Supporting Competency Question-driven Ontology Authoring

Yuan Ren, Artemis Parvizi, Chris Mellish, Jeff Z. Pan, Kees van Deemter

University of Aberdeen, UK

Robert Stevens

University of Manchester, UK

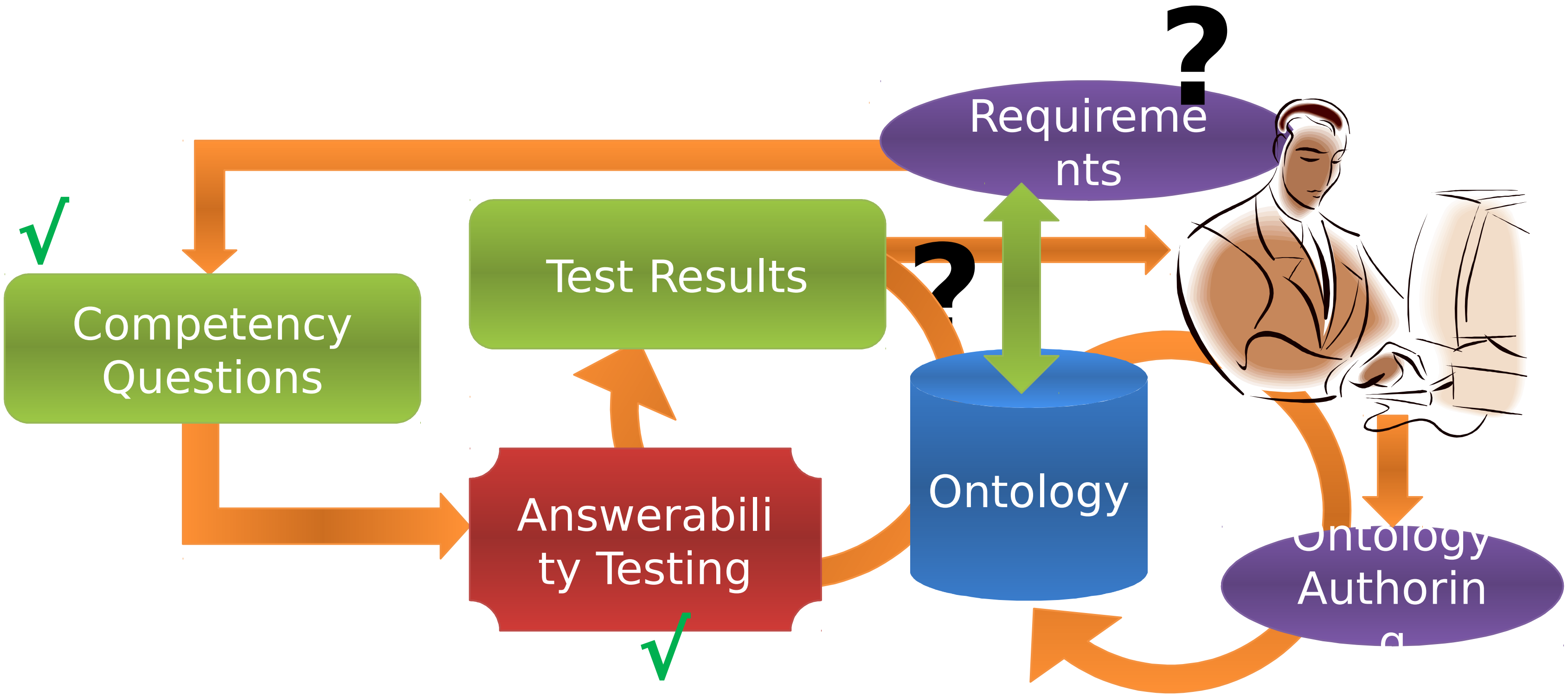
Ontology

- Provide schema-level knowledge to linked data
  - Specifying vocabularies
    - E.g. Pizza, Food, PizzaTopping
  - Specifying relations between terminologies
    - E.g. Pizza SUBCLASSOF Food, Pizza SUBCLASSOF hasTopping SOME PizzaTopping

- Modern ontology technologies are quite complex
  - Logic underpinning: Description Logics
  - Representation languages: RDF(S), OWL
  - Query language: SPARQL
  - Rule language: SWRL, RIF

Ontology Authoring

- Is difficult for authors who are unfamiliar with DLs, RDF, SPARQL, OWL, etc.
  - Difficult to specify and verify satisfaction of requirements
- Our vision: Competency Question-driven Ontology Authoring



CQs in Ontology Authoring

- A typical CQ: Which pizza has some cheese topping?

- Questions that people expect the constructed ontologies to answer
- Useful for novice users:
  - in natural languages
  - about domain knowledge
  - requires little understanding of ontology technologies

CQs in Ontology Authoring

- A typical CQ: Which pizza has some cheese topping?

  - Existing work focused on answering CQs directly
    - But is the answer meaningful?

- The ability to ***answer CQs meaningfully*** can be regarded as a functional requirement of the ontology

- Answer: empty set

- Possible scenarios
  - Pizza does not exist
  - Cheese topping does not exist
  - Pizzas are not allowed to have cheese topping
  - The ontology has not been populated with any cheesy pizza yet
  - …

CQs in Ontology Authoring

- A typical CQ: Which pizza has some cheese topping?

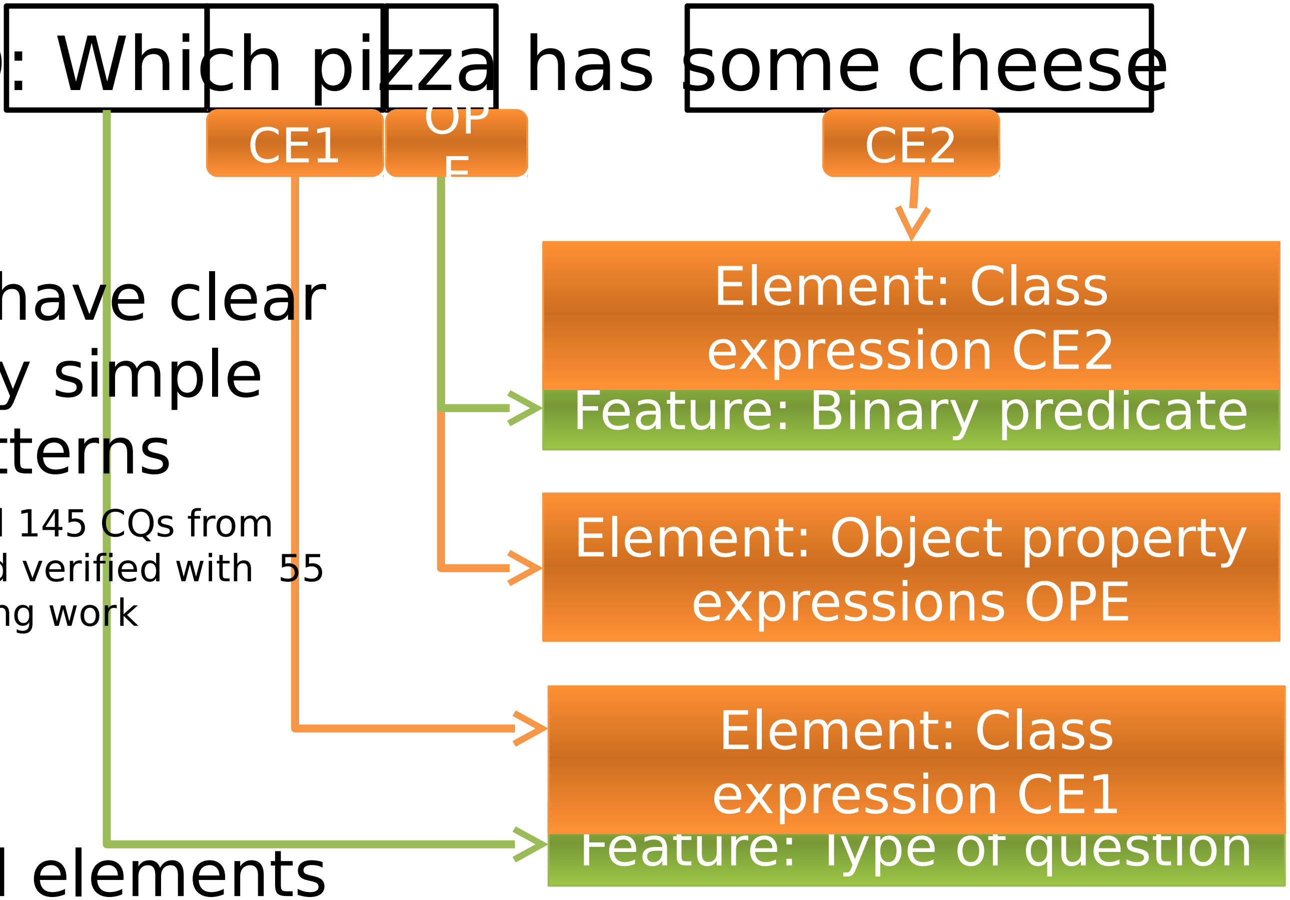
- A CQ comes with certain *presuppositions*
  - *Some conditions the speakers assume to be met*

