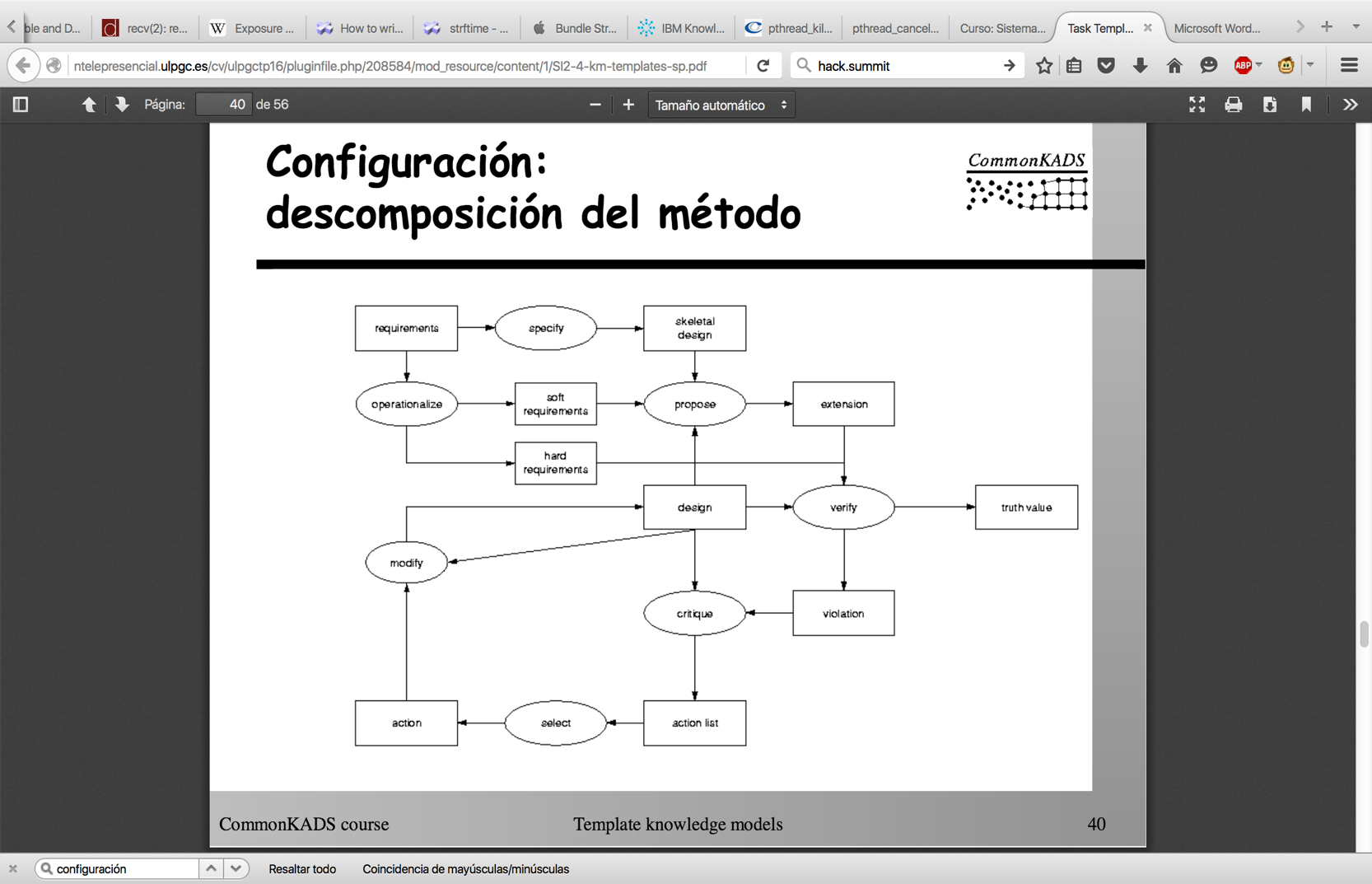
**Grupo: Laura del Pino Díaz y Aitor del Pino Saavedra Hernández**

1. **Identificar y justificar el tipo genérico de tarea según la clasificación de CommonKAD.**

Es una tarea sintética del tipo configuración porque no tiene restricciones temporales.

1. **Describir el control de la tarea, roles e inferencias para la tarea seleccionada.**



Se plantea una modