



TAO Bootstrapping Methodology

Hai Wang

University of Southampton

Funded by: European Commission – 6th Framework
Project Reference: IST-2004-026460



Goal of TAO

3 Semantic Web Services (SWS), and Service-Oriented Architectures (SOA)

- 2 Enterprise Applications Integration (EAI) will become **less costly** and more **reliable**
- 2 B2B and B2C eCommerce systems will become
 - ± more **reusable**
 - ± more **dynamic and adaptive**
 - ± more **scalable**
 - ± more **saleable**

3 **However**, the industrial take-up of Semantic Web technologies in developing services and applications has been slower than expected

3 **Goal of TAO**

- 2 Systematic methodology and support tools to migrate existing legacy applications **to open, semantics-based SOA**

Outline

3 Introduction

- 2 The need for a methodology
- 2 Challenges

3 The use of Formal Models

- 2 Comparison between OWL-S and WSMO
 - ± View-based (informal) comparison
 - ± Formal Model comparison

3 Procedural Methodology

- 2 Breakdown
- 2 Example - Amazon Services



Goal of TAO

from the Methodology perspective

3 Support the creation of semantic web services

2 Engineer starts with:

- ± Existing legacy system
- ± Domain knowledge is generated from several sources
 - *Documentation generated by service provider*
 - *Domain information*
 - *User forums / blogs / community*


2 Needs to generate:

- ± Service / Domain Ontologies
- ± Annotated Web Services Semantically
- ± Annotated Documentation to support
 - *User access to Services*
 - *Subsequent Development/Refinement*

3 Challenge

- #### 2 To develop a methodology that describes (abstractly) how to transition from existing legacy system to Semantically Rich Service Environment

Methodology Challenges

- 
- 3 Understanding salient or mandatory components necessary for representing Semantic Web Services
 - 2 Task is **not to develop** new SWS frameworks!
 - 2 Task is to **understand what facts that existing frameworks require** to be effectively used
 - 3 Generate and utilise ontology for supporting:
 1. the annotation of services
 - *Ontologies should support service-based activities (e.g. search, composition) with business/service community*
 2. the annotation of user documentation
 - *Ontologies should support the search and retrieval (question / answer) of documentation by user/developer community*
 - Generate a **single** ontology for both

Outline

3 Introduction

- 2 The need for a methodology
- 2 Challenges

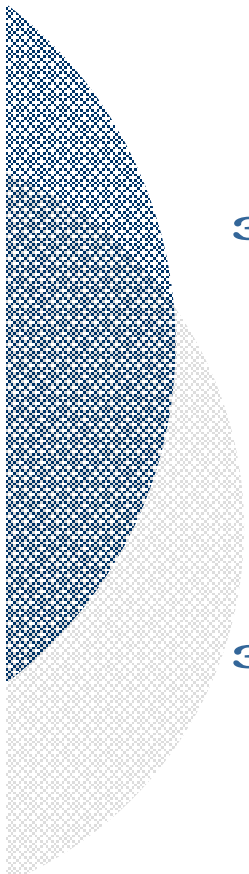
3 The use of Formal Models

- 2 Model and evaluate the main representational ontologies from a software engineering perspective

3 Procedural Methodology



Formal models of OWL-S/WSMO

- 
- 3 Establish formal models of the candidate Semantic Web Service Frameworks
 - ± To better understand advantages / disadvantages
 - ± Align final methodology to software engineering methodologies
 - 3 Problem with the existing SWS standards
 - 2 Syntax, static and dynamic semantics of the languages are separately described, informally or semi-formally.
 - ± *Redundancy and contradiction*
 - ± *Lack of formal semantics*
 - ± *Difficult to be extended and reused consistently*
 - 3 To support common understanding and facilitate standardization and tool development, a complete, consistent, rigorous and extendible formal model of OWL-S/WSMO is desirable

Formal models of OWL-S/WSMO

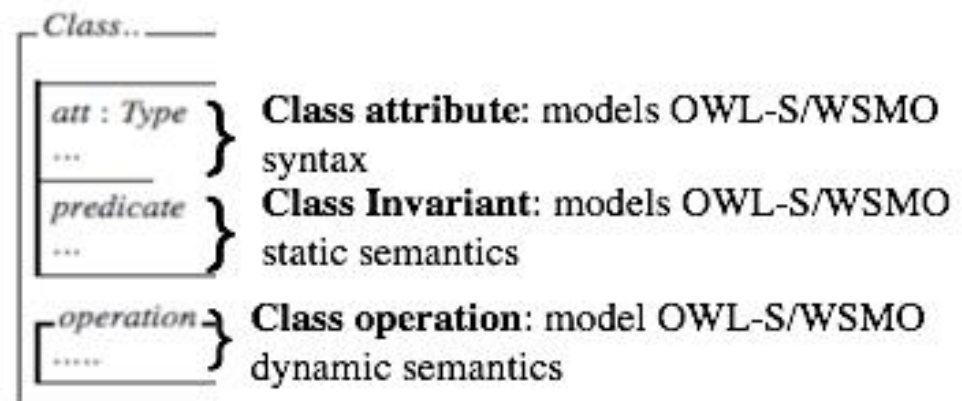
3 A single denotation model of all aspects of OWL-S/WSMO using **Object-Z (OZ)**

2 Why Object-Z --

- ± an OO extension of the Z formal language
- ± Provides good support for modularity and reusability, thus enabling clarity of model
- ± The semantics of OZ are well studied
- ± Provides useful modeling constructs

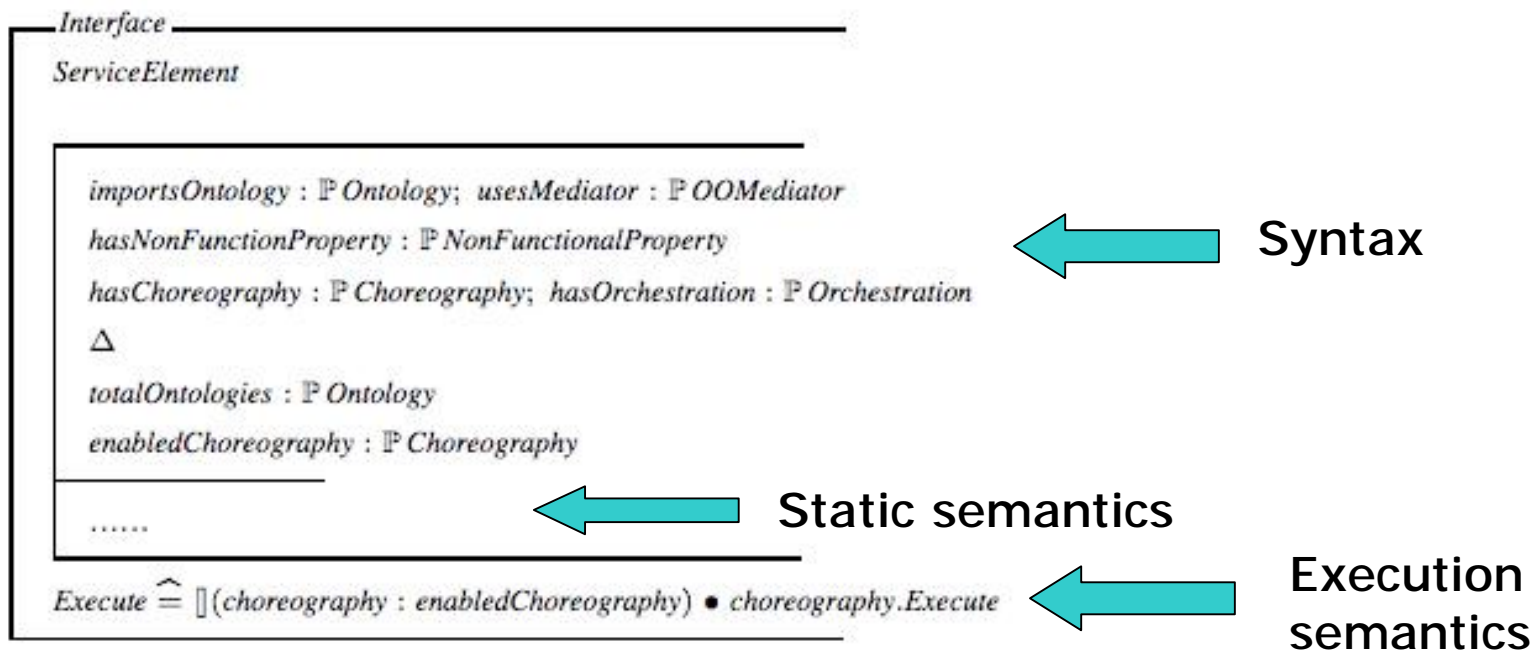
2 Decomposed into

- ± Static Model
- ± Dynamic Model



Formal models of OWL-S/WSMO

3 For example -- WSMO Interface



Formal models of OWL-S/WSMO

3 Benefits of the formal models

- 2 Checking the consistency of WSMO/OWL-S languages using OZ tools
- 2 Providing a unified, consistent and precise description of WSMO/OWL-S
- 2 Reasoning WSMO/OWL-S by using exiting formal tools directly
- 2 Formal comparing between WSMO/OWL-S

Comparing OWL-S and WSMO

3 A view-based comparison

2 System **view**

“a representation of a whole system from the perspective of a related set of concerns.” -- *IEEE Std-1471-2000*

2 More **comprehensive** and **complete** comparison

2 Different viewpoints

± Service **requesters**

± Service **providers**

± Service **brokers**

Comparing OWL-S and WSMO

3 The requester's viewpoint -- *How the client's request is described?*

2 Profile vs. Goal

2 Single vs. Separate view

2 other issues:

± *Non-functional property*

- *WSMO predefines many NF properties*
- *OWL-S leave the flexibility to users*

± *Request capacity*

- *WSMO -- state machine based*
- *OWL-S -- IOPE model based*

± *Reusability of requests*

- *WSMO -- GGMediator*
- *OWL-S -- Profiles subsumption*

Comparing OWL-S and WSMO

3 The provider's viewpoint -- *How the service is described and advertised?*

2 Service *capability* description

± OWL-S (*IOPE*) and WSMO (*State machine*)

± Precondition *and* assumption

± *Logic formulas*

◦ WSMO-- WSMX

◦ OWL-S -- no predefined

± *Dual descriptions of OWL-S service*

◦ *Profile and service model*

Comparing OWL-S and WSMO

3 The broker's viewpoint -- *How the service is used?*

2 *Choreography*

- ± WSMO -- "Choreography"
- ± OWL-S does not provide an explicit definition of choreography but instead focuses on how the atomic processes are grounded.

2 *Orchestration*

- ± WSMO -- "Orchestration"
- ± OWL-S does not provide an explicit definition of orchestration but instead focuses on a process based description of how complex Web services invoke atomic Web services.

Comparing OWL-S and WSMO

3 Other issues

2 Ontology languages -- *F-logic* vs. *OWL*

- ± Permissions vs prohibitions
- ± Composition vs enumeration
- ± Meta-modelling
- ± Frames are analogous to familiar paradigms
- ± **Open** vs **closed** world assumptions.
- ± Unique name assumption

2 Mediator

Comparing OWL-S and WSMO

3 Provide an abstract perspective of the ontologies of both

≈ Understand the differences to avoid the mistakes of a 1-to-1 comparison

± OWL-S is an ontology

± WSMO is a framework

◦ Includes more than just an ontology



≈ OWL-S

± uses the agent planning approach and models Web services as processes

± is better integrated with the existing Web standards

≈ WSMO

± based on problem solving techniques and models Web services as state machine

± provides mature execution environments

± the explicit definitions of choreography and orchestration enable the better service reuse and composition

Selecting Service Framework

- ◆ The current work has converged on using **SA-WSDL** for semantically annotating Web Services
 - ◆ Neutral with respect to current ontology representation languages (WSML and OWL)
 - ◆ Provides a simple framework to facilitate emphasis on the ontological description of the domain, rather than concerns over generating procedural axioms
 - ◆ Choreography and Orchestration issues are being put aside for now
 - ◆ Convergence on a single ontology for both service and documentation annotation
- ◆ Only W3C recommendation so far