- A CQ can be *meaningfully answered* only when its presuppositions are satisfied

- Classes *Pizza*, *CheeseTopping* should occur in the ontology
- Property *has(Topping)* should occur in the ontology
- The ontology should *allow Pizza to have CheeseTopping*
- …

CQs in Ontology Authoring

- A typical CQ: Which pizza has some cheese topping?



CE1

OP
E

CE2

Element: Class expression CE2

Feature: Binary predicate

Element: Object property expressions OPE

Element: Class expression CE1

Feature: Type of question

- CQs usually have clear and relatively simple syntactic patterns
  - We investigated 145 CQs from two corpora and verified with 55 CQs from existing work

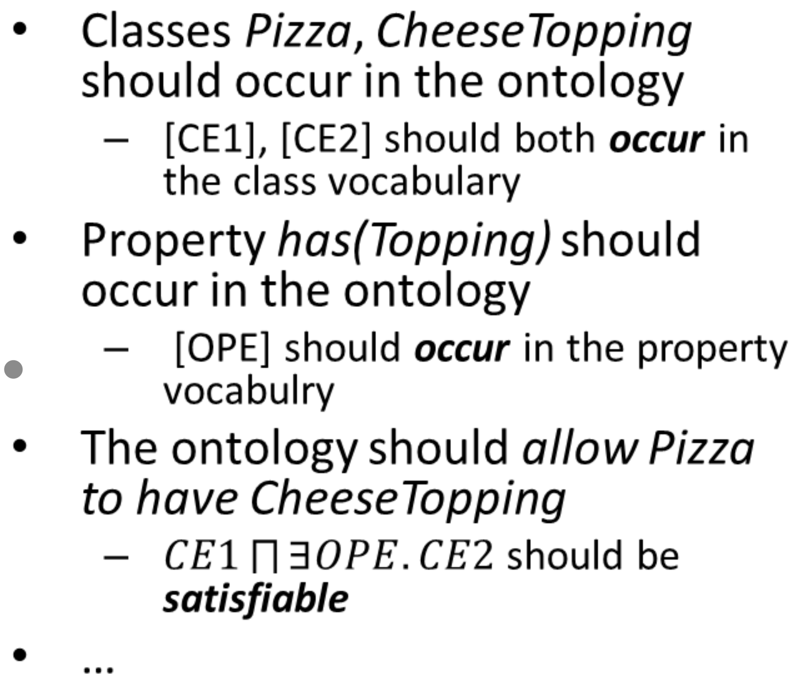
- Features and elements can be extracted

CQs in Ontology Authoring

- A typical CQ: Which pizza has some cheese topping?

