

TRABAJO FINAL

Presentación Proceso y Ontología - Ejemplo

Diciembre 2009

- Bertoni , Francisco
- Villanueva, Sebastián

Índice

Introducción	3
Arquitectura General del Sistema	4
Módulos del Sistema	5
Responsabilidades.....	9
Modulo Reasoner	10
Modulo Ontology Analyzer	12
Ontología.....	13
Conceptos representados	14
Explicación de las relaciones	15
Entrenamiento de la Ontología.....	17
Métricas.....	18

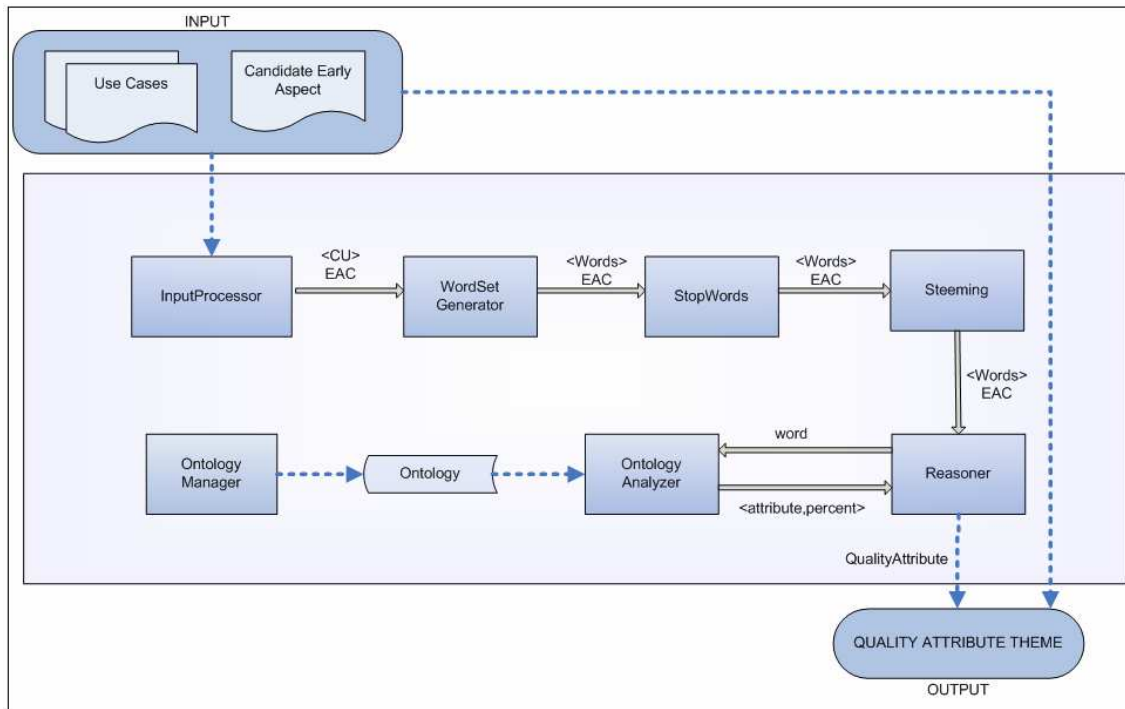
Introducción

En este documento se presentara el proceso y los artefactos definidos para alcanzar el objetivo de este trabajo final: *identificar el grado de pertenencia de un conjunto de casos de usos y de early aspects candidatos que los relacionan, a un atributo de calidad*. La detección y el análisis de pertenencia se realiza mediante la utilización de técnicas de análisis de texto y, sobre todo, a través del razonamiento sobre una ontología que se ha definido para representar el dominio en cuestión: atributos y escenarios de calidad. Dicha ontología está fundamentada en las definiciones realizadas por el SEI¹ respecto al dominio de análisis. La ontología y su uso también serán explicadas en detalle en el presente documento.

Por último, se realizara un ejemplo sobre el cual se tomaran distintas métricas para evaluar la técnica propuesta.

