
Karlsruhe Institute of Technology


N_I


N_C


N_R

$A \in N_C$

$R \in N_R$

$\lfloor C \sqcap D$

$C \sqcup D$


$\neg C$

$\lfloor \forall R.C$

$\lfloor \exists R.C$

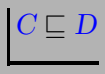
⌊□

$\underline{\lfloor \neg C}$

$\lfloor \forall R.C$

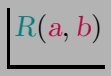
$\exists R.\ C$

$$C, D \to A \mid \top \mid \bot \mid C \sqcap D \mid C \sqcup D \mid \neg C \mid \forall R.C \mid \exists R.C$$


$C \sqsubseteq D$

$C \equiv D$

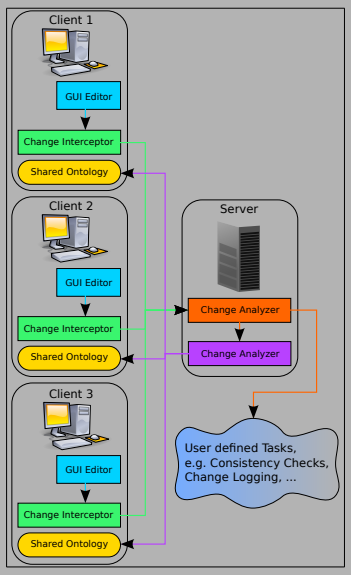

C(a)

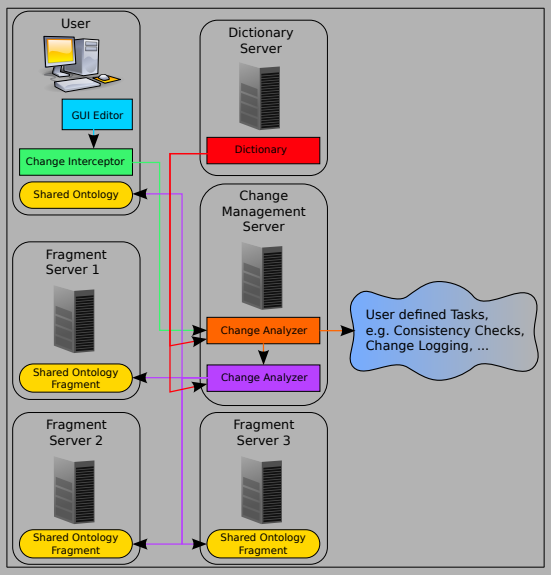

$R(a, b)$


$\Sigma(\mathcal{O})$

$Pizza, CheeseTopping \in N_C, hasTopping \in N_R$

$$(Pizza \sqcap \forall hasTopping.CheeseTopping) \sqsubseteq CheeseyPizza$$


Client 1
GUI Editor
Change Interceptor
Shared Ontology
Client 2
GUI Editor
Change Interceptor
Shared Ontology
Client 3
GUI Editor
Change Interceptor
Shared Ontology
Server
Change Analyzer
Change Analyzer
User defined Tasks,
e.g. Consistency Checks,
Change Logging, ...


User
GUI Editor
Change Interceptor
Shared Ontology
Dictionary Server
Dictionary
Change Management Server
Change Analyzer
Change Analyzer
User defined Tasks, e.g. Consistency Checks, Change Logging, ...
Fragment Server 1
Shared Ontology Fragment
Fragment Server 2
Shared Ontology Fragment
Fragment Server 3
Shared Ontology Fragment

$\underline{\lfloor o}$

φ

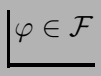
$\varphi \subseteq o$

$\underline{\lfloor \mathcal{I}}$

$\underline{\lfloor \mathcal{F}}$


ϕ

$\underline{\mathcal{I} \to \mathcal{F}}$


$\varphi \in \mathcal{F}$

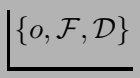
$$\boxed{\exists \iota \in \mathcal{I} : \phi(\iota) = \varphi}$$

$$\Sigma = \{\sigma(\varphi) \mid \varphi \in \mathcal{F}\}$$

$\underline{\lfloor \mathcal{F}}$

$$\mathcal{D} = \{\mathcal{I}_\varphi, \Sigma, \phi\}$$

$\underline{\mathcal{D}}$


$\{o, \mathcal{F}, \mathcal{D}\}$

⌊△

$\underline{\omega}$

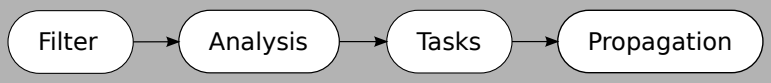
$$\nu = \{\Delta, \upsilon, \tau, \omega\}$$

$Change \to Boolean$

$n \in \mathbb{N}$


γ_i

$$\Theta = \bigcup_{i=0..n} \gamma_i$$


Filter
Analysis
Tasks
Propagation

Algorithm 1 ApplyChange(ν)

Require: Shared Ontology o, Change Filter Θ
  if not Filter(ν, Θ) **then**
    $\alpha \leftarrow$ Analyse(ν)
    Trigger(α)
    Propagate(α, ν)
    return applied changes on o
  else
    // ignore change
  end if

Algorithm 2 FILTER(ν, Θ)

Require: Criteria $\chi_i \in \Theta \qquad i = 1, 2, ..., n \qquad i, n \in \mathbb{N}$

```
for i = 1 to n do
  if χ_i(ν) then
    return true
  end if
end for
return false
```

```
@ProxyMethod
@PropagatingMethod
public List<OWLOntologyChange> applyChange(OWLOntologyChange
    change) {
    final SharedObjectMsg msg = SharedObjectMsg.createMsg("
        applyChangeSilent", change);
    try {
      // Send change to everyone
      sendSharedObjectMsgTo(null, msg);
    } catch (IOException e) {
      // Error processing
      e.printStackTrace();
    }
    if(calledByOntologyManager()) {
      return ontology.applyChange(change);
    }
    return ontology.getOWLOntologyManager().applyChange(change
        );
  }
```

Listing 7.1: The *applyChanges* method of the OWLReplicaOntology implementation

```
    public void applyChangeSilent(OWLOntologyChange change) {
2   change.setOntology(ontology);
    ontology.getOWLOntologyManager().applyChange(change);
4 }
```

Listing 7.2: The *applyChangesSilent* method of the OWLReplicaOntology implementation

```
% latex2html id marker 1571
[caption={Example of OWL axioms},label=owlclass,language=XML]
  <owl:Class rdf:about="#CheeseTopping">
    <rdfs:label xml:lang="pt">CoberturaDeQueijo</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#PizzaTopping"/>
    </rdfs:subClassOf>
  </owl:Class>
```