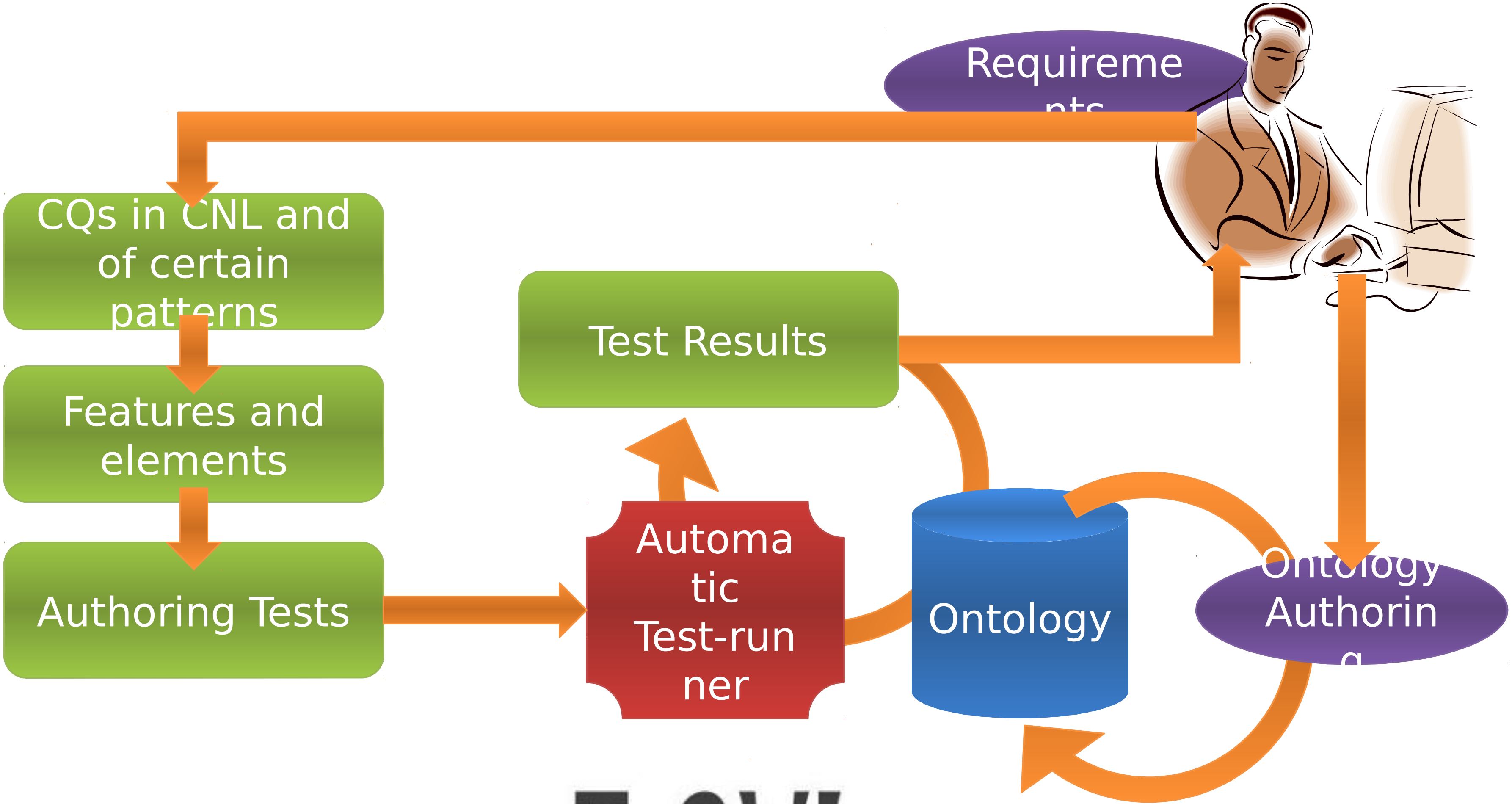
CE1 OPE CE2

- Satisfiability of CQ presuppositions can be verified by **authoring tests** generated based on its features and elements



- Classes *Pizza*, *CheeseTopping* should occur in the ontology
  - [CE1], [CE2] should both ***occur*** in the class vocabulary
- Property *has(Topping)* should occur in the ontology
  - [OPE] should ***occur*** in the property vocabulry
- The ontology should *allow Pizza to have CheeseTopping*
  - $CE1 \sqcap \exists OPE.CE2$ should be ***satisfiable***
- …

A Competency Question-driven Ontology Authoring Pipeline



Requireme
nts

CQs in CNL and
of certain
patterns

Features and
elements

Authoring Tests

Test Results

Automa
tic
Test-run
ner

Ontology

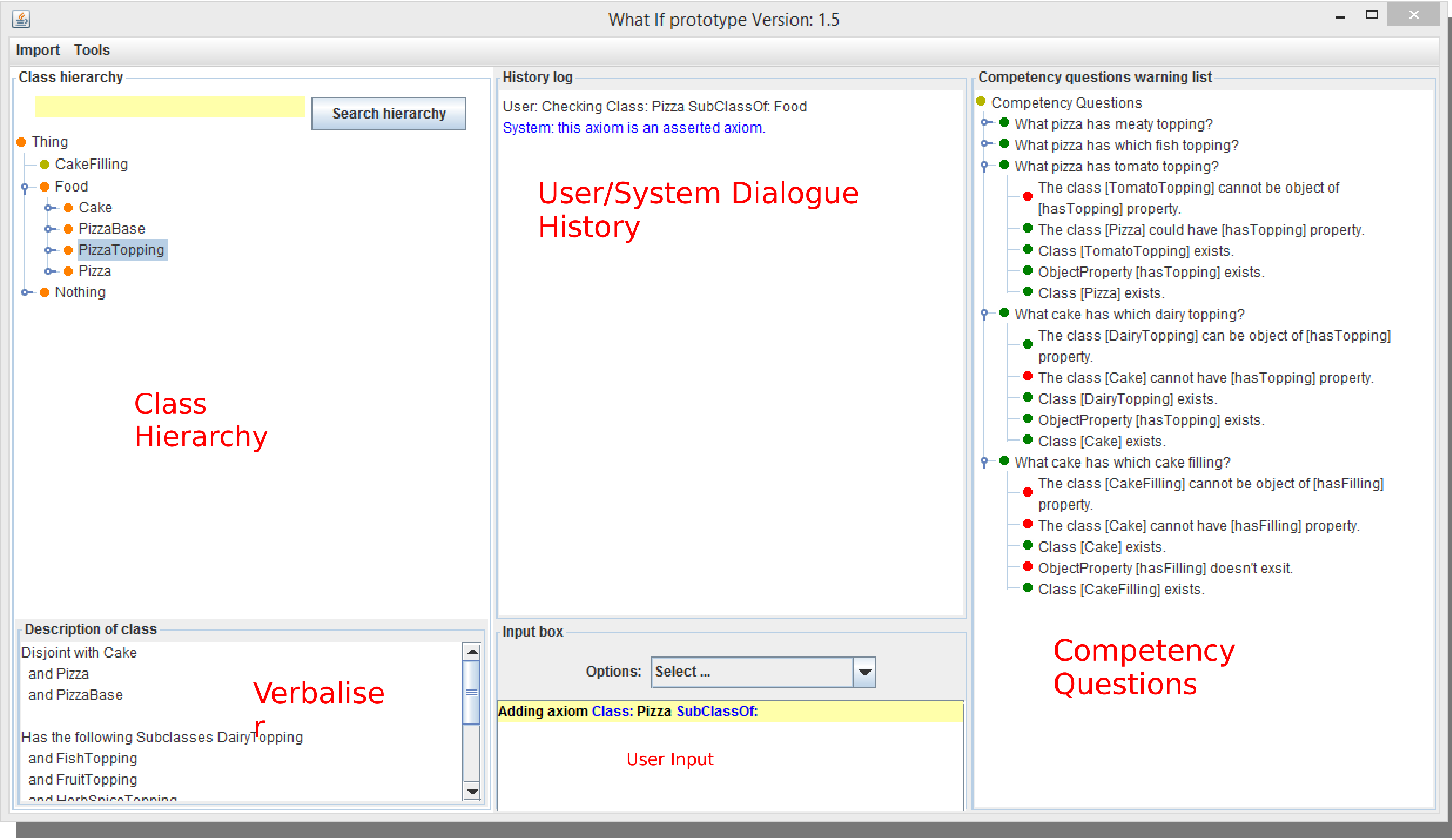
Ontology
Authorin
g

TrOWL
www.trowl.eu

Supporting the CQOA Vision: Basic Ideas

- Using a dialogue-style interface, allowing users to
  - Perform authoring with speech acts in controlled natural languages
  - Review the authoring history and consequences
- Providing feedbacks upon user action so that
  - Users immediately know the consequence of authoring actions
    - In terms of entailments and AT satisfiability
- Registering different reasoning tasks and invoking reasoner on the fly to
  - Responsively update entailment results
  - Constantly monitor satisfiability of ATs

Prototype Interface



- Thing
  - CakeFilling
  - Food
    - Cake
    - PizzaBase
    - PizzaTopping
    - Pizza
  - Nothing

User: Checking Class: Pizza SubClassOf: Food

System: this axiom is an asserted axiom.

- Competency Questions
  - 🟢 What pizza has meaty topping?
  - 🟢 What pizza has which fish topping?
  - 🟢 What pizza has tomato topping?
    - 🔴 The class [TomatoTopping] cannot be object of [hasTopping] property.
    - 🟢 The class [Pizza] could have [hasTopping] property.
    - 🟢 Class [TomatoTopping] exists.
    - 🟢 ObjectProperty [hasTopping] exists.
    - 🟢 Class [Pizza] exists.
  - 🟢 What cake has which dairy topping?
    - 🟢 The class [DairyTopping] can be object of [hasTopping] property.
    - 🔴 The class [Cake] cannot have [hasTopping] property.
    - 🟢 Class [DairyTopping] exists.
    - 🟢 ObjectProperty [hasTopping] exists.
    - 🟢 Class [Cake] exists.
  - 🟢 What cake has which cake filling?
    - 🔴 The class [CakeFilling] cannot be object of [hasFilling] property.
    - 🔴 The class [Cake] cannot have [hasFilling] property.
    - 🟢 Class [Cake] exists.
    - 🔴 ObjectProperty [hasFilling] doesn't exsit.
    - 🟢 Class [CakeFilling] exists.