¹ Bass, L., Clements, P., Kazman, R., *Software architecture in practice*, 2da edición, Addison-Wesley, 2003.

Arquitectura General del Sistema

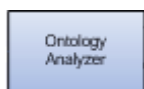


- Diagrama de componentes y conectores -

Referencias



Información estática



Módulo



Lectura de información estática



Flujo de datos



Elementos de una lista



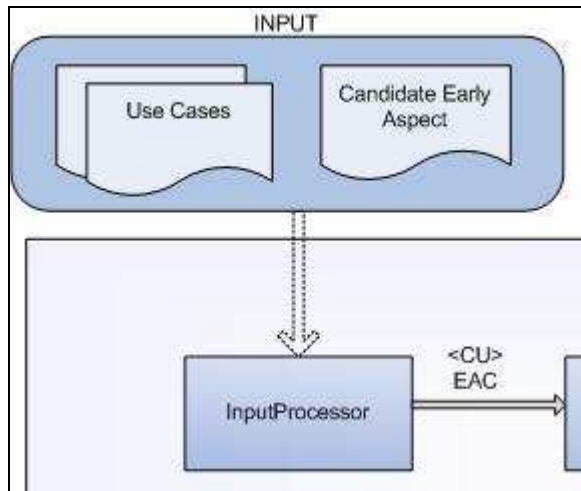
Entrada del sistema



Salida del sistema

Módulos del Sistema

INPUT PROCESSOR



Entrada: casos de uso y early aspects candidatos.

Salida: representación interna de Casos de Uso y early aspects candidatos.

El modulo **InputProcessor** es el encargado de leer los documentos de entrada en el formato que vengan (pdf, doc, xml, etc) y llevarlo a una representación intermedia. Si bien aun no se han definido las estructuras de clases para representar los diferentes objetos, se supone que en este modulo se instanciaran objetos del tipo "Use Case" y "Early Aspect Candidate" que serán entregados al siguiente módulo.

También es cierto que en esta primera etapa se utilizarán solo los template de Rational como especificación de los casos de uso, que vienen en formato doc.

WORDSET GENERATOR



Entrada: representación interna de los casos de uso y del early aspect.

Salida: listado de palabras y representación interna del early aspect.

Este modulo extrae las palabras de las diferentes secciones de los casos de uso (nombre, descripción, flujos, etc.) y devuelve un listado de las mismas.

Tanto en este módulo como en los siguientes, el early aspect seguirá fluyendo sin modificaciones. Más adelante se evaluará como utilizar la información del early aspect una vez que se obtengan los resultados sin tenerlo en cuenta.

STOP WORDS

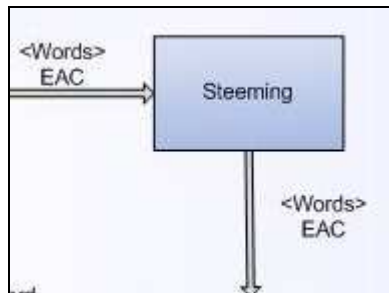


Entrada: lista de palabras y early aspect candidato.

Salida: lista de palabras modificada y early aspect candidato.

Elimina las denominadas "stop words" de la lista de palabras. Estas palabras son aquellas que tienen poca relevancia, como por ejemplo: about, actually, after, again, against, best, etc.

STEMMING

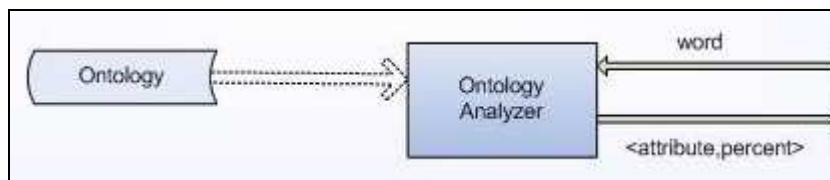


Entrada: lista de palabras y early aspect candidato.

