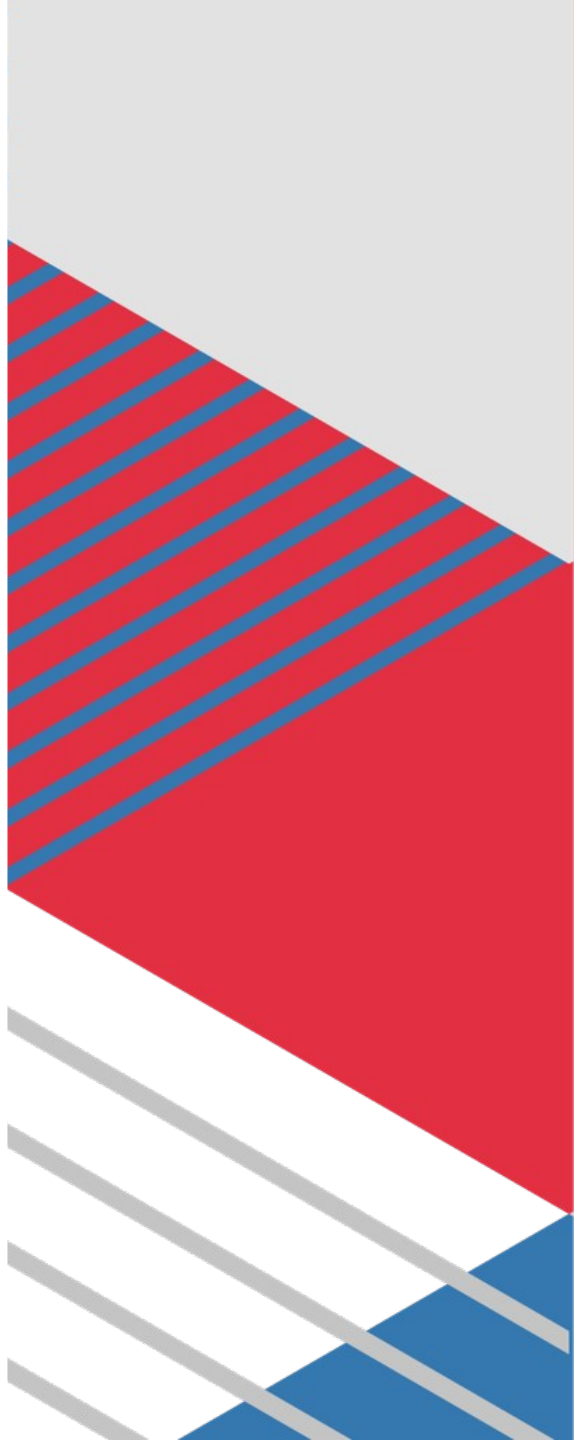
Svantaggi

- Progetto di intersse accademico
- Piccole dimensioni
- Aggiornamenti non costanti

- Ottenere ℂDuce
- Compatibilità
- Formattazione dell’output
- Reperire informazioni

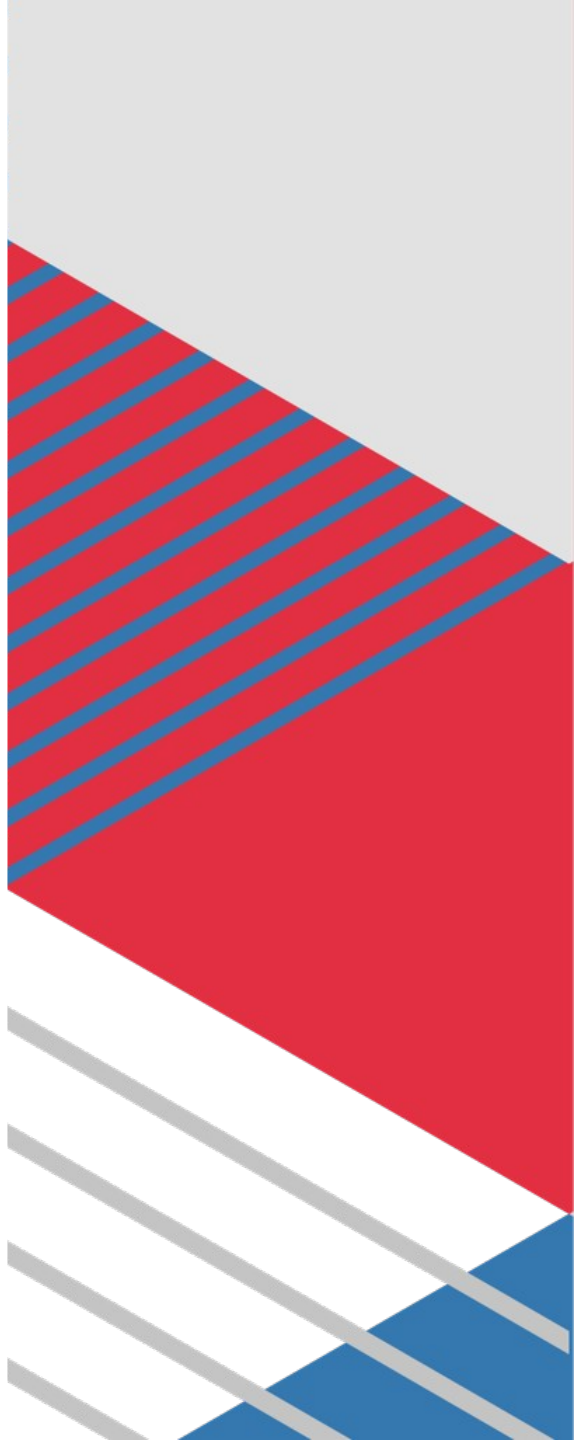
↓

- Difficoltà nella scrittura del codice



Vantaggi

- Codice sintetico e leggibile
- Sistema di tipi
- Funzioni di ordine superiore
- Trasformazioni potenti e corrette



Sviluppi futuri

- Confronto con esperti del settore
- Creazione di strumenti generali
- Sviluppo di interfaccia grafica

Grazie per l'attenzione

Domande?

UNIVERSITÀ
DI TORINO