Workshop: Estrategias para Mejorar la Separación de Concerns

Cursada 2012

El objetivo de este workshop es repasar los conceptos acerca de la problemática de crosscutting concerns en etapas tempranas del desarrollo y aplicar dichos conocimientos sobre la especificación de un sistema real. Para llevar a cabo este trabajo, se contara con la ayuda de una herramienta automatizada que facilita la búsqueda y especificación de los crosscutting concerns pertinentes al sistema. Las tareas a realizar en este workshop se pueden dividir en tres partes. La primer parte consiste en la definición de un conjunto de "consultas" (símil a SQL) para poder encontrar los concerns. La segunda parte consiste en la ejecución de un motor de búsqueda que use dichas consultas para automatizar la detección de concerns. Posteriormente, se deberán analizar los resultados, descartando concerns irrelevantes o errores en la especificación, para así luego especificar detalladamente el significado de cada crosscutting concern en el contexto del sistema. Por ultimo, en la tercera parte se deberá analizar cada crosscutting concern en particular, de forma tal determinar y comprender las relaciones de dichos concerns con cada requerimiento funcional al cual afectan. Para ello, se seguirán las practicas definidas en el proceso de AORE explicado en la materia y se aprovechara el soporte de REAssistant para registrar dichas especificaciones.

Para realizar la primera parte del trabajo, se les brindará una breve introducción a REAssistant, que es la herramienta que utilizaran para efectuar las consultas y detectar los crosscutting concerns en las especificaciones de requerimientos del sistema asignado. Adicionalmente, se dictará una explicación detallada de las anotaciones generadas por el tool y de la sintaxis del lenguaje de consultas, de forma tal estén preparados para escribir consultas efectivas y completas. La cátedra estará disponible para responder consultas durante este proceso. Esta tarea debiera llevar como máximo una hora.

El entregable de esta primera parte es consecuentemente el conjunto de reglas para reconocer crosscutting concerns, y los queries asociados a cada una de las reglas.

Para realizar la segunda parte del workshop, utilizaran el queryset desarrollado anteriormente para ejecutar el motor de consultas de REAssistant sobre el caso de estudio asignado. Con la ayuda de esta herramienta, serán capaces de localizar posibles crosscutting concerns en las especificaciones. Debido a que es posible que la herramienta cometa errores en este proceso, será necesario validar el sentido de dichos concerns de forma tal descartar sugerencias equivocadas.

Los entregables de esta etapa son:

1. Crosscutting concerns candidatos detectados por REAssistant, indicando las sentencia/s involucradas en cada uno de ellos (archivo REA sin intervención de los analistas funcionales)

2. Análisis de crosscutting concerns candidatos, detectando aquellos candidatos que son falsos positivos y aquellos que son verdaderos positivos.
3. Lista revisada y filtrada de crosscutting concerns, incluyendo una descripción por cada uno de ellos que clarifique su comportamiento (archivo REA con la correspondiente intervención).

La tercera y ultima parte del trabajo consiste en realizar las actividades del proceso de Ingeniería de Requerimientos orientada a Aspectos (AORE) presentados durante la cursada de la materia. Los entregables de esta segunda etapa son:

1. Tabla de composición de requerimientos (RCT) (Provisto por el tool).
2. Identificación de los join points en las especificaciones (Provisto por el tool).
3. Determinar el tipo de composición de cada EA con otros comportamientos (Determinado por los analistas).
4. Especificar la realización de cada aspecto temprano por cada join point. En el caso de tener realizaciones similares, evitar repeticiones. Asimismo, si considera que la realización es poco informativa, no es necesario detallarla (Determinado por el analista).

Los entregables de la tercera parte formaran una nueva versión del modelo generado por REAssistant, en donde se tiene los crosscutting concerns detectados y sus influencias sobre otros requerimientos.

Paralelamente a la realización del trabajo, discutir y documentar las implicancias de cada crosscutting concern sobre las decisiones arquitectónicas a tomar en las siguientes etapas del desarrollo, la relación entre cada concern y atributos de calidad general, y el reconocimiento de potenciales riesgos del proyecto.

Anotaciones:

Annotation: Entidad abstracta dentro de los modelos que representa meta-información dentro de un texto. Tiene dos propiedades que delimitan la extensión de dicha información, llamadas ***begin*** y ***end***. Todas las anotaciones a continuación son especializaciones de **Annotation**.

- ***begin***: valor entero que registra el offset de inicio de la anotación (ej., 15).
- ***end***: valor entero que registra el offset de fin de la anotación (ej., 25).

Sentence: Representa una oración detectada en las especificaciones de casos de uso. No contiene propiedades excepto de las heredadas de **Annotation** (***begin*** y ***end***).

Token: Marca y delimita la extensión de una palabra en una oración.