Salida: lista de palabras modificada y early aspect candidato.

Este modulo aplica la técnica de Stemming sobre cada una de las palabras de la lista. De esta forma, las palabras que conforman la lista se reducen a su raíz.

ONTOLOGY ANALYZER



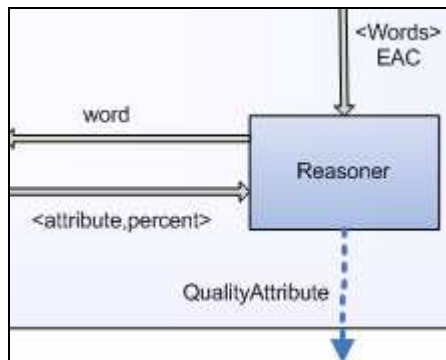
Entrada: palabra, Ontología.

Salida: lista de pares atributo-porcentaje.

Toma como entrada una palabra e identifica con cuantos atributos de calidad se relaciona y con qué porcentaje lo hace con cada uno. Para ello, consulta el conjunto de los escenarios particulares en los cuales aparece esa palabra (en cualquier parte del escenario: estimulo, enviornment, etc.). Luego, de ese conjunto, calcula el porcentaje de escenarios que se relacionan con un atributo de calidad para cada atributo de calidad devolviendo la lista de pares atributo de calidad-porcentaje.

Debido a la importancia de este módulo, se realizará una explicación detallada del mismo más adelante donde se además de mostrará la ontología que será utilizada.

REASONER



Entrada: lista de palabras y early aspect candidato.

Salida: Quality Attribute.

Este módulo por cada palabra de la lista de palabras consulta al módulo Ontology Analyzer para obtener el grado de relación que tiene cada palabra de la lista con los atributos de calidad. A partir de ello, calcula, para la lista de palabras, cual es el atributo de calidad con que se relaciona en mayor medida. Con este atributo de calidad generará el Quality Attribute Theme.

Este módulo, al igual que el anterior, será explicado con mayor detalle más adelante.

ONTLOGY MANAGER

El Ontology Manager es el modulo encargado de brindar una capa de abstracción sobre la ontología. Básicamente brinda métodos que permiten crear, borrar o modificar instancias; crear, borrar o modificar relaciones; consultar instancias o relaciones existentes, etc.

En este momento lo ubicamos a este modulo simplemente comunicandose con la ontología, pero es probable que también se encuentre entre el Modulo Reasoner y la Ontologia.

Responsabilidades

InputProcessor:

- Leer entrada
- Transformar entrada en representación intermedia

WordSetGenerator

- Generar lista de palabras alfanuméricas

StopWords

- Eliminar Stop Words

Stemming

- Reducir palabras a su raíz

Reasoner

- Procesar ranking quality attributes
- Retornar Quality Attribute Theme ganador

Ontology Analyzer

- Identificar QAs relacionado (escenarios particulares y generales)
- Determinar porcentajes

Ontology Manager

- Agregar, eliminar, modificar y consultar instancias.
- Agregar, eliminar, modificar y consultar relaciones.
- Proveer una interfaz a nuestro sistema que permita el manejo de la ontología.

Modulo Reasoner

El módulo *Reasoner* tiene como entrada una lista de términos, extraídos de los casos de uso que se encuentran agrupados mediante un *Early Aspect*. Por cada uno de estos términos se consultará al modulo *Ontology Analyzer*, que devolverá el grado de correspondencia que tiene con cada atributo de calidad.

A medida que se vaya recorriendo la lista y consultado al módulo *Ontology Analyzer* se irá generando una tabla. Una vez finalizada la iteración sobre las palabras se utilizará esta tabla para calcular el promedio de correspondencia de todas las palabras de la lista con cada atributo de calidad.