TAO Formal Methodology Publications

3 Journals / Book chapter

- ± H. H. Wang, N. Gibbins, T.R. Payne, A.Saleh. “Transitioning Applications to Semantic Web Services: An Automated Formal Approach”, *Journal of Interoperability in Business Information Systems*, 2008

3 Conference:

≈ WSMO Papers

- ± H.H. Wang, T.R. Payne, N.Gibbins and A.Saleh. “Formal Model of Semantic Web Service Ontology (WSMO) Execution”, ICECCS'08
- ± H.H. Wang, N.Gibbins, T.R. Payne, A.Saleh, and J. Sun. “A Formal Semantics Model of WSMO”, ICECCS'07

≈ OWL-S Papers

- ± H.H. Wang, A.Saleh, T.R. Payne, and N.Gibbins, “Formal Specification of OWL-S with Object-Z”, 1st ESWC'07 Wkshp on OWL-S
- ± H.H. Wang, T.R. Payne, N.Gibbins and A.Saleh. “Formal Specification of OWL-S with Object-Z: the Dynamic Aspect”, WISE07
- ± H.H. Wang, A.Saleh, T.R. Payne, and N.Gibbins, “Formal Specification of OWL-S with Object-Z: the Static Aspect” *Web Intelligence (WI) '07*