Disjoint with Cake
and Pizza
and PizzaBase

Has the following Subclasses DairyTopping
and FishTopping
and FruitTopping
and HerbSpiceTopping

Adding axiom Class: Pizza SubClassOf:

User/System Dialogue History

Class Hierarchy

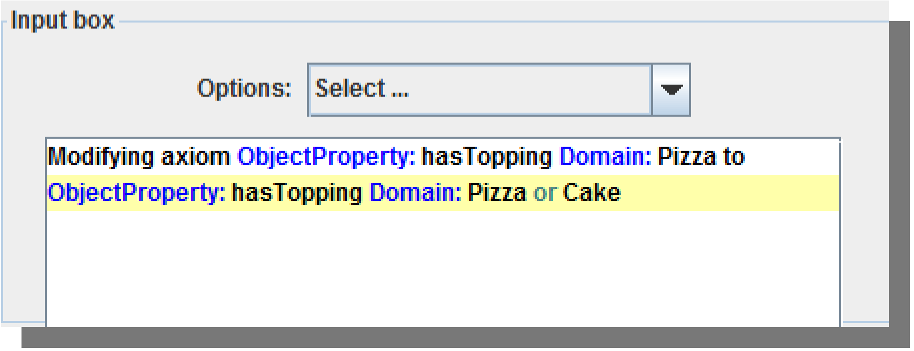
Verbalise
r

User Input

Competency Questions

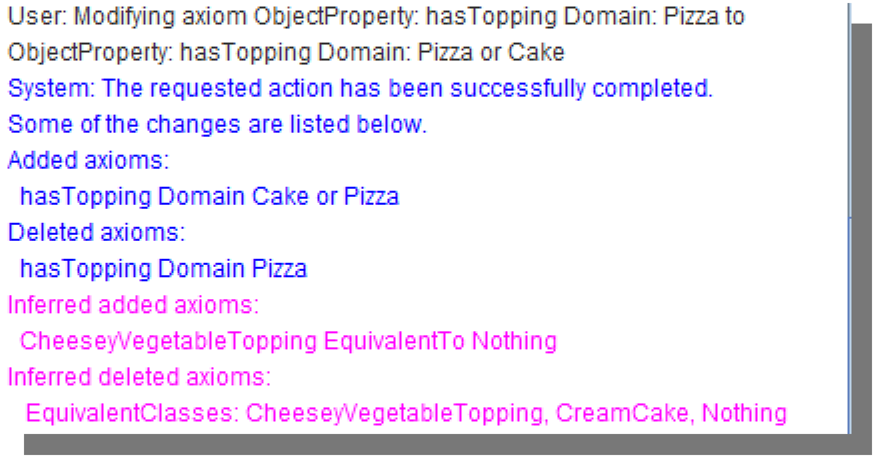
Challenges and Solutions

- Which controlled natural language to use?
  - Comprehensive enough for ontology authoring
  - Easy to learn and understand
  - Easy to parse
- Currently using Manchester Syntax
  - An OWL serialisation, covering all OWL expressiveness
  - Semi-natural
  - Parser available
- User selects a speech acts and then input the CNL formula
- Extending to OWL Simplified English in the future



Options: Select ...

Modifying axiom ObjectProperty: hasTopping Domain: Pizza to
ObjectProperty: hasTopping Domain: Pizza or Cake

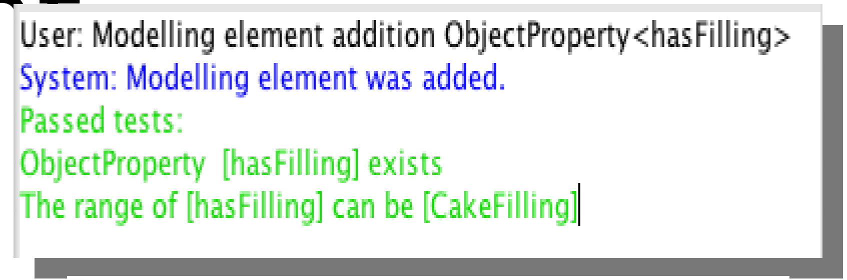


```
User: Modifying axiom ObjectProperty: hasTopping Domain: Pizza to
ObjectProperty: hasTopping Domain: Pizza or Cake
System: The requested action has been successfully completed.
Some of the changes are listed below.
Added axioms:
 hasTopping Domain Cake or Pizza
Deleted axioms:
 hasTopping Domain Pizza
Inferred added axioms:
 CheeseyVegetableTopping EquivalentTo Nothing
Inferred deleted axioms:
  EquivalentClasses: CheeseyVegetableTopping, CreamCake, Nothing
```