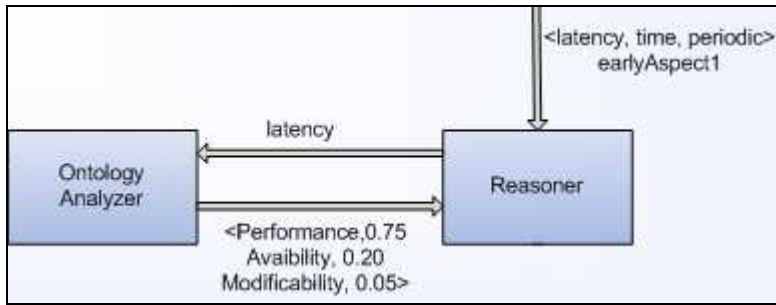
Ejemplo:

Supongamos una lista pequeña, de solo tres términos (latency, time, periodic). La tabla 1 muestra un resumen de lo que obtendríamos luego de recorrer la lista.

	Performance	Availability	Modifiability
Latency	0.75	0.20	0.05
Time	0.6	0.1	0.3
Periodic	0.65	0.3	0.05
TOTAL	2	0.6	0.4
PROMEDIO	0.67	0.2	0.13

Tabla 1

En un principio este módulo tuvo como entrada la lista formada por las palabras latency, time y periodic. Entonces el módulo comienza a recorrer esta lista seleccionando en primer medida la palabra latency. Con este término invoca al módulo *Ontology Analyzer* que responde que *latency* se corresponde con un 0.75 con Performance, 0.20 con Availability y el restante 0.05 con Modifiability

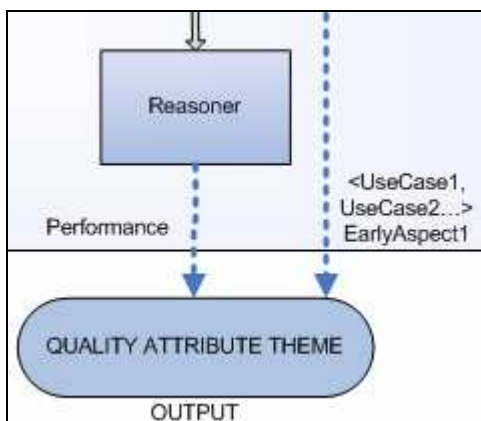


De manera similar se continúa con los siguientes términos, formando el cuadro que se muestra en la Tabla 1.

La fila TOTAL se calcula sumando todos los porcentajes de cada palabra, para cada atributo. Luego este valor se divide por la cantidad de palabras, formando la fila PROMEDIO.

De esta manera, para esta lista de términos, se puede asegurar que hay una posibilidad de 0.67 de que se relacione con Performance, un 0.2 que se relacione con Availability y un 0.13 que lo haga con Modificability.

La salida será, entonces, el quality attribute *Performance* que junto a los casos de uso y el early aspect de entrada formaran el Quality Attribute Theme.



Modulo Ontology Analyzer

El modulo Ontology Analyzer tiene como entrada un término y como salida una lista de pares “atributo, probabilidad”, indicando el grado de relación entre ese término y cada uno de los atributos de calidad. Para ello se vale de una ontología.

Por cada término se calcula que grado de pertenencia tiene éste con los distintos atributos de calidad. Esto se realiza evaluando todos los escenarios particulares con los que hemos "entrenado" a la ontología.

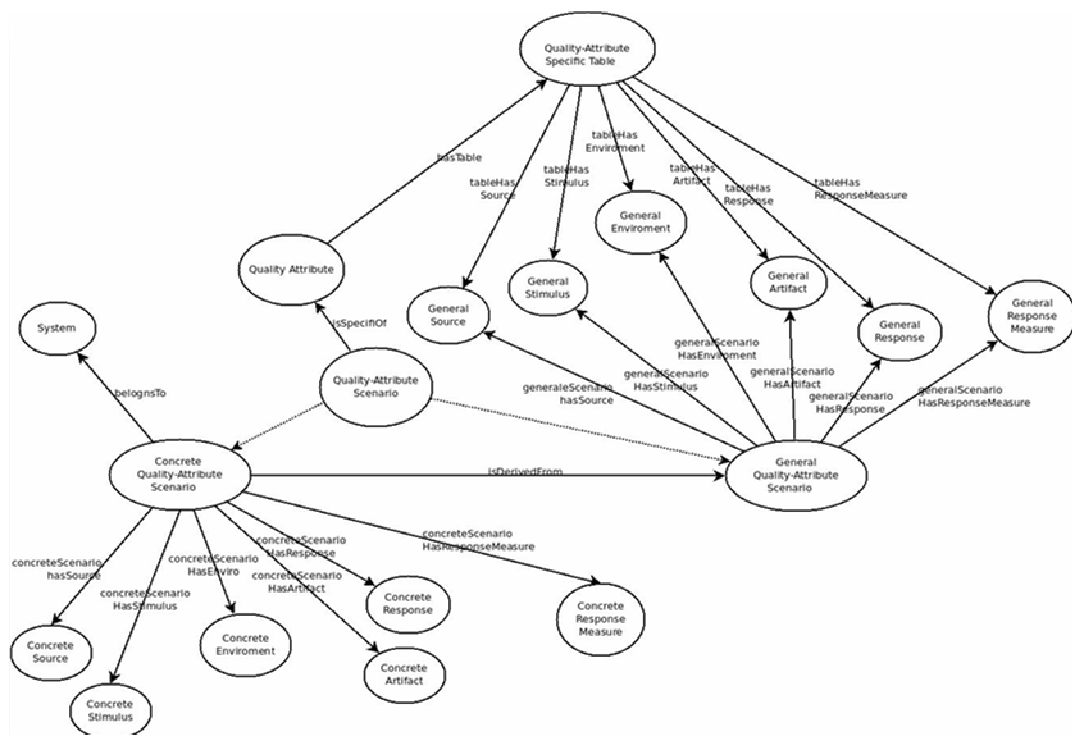
Por ejemplo, supongamos que como entrada tenemos el término "latency". Chequeando la ontología, encontramos que esa palabra aparece en 40 escenarios concretos, en alguna de las partes que lo componen (Stimulus, Environment, Response, etc.).

De esos 40 escenarios concretos o particulares, 30 de ellos se relacionan con el atributo de calidad "Performance", 8 con "Availability" y 2 con "Modifiability". De esta manera podemos concluir que ese concepto, latency, se relaciona en un 75% de certeza con "Performance", 20% con "Availability" y 5% con "Modifiability". En este caso el modulo devolvería <Performance 0.75, Availability 0.2, Modifiability 0.05>.

Ontología

La primera intención para definir la ontología a utilizar fue buscar alguna ya creada para el dominio que nos pueda ser de utilidad. Desafortunadamente no encontramos ninguna existente, por lo que se optó por construir una propia. Ésta se armó principalmente a partir del capítulo 4 del libro "Software Architecture in Practice". De esta manera los conceptos están extraídos de allí, y las relaciones corresponden, en su mayoría, a citas textuales que aparecen en el capítulo.

Acá se presenta la ontología completa que se creó, que correspondería a todo el dominio de análisis. Igualmente la parte que se va a utilizar, como se explicó anteriormente, es solo la de escenarios particulares. Una vez que se vean los resultados obtenidos se irá viendo si, eventualmente, se agrega el resto de la información y también la forma de cómo ello se realizaría.



Conceptos representados:

Quality Attribute: Los atributos de calidad de un sistema son requerimientos no funcionales o características que, en alguna medida, se desea que el sistema posea. Ejemplos de atributos de calidad son: performance, modificability, security, etc.

Concrete Quality-Attribute Scenario: A concrete quality-attribute scenario is a quality-attribute specific requirement. Los escenarios ayudan a describir los atributos de un sistema. Esta formado por seis partes: source of stimulus- stimulus, enviroment, artifact, response y response measure.