Outline

3 Introduction

- 2 The need for a methodology
- 2 Challenges

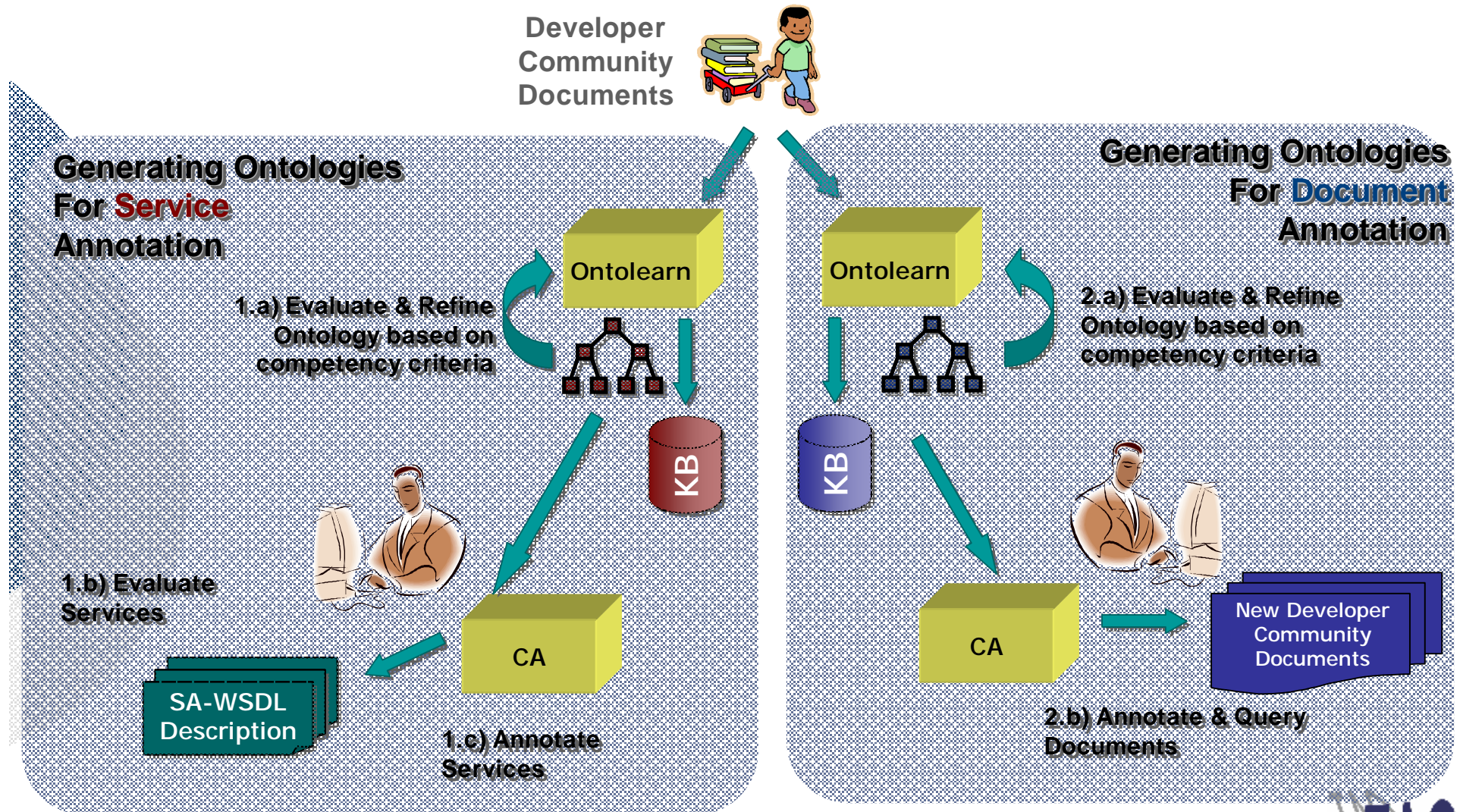
3 The use of Formal Models

3 Procedural Methodology

- 2 Define the stages when actually generating the services using TAO Tools
- 2 Provide a set of worked examples to guide developers and engineers



Methodology Architecture



Procedural Methodology Key Steps

3 Key Stages within Evolving Methodology

1. Source Document Identification
2. Learning the Ontology
3. Annotating...

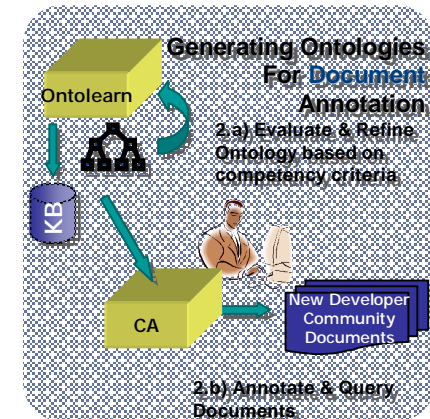
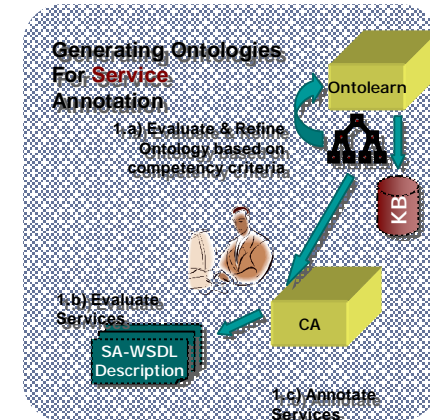
3 Services

- a) Creating Service
- b) Annotating Service
- c) Evaluating Resulting Service

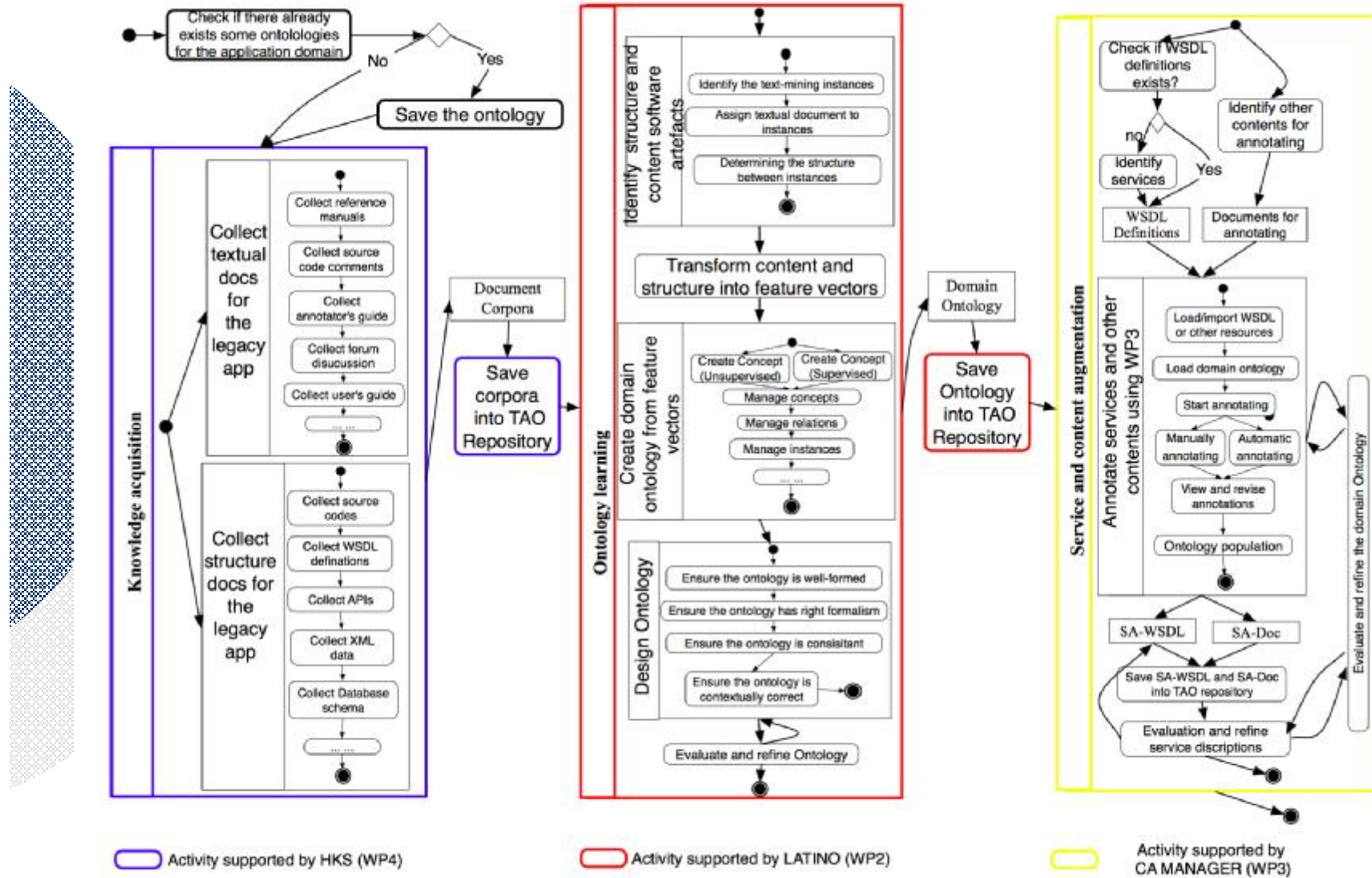
3 Content Augmentation

- a) Supporting Annotation

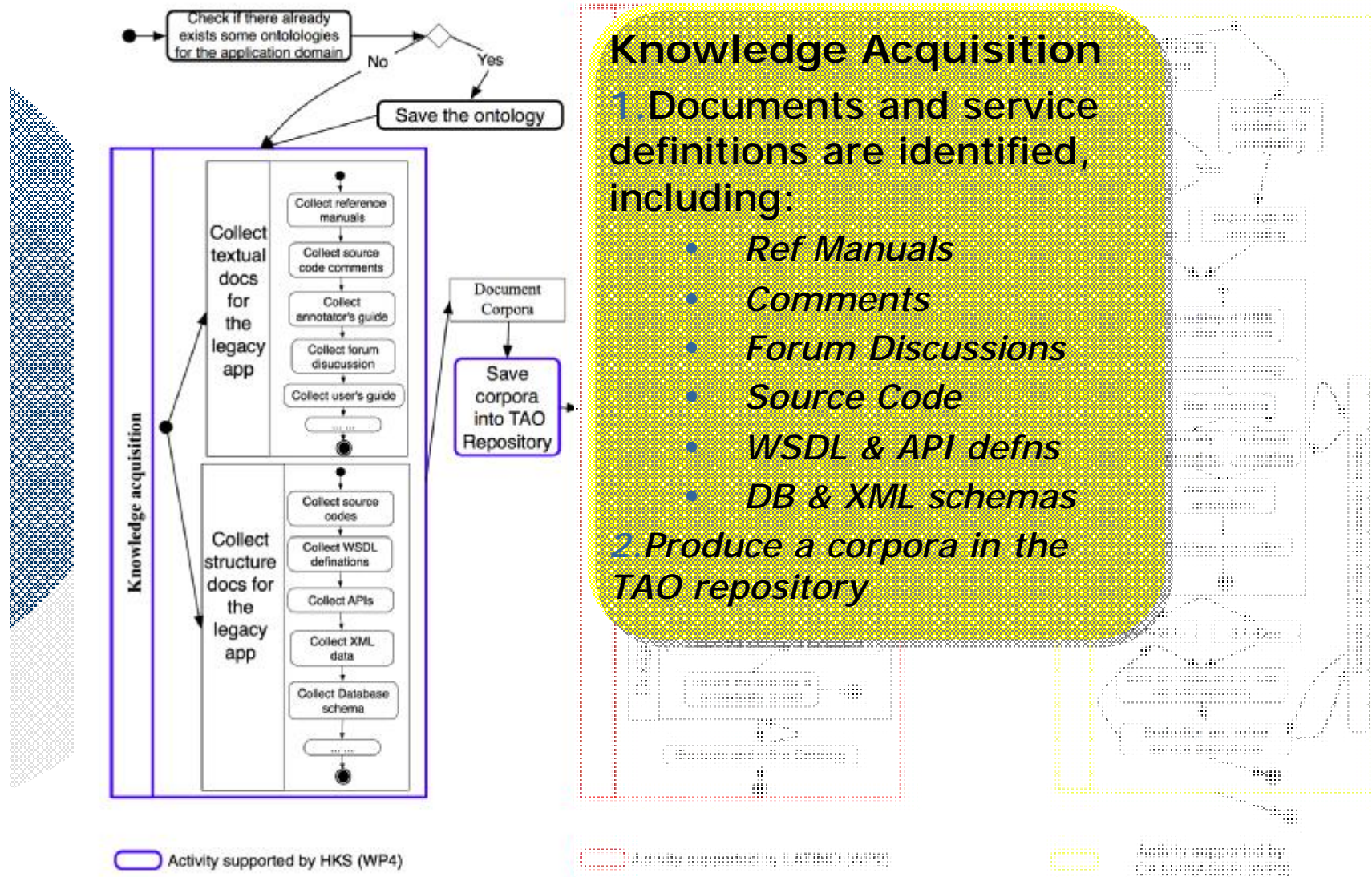
4. Ontology Evaluation



Overview of Methodology



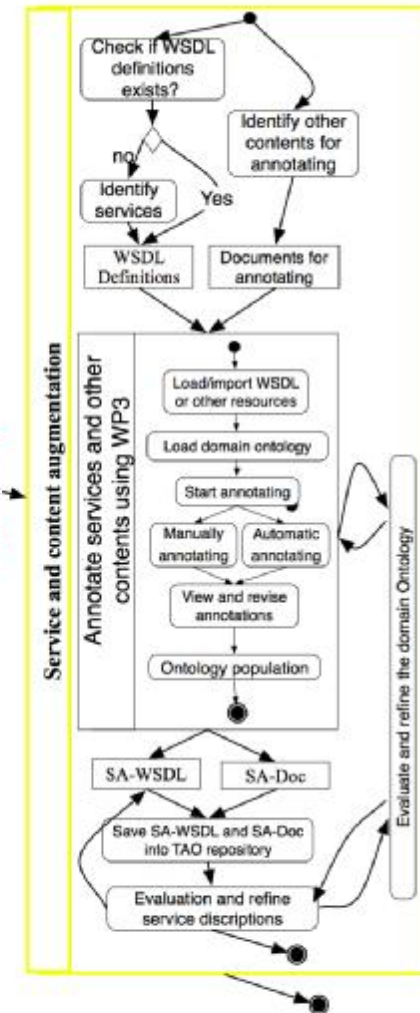
Overview of revised Methodology



Overview of revised Methodology

Service and Document Annotation

1. Construct WSDL documents (if necessary):
2. Annotate using CA / Gate tools
 1. *Load ontology & WSDL / document resources*
 2. *Automated initial annotation*
 3. *Refine / revise annotations*
 4. *Use other CA tools to refine ontologies*
3. Generate SA-WSDL from annotated WSDL description



Activity supported by W3C (RFP4)

Activity supported by IATFD (RFP3)

Activity supported by CA MANAGER (WP3)



End

Questions? Discussion welcome!