

Ingeniería de ontologías

Metodologías ágiles de desarrollo de ontologías
Desarrollo guiado por pruebas



Desarrollo de ontologías: metodologías ágiles

- Ontologías de alcance limitado (no extremadamente complejas)
- Datos de prueba reales, definir Abox en forma temprana (importancia de los datos)
- Modelos autoexplicativos
- Escenarios de uso

Enfoques híbridos

Simplified Agile Methodology for Ontology Development (SAMOD)

Proceso de desarrollo

- simple
- iterativo (3 pasos)
- guiado por pruebas
- escenarios y datos de ejemplo
- ontologías “siempre listas para usar” y “entendibles por humanos”

Roles

- Experto de dominio
- Ingeniero de ontologías

SAMOD – Requerimientos, modelos

Escenario motivador – *MS* (con ejemplos)

Pregunta de competencia – *CQ*

- Requerimiento informal, con respuesta esperada, respuestas de ejemplo
- Jerarquía de preguntas

SQ: consulta SPARQL que corresponde a *CQ*

Glosario de términos – *GoT*

En cada **iteración** i_n se libera una versión del modelo final de la ontología

Modelo actual en $i_n \rightarrow$ versión del modelo liberada en i_{n-1}

Modelet \rightarrow modelo de i_n , independiente del modelo actual

SAMOD – Pruebas: modelo, datos, consultas

Model test (MS, TBox, GoT): validez del modelo

- TBox consistente (test unitarios O.K.) → FORMAL
- TBox cubre MS y GoT adecuado → INFORMAL

Data test (MS, TBox, ABox, ejemplos MS): validez del modelo y los datos

- TBox consistente con ABox
- ABox representa ejemplos del MS, TBox autoexplicado (adecuado para generar datos)

Query test (TBox, ABox, CQ, SQ): validez del modelo, los datos y las consultas

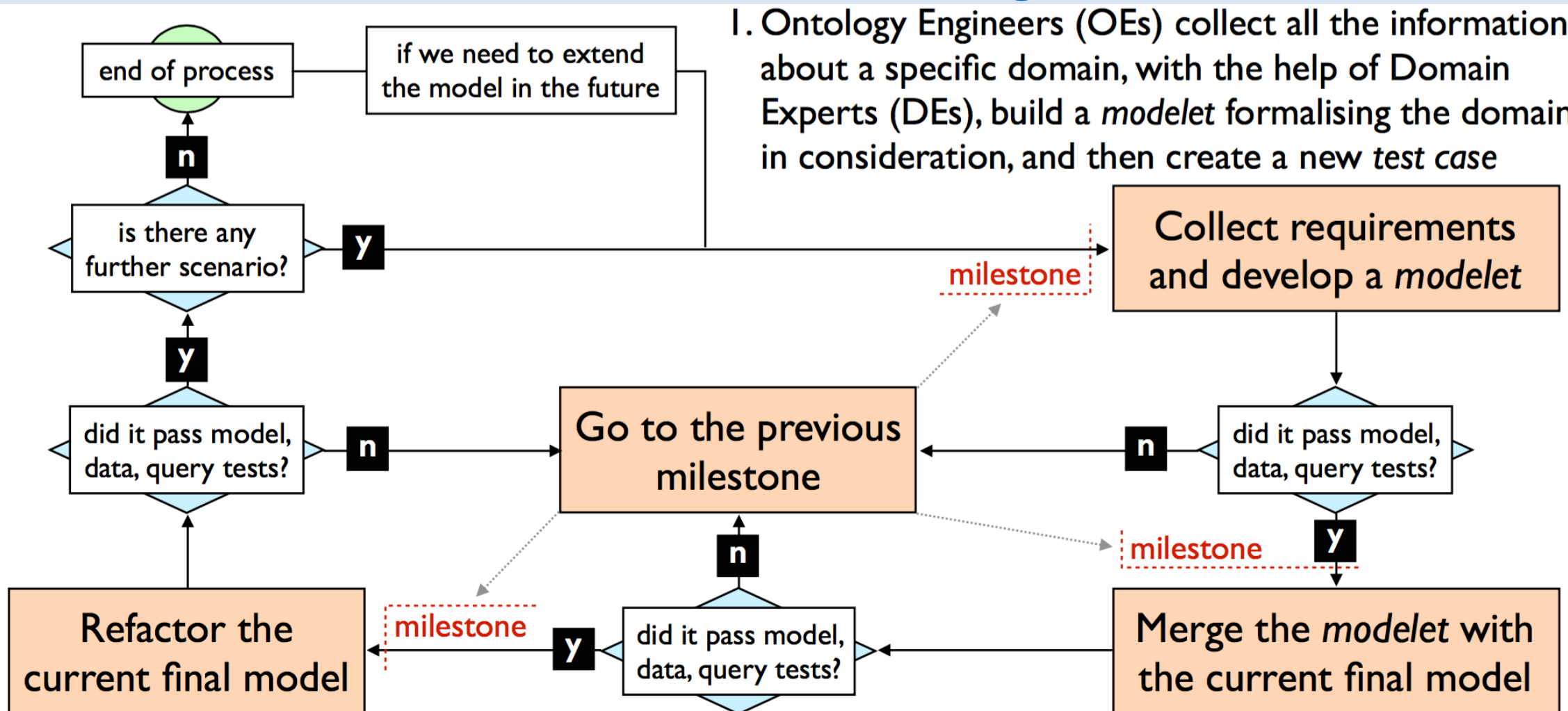
- Cada SQ ejecuta correctamente para TBox + ABox
- Cada SQ mapeada a una CQ, al ejecutarla el resultado corresponde al resultado esperado de CQ