General Quality-Attribute Scenario: Escenarios que son independientes del sistema y pueden, eventualmente, pertenecer a cualquiera.

Concrete Quality-Attribute Scenario: Escenarios que son específicos de un sistema bajo consideración.

Quality Attribute-Specific table: Cada atributo de calidad tiene una tabla que brinda todos los valores para cada una de las seis partes de un escenario. Estos valores son independientes de un sistema. Un escenario general es generado eligiendo uno de estos valores para cada parte.

(General|Concrete) Source of Stimulus: Alguna entidad (humano, sistema de computación, etc.) que genera un estímulo.

(General|Concrete) Stimulus: Condición que requiere ser considerada cuando llega al sistema.

(General|Concrete) Enviroment: El estímulo ocurre bajo ciertas condiciones. El sistema puede estar sobrecargado o puede estar en funcionamiento, o cualquier otra condición podría ser verdadera.

(General|Concrete) Artifact: Algún artefacto es estimulado. Esto puede ser el sistema completo o una parte de mismo.

(General|Concrete) Response: La respuesta es la actividad ocurrida después de arribado el estímulo.

(General|Concrete) Response measure: Cuando la respuesta ocurre, ésta debe ser medida de alguna manera, para que los requerimientos puedan ser testeados.

¿Cual es la diferencia entre estos últimos seis conceptos en sus versiones general y concrete? Es decir, ¿cual es la diferencia entre general y concrete response, o entre general y concrete artifact, por ejemplo?

La diferencia es casi la misma que existe entre escenarios generales y escenarios concretos.

Un estímulo general, por ejemplo, es una caracterización de un estímulo que puede pertenecer a cualquier sistema, y es parte de un escenario general. Por el otro lado, un estímulo concreto es parte de un escenario concreto, que pertenece a un sistema en particular.

Por ejemplo, un "internal source" es una fuente de estímulo general, pero un escenario concreto que derive de ese escenario general, no va a tener ese término. En este caso, esa "internal source", sería un administrador de sistema, por ejemplo que sería una instancia del concepto **Concrete Source of Stimulus**.

Explicación de las relaciones

"For each attribute there is a table that gives possible system-independent values for each of the six parts of a general quality-attribute scenario. This is generated by choosing one value for each element."

Esto explica la relación **hasTable** entre **Quality Attribute** y **Quality-Attribute Specific Table**.

Cada tabla brinda los posibles valores para cada una de las partes de un escenario general. Esto se muestra en las relaciones **tableHasSource**, **tableHasStimulus**, **tableHasEnvironment**, **tableHasArtifact**, **tableHasResponse** y **tableHasResponseMeasure**.

Un escenario general se forma eligiendo uno de estos valores para cada una de las seis partes. Luego las relaciones **generalScenarioHasSource**, **generalScenarioHasStimulus**, **generalScenarioHasEnvironment**, **generalScenarioHasArtifact**, **generalScenarioHasResponse** and **generalScenarioHasResponseMeasure** muestran esto.

"We use quality-attribute-specific tables to create general scenarios and from these derive system-specific scenarios"

Esta es la razón por la cual existe la relación **isDerivedFrom** entre **Concrete Quality-Attribute Scenario** y **General Quality-Attribute Scenario**.

Al igual que un escenario general, cada escenario concreto también está compuesto por seis partes y las relaciones **concreteScenarioHasSource**, **concreteScenarioHasStimulus**, **concreteScenarioHasEnvironment**, **concreteScenarioHasArtifact**, **concreteScenarioHasResponse** y **concreteScenarioHasResponseMeasure** muestran esto.

"To make the general scenarios useful for a particular system, you must make them system specific.