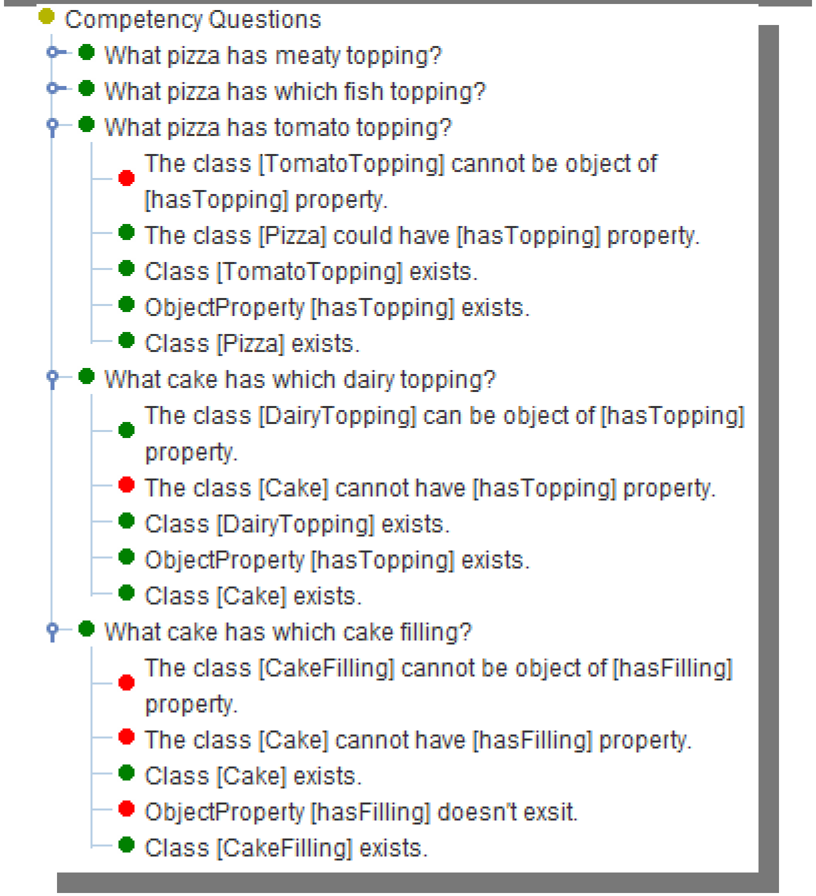
Challenges and Solutions cont

- How to generate the feedback to users?
  - What?
  - When?
  - Where?

- Current feedback mechanism
  - What:
    - Static: the status of the ontology and CQ/AT
    - Dynamic: the consequence of authoring action
  - When:
    - Dynamic: when things ***change***
  - Where:
    - Written feedback in dialogue history
    - Graphical changes in CQ/AT and concept hierarchy



User: Modelling element addition ObjectProperty<hasFilling>
System: Modelling element was added.
Passed tests:
ObjectProperty [hasFilling] exists
The range of [hasFilling] can be [CakeFilling]



● Competency Questions
- ● What pizza has meaty topping?
- ● What pizza has which fish topping?
- ● What pizza has tomato topping?
  - ● The class [TomatoTopping] cannot be object of [hasTopping] property.
  - ● The class [Pizza] could have [hasTopping] property.
  - ● Class [TomatoTopping] exists.
  - ● ObjectProperty [hasTopping] exists.
  - ● Class [Pizza] exists.
- ● What cake has which dairy topping?
  - ● The class [DairyTopping] can be object of [hasTopping] property.
  - ● The class [Cake] cannot have [hasTopping] property.
  - ● Class [DairyTopping] exists.
  - ● ObjectProperty [hasTopping] exists.
  - ● Class [Cake] exists.
- ● What cake has which cake filling?
  - ● The class [CakeFilling] cannot be object of [hasFilling] property.
  - ● The class [Cake] cannot have [hasFilling] property.
  - ● Class [Cake] exists.
  - ● ObjectProperty [hasFilling] doesn't exsit.
  - ● Class [CakeFilling] exists.

Challenges and Solutions cont.

- How to ensure reasoning efficiency

- Currently using approximation-based reasoner TrOWL
  - Approximate OWL 2 DL ontologies into OWL 2 EL ontologies
  - Reduce reasoning complexity
  - Reasoning is automatic and transparent to users

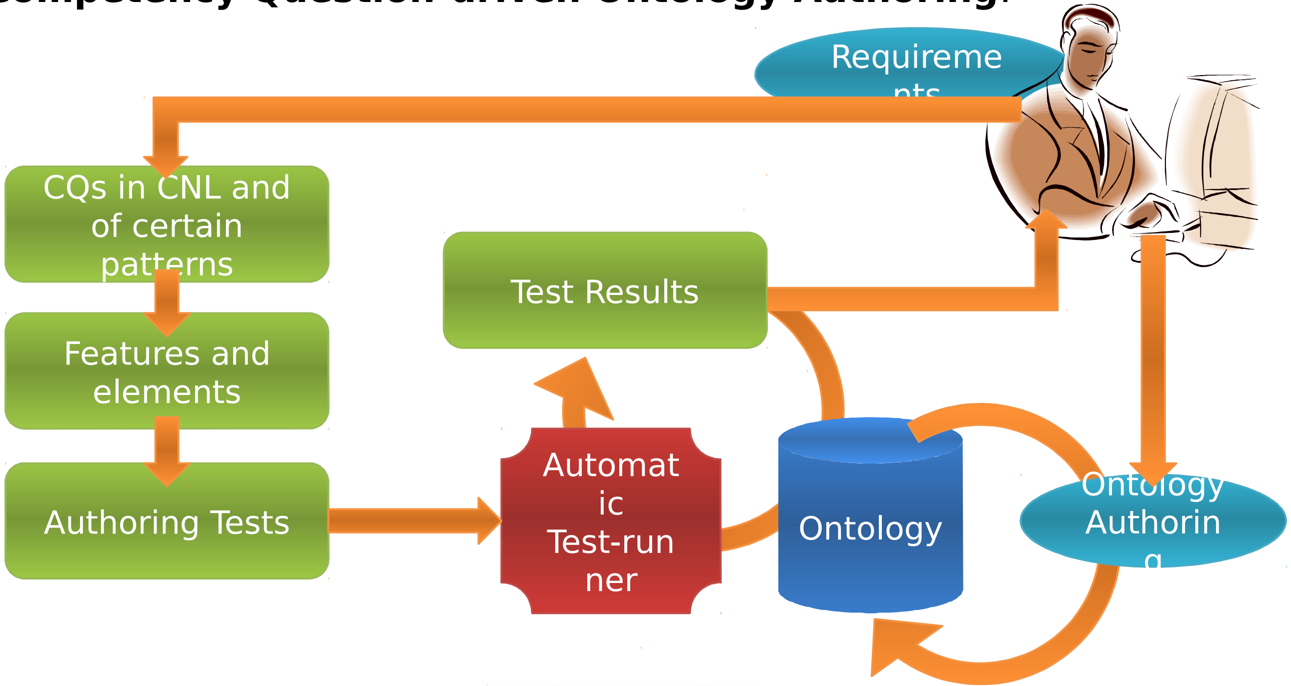
- Moving towards stream reasoning
  - Update only the reasoning results affected by the changes of ontology

Summary of the Work

- An ontology authoring environment can be developed to support Competency Question-driven Ontology Authoring
  - Using a dialogue-based interface
  - Generating informative, comprehensive and intuitive feedbacks
  - Running a reasoner on the fly
- Future challenges
  - Extending the CQ features and presuppositions
  - Investigating different CNL, e.g. OWL Simplified English
  - Developing more informative selection/grouping/ordering strategies for feedbacks
  - Investigating how to provide explanation along with feedbacks
  - Investigating how to provide guidance in addition to feedbacks

Thank You!

- This research has been funded by EPSRC project: *WhatIf: Answering "What if…" questions for Ontology Authoring*.
- The work on CQs has been published:
  - Yuan Ren, Artemis Parvizi, Chris Mellish, Jeff Z. Pan, Kees van Deemter and Robert Stevens. **Towards Competency Question-driven Ontology Authoring**. ESWC2014



Requirements
CQs in CNL and of certain patterns
Features and elements
Authoring Tests
Test Results
Automatic Test-runner
Ontology
Ontology Authoring