Caso de prueba: T(MS, CQ, GoT, TBox, ABox, SQ)

BoT: conjunto de casos de prueba

SAMOD - Metodología

1. Ontology Engineers (OEs) collect all the information about a specific domain, with the help of Domain Experts (DEs), build a *modelet* formalising the domain in consideration, and then create a new *test case*



3. OEs refactor (important: reuse existing knowledge) the current model, focussing on the last part added in the previous step

2. OEs merge the *modelet* of the new *test case* with the current model produced by the end of the last process iteration (first iteration: *modelet* becomes current model)

SAMOD – Paso 1: requerimientos y modelet

Escribir MS y CQs

Al crear el modelet:

- Mantener en lo posible pocos conceptos y relaciones
- Usar patrones de diseño
- Enfoque middle-out para modelar conceptos
- Mantener el modelo simple (sin agregar elementos innecesarios)
- Nombres de conceptos y relaciones autoexplicativos

Validaciones: modelo, datos y consultas (formal e informal)

$T_n (MS_n, CQ_n, GoT_n, modelet_n, ABox_n, SQ_n)$

SAMOD – Paso 2: merge modelet al modelo actual

$$TBox_n = TBox_{n-1} + modelet_n$$

fusionando entidades semánticamente equivalentes

Actualizar BoT:

- Intercambiar $TBox_{n-1}$ por $TBox_n$
- Ajustar Abox y SQ con nuevos nombres de entidades

Validaciones: modelo, datos y consultas (formal)

$Tbox_n \rightarrow$ **modelo actual**

SAMOD – Paso 3: refactorizar el modelo actual

Considerando principalmente la parte agregada en la actual iteración, *refactorizar* aplicando las siguientes prácticas:

- Reusar modelos existentes (reúso o mapping de entidades)
- Agregar anotaciones, comentarios, procedencia
- Enriquecer el modelo actual con nuevas restricciones
 - Keys, transitividad, simetría, property chains, inversas → inferencias
 - inferir conocimiento
- NO SOBRESPECIFICAR (no declarar lo que se puede inferir)

Validaciones: modelo, datos y consultas

SAMOD – Ejemplo – Recetas – Iteración 1

MS₁: Se desea una aplicación web de descubrimiento de recetas publicadas en diferentes sitios web. De las recetas se debe poder obtener sus ingredientes y su proceso de elaboración.

CQ_{1.1}: ¿Cuáles son los ingredientes de una receta dada? ----- **SQ_{1.1}**

CQ_{1.2}: ¿Cuáles son las instrucciones para elaborar una receta dada?

GoT₁: Receta, Ingredientes, Instrucciones, Comida

Modelet₁:

$Receta \sqcap SituacionInicial \sqsubseteq \perp$ $Receta \sqcap Proceso \sqsubseteq \perp$
 $Receta \sqcap Instruccion \sqsubseteq \perp$ $Proceso \sqcap Instruccion \sqsubseteq \perp$
 $Receta \sqsubseteq \exists requiere.SituacionInicial$
 $\exists describe.T \sqsubseteq Receta$ $T \sqsubseteq \forall describe.Proceso$

```

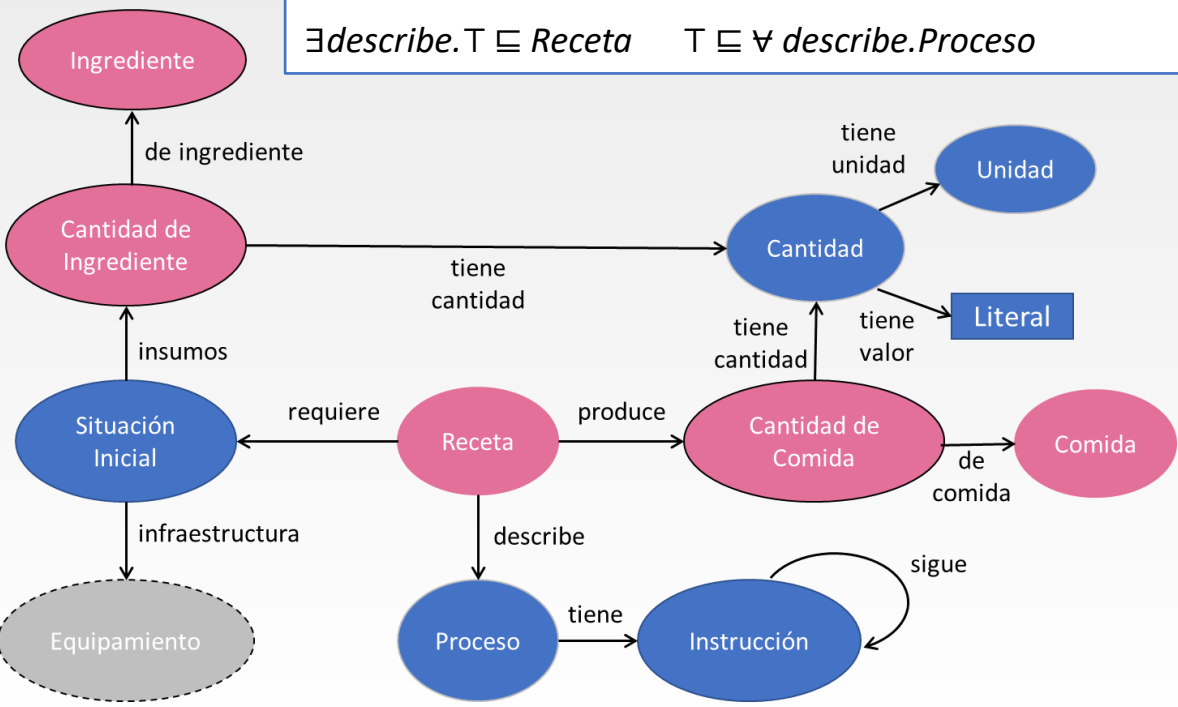
select ?ingrediente
where
{
    rec:asado req:requiere ?sitInicial .
    ?sitInicial rec:insumos ?cantIngrediente .
    ?cantIngrediente rec:deIngrediente ?ingrediente.
}
    
```

```

select ?instruccion
where
{
    rec:asado req:describe ?proceso .
    {?proceso rec:tiene ?instruccion .}
    UNION
    {?proceso rec:tiene/rec:sigue* ?instruccion .}
}
    
```

SQ_{1.2}

T₁(MS₁, CQ₁, GoT₁, modelet₁, ABox₁, SQ₁)



SAMOD – Ejemplo – Recetas – Iteración 2

MS₂: Se desea poder obtener también el contenido nutricional de las recetas.