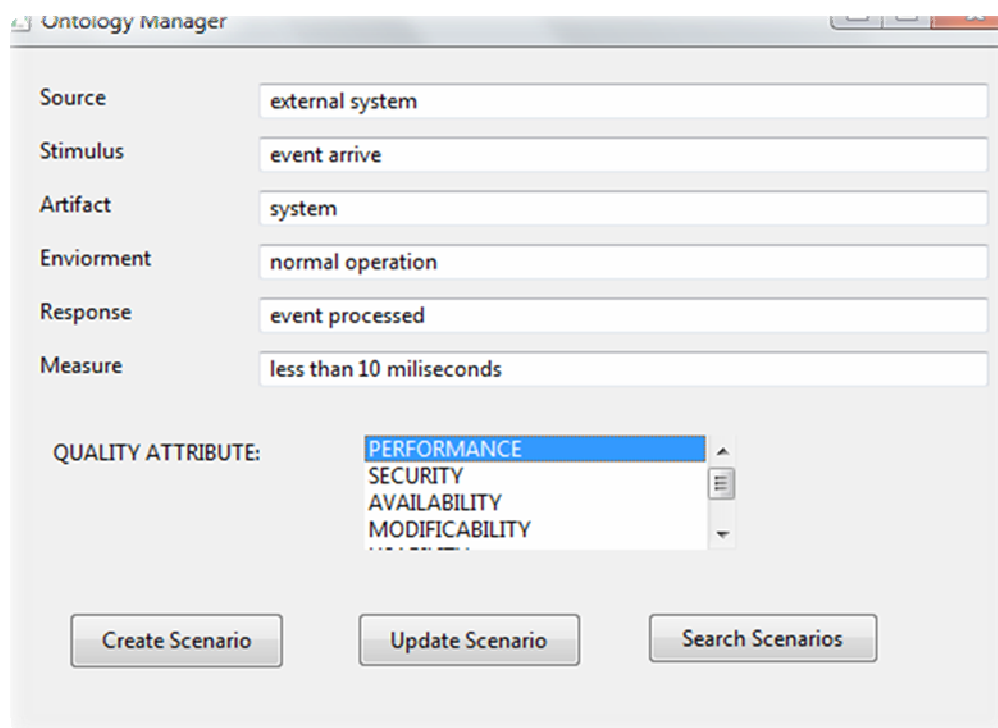
Making a general scenario system specific means translating it into concrete terms for a particular system"

De acá se deriva la relacion **belongsTo** entre **Concrete Quality-Attribute Scenario** y **System**. Los escenarios concretos pertenecen a un sistema en particular.

Entrenamiento de la Ontología

La idea para entrenar la ontología es agregarle escenarios particulares, indicando a que atributo de calidad pertenece y las partes que lo componen, así de esta manera se pueden crear las instancias correspondientes en la ontología.

Esto se realiza mediante el módulo OntologyManager, que es básicamente un ABM de instancias, dentro de la ontología.



The screenshot shows the 'Ontology Manager' window with the following configuration:

Field	Value
Source	external system
Stimulus	event arrive
Artifact	system
Environment	normal operation
Response	event processed
Measure	less than 10 miliseconds

Below the fields, there is a 'QUALITY ATTRIBUTE:' label and a dropdown menu. The dropdown is open, showing the following options:

- PERFORMANCE (highlighted)
- SECURITY
- AVAILABILITY
- MODIFICABILITY

At the bottom of the window, there are three buttons: 'Create Scenario', 'Update Scenario', and 'Search Scenarios'.

Métricas

Para medir el funcionamiento de la aplicación, se ingresarán conjuntos de Casos de Usos relacionados por Early Aspects Candidatos, conociendo de antemano que Quality Attribute los relaciona, es decir, con que QA formarán el QAT². Se realizará el procesamiento de estos datos por la aplicación y una vez obtenido el resultado se podrá calcular la tasa de efectividad a partir de la siguiente fórmula:

$$\text{Tasa de efectividad} = \# \text{ QAT correctos} / \# \text{ total QAT}$$

Eventualmente al avanzar con el trabajo se definirán nuevas métricas en caso de ser necesario (precisión, recall, specificity, etc).

² QAT: Quality Attribute Theme