- ***POS***: string que representa la función sintáctica de la palabra en la oración
  (ej., VBZ por verbo en 3ra persona en singular presente o NNS para sustantivo en plural[1])
- ***stem***: string que representa la raíz truncada de la palabra en cuestión
  (por ejemplo: operate, operating, operates, operation, operative, operatives, operational -> oper)
- ***lemma***: string que representa la raíz morfológica de la palabra (sin truncar)
  (por ejemplo: operating, operates, operation, operative, operatives, operational -> operate; am, is, are -> be)
- ***stopword***: valor booleano que indica que la palabra es relevante.

Argument: Es un componente sintáctico que representa una operando de una oración. Se organizan alrededor de un **Predicate**. Un ejemplo es un sujeto, o un objeto indirecto de una oración.

- ***label***: representa el tipo de argumento de un predicado. Ejemplos debajo:

A0	subject
A1	object
A2	indirect object
Adjuntos	
AM-ADV	adverbial modification
AM-DIR	direction
AM-DIS	discourse marker
AM-EXT	extent
AM-LOC	location
AM-MNR	manner
AM-MOD	general modification

[1] Mirar http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html para mas detalle.

AM-NEG negation
AM-PRD secondary predicate
AM-PRP purpose
AM-REC recipricol
AM-TMP temporal
Other Labels
C-arg continuity of an argument/adjunct of type arg
R-arg reference to an actual argument/adjunct of type arg

- ***description***: representa el significado semántico del argumento en relación al predicado del cual forma parte.
  (por ejemplo, la entidad calculada o la forma en que se realiza una operación)

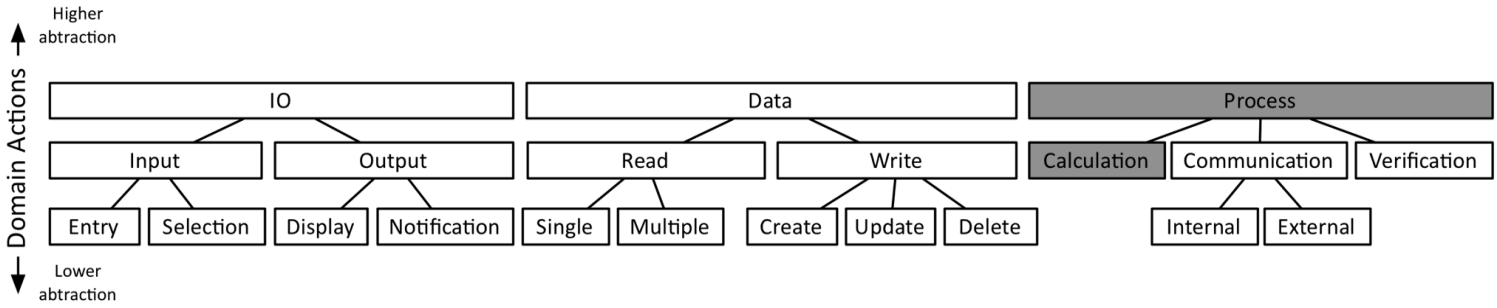
Predicate: Representa un verbo principal en una oración. Esta relacionado directamente con sus **Argument**.

- ***label***: registra el identificador único de un verbo, el cual registra su sentido en la oración. (ej., validate.01 o create.03)
- ***desc***: texto que indica el significado del verbo en cuestión (ej., to give something away)
- ***passiveVoice***: valor booleano que indica si esta en voz pasiva o activa.

DomainAction: Es una interpretación del significado de un predicado en el contexto de los casos de uso. Los domain actions son organizados en una jerarquía estricta, que va desde un nivel de abstracción bajo a uno alto. Los de bajo nivel son acciones abstractas mientras que las de alto nivel representan categorías de acciones.

- ***label***: representa el tipo de **DomainAction** asociado a un predicado; por ejemplo, Entry (por algún ingreso de datos) o Selection (por alguna selección de opciones), Input (en engloba las dos DA anteriores y se refiere a alguna operación de entrada), o IO (la cual representa cualquier comportamiento que englobe la entrada/salida de información del sistema).

La jerarquía tiene tres niveles, y se puede visualizar en el siguiente grafico:



La sintaxis de EMF/Query2 es bastante intuitiva. Considere el siguiente ejemplo para el concern de Persistencia como guía:

Query1 (para pistas explicitas):

select S — *Seleccionar los sentencias S*

from [#Sentence#] as S, [#Token#] as T — *De las anotaciones de tipo Sentence como S y de tipo Token como T*

where for T(lemma = 'database' or lemma = 'storage' or stem = 'persistent' or lemma = 'rolled-back' or lemma='table' or lemma='populate') — *Donde alguna de las siguientes propiedades de T sean coincidentes con un valor dado*

where T.begin > S.begin

where T.end < S.end — *Donde el Token T este dentro del Sentence S*

Query2 (para interpretaciones más sutiles):

select S — *Seleccionar los sentencias S*

from [#Sentence#] as S, [#DomainAction#] as DA — *De las anotaciones de tipo Sentence como S y de tipo DomainAction como DA*

where for DA(label = 'Data — *Donde la propiedad label de DA tenga el valor "Data"*

where DA.begin > S.begin

where DA.end < S.end — *Donde la DomainAction DA este dentro de la Sentence S*