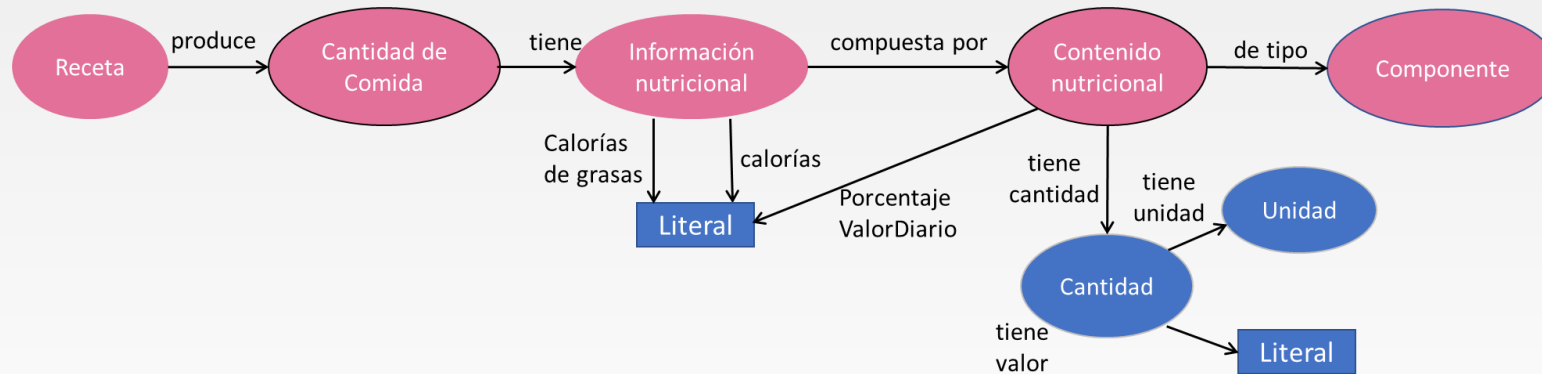
CQ_{2,1}: ¿Cuáles son los componentes nutricionales de una receta dada? ----- **SQ_{2,1}**

GoT₂: Información nutricional, Contenido nutricional, calorías, porcentaje valor diario

Modelet₂:

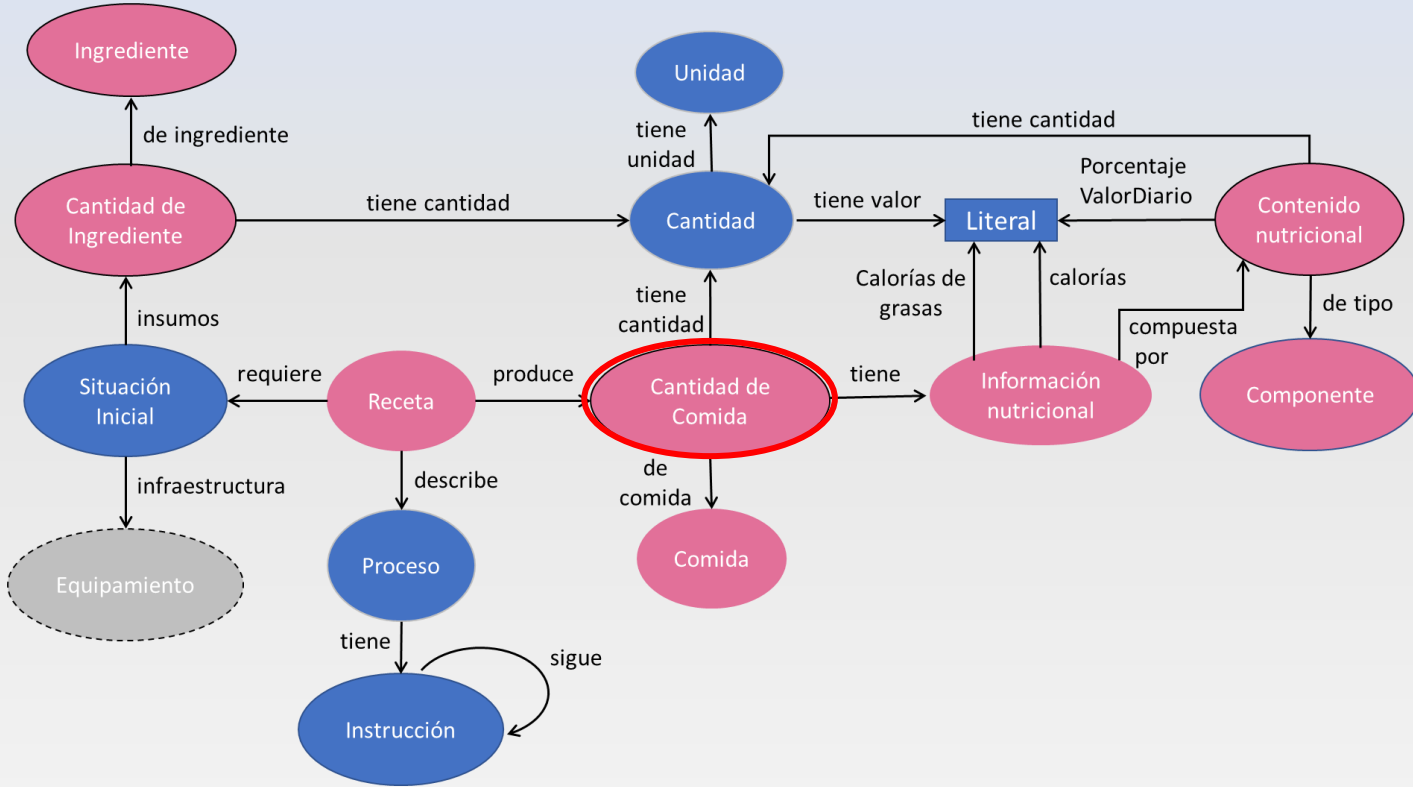
$InformacionNutricional \sqcap ContenidoNutricional \sqsubseteq \perp$
 $ContenidoNutricional \sqcap Componente \sqsubseteq \perp$
 $ContenidoNutricional \sqsubseteq \exists deTipo.Componente$
 $\exists deTipo.\top \sqsubseteq ContenidoNutricional \quad \top \sqsubseteq \forall deTipo.Componente$

```
select ?componente
where
{
  rec:asado req:produce ?cantComida .
  ?cantComida rec:tiene ?infNutricional .
  ?infNutricional rec:compuestaPor ?contNutricional.
  ?contNutricional rec:deTipo ?componente .
}
```



$T_2(MS_2, CQ_2, GoT_2, modelet_2, ABox_2, SQ_2)$

SAMOD – Ejemplo – Recetas – Iteración 2: merge y refactorización



$Receta \sqcap SituacionInicial \sqsubseteq \perp$ $Receta \sqcap Proceso \sqsubseteq \perp$
 $Receta \sqcap Instruccion \sqsubseteq \perp$ $Proceso \sqcap Instruccion \sqsubseteq \perp$
 $Receta \sqsubseteq \exists requiere.SituacionInicial$
 $\exists describe.T \sqsubseteq Receta$ $T \sqsubseteq \forall describe.Proceso$
 $InformacionNutricional \sqcap ContenidoNutricional \sqsubseteq \perp$
 $ContenidoNutricional \sqcap Componente \sqsubseteq \perp$
 $ContenidoNutricional \sqsubseteq \exists deTipo.Componente$
 $\exists deTipo.T \sqsubseteq ContenidoNutricional$ $T \sqsubseteq \forall deTipo.Componente$
requiere o insumos o delIngrediente \sqsubseteq tieneIngrediente

$BoT = \{T_1, T_2\}$

Bibliografía

Silvio Peroni. A simplified agile methodology for ontology development. OWLED, 2016

Silvio Peroni. SAMOD: an agile methodology for the development of ontologies. figshare. <http://dx.doi.org/10.6084/m9.figshare.3189769>, 2016

C. Maria Keet and A. Lawrynowicz. Test-driven development of ontologies. ESWC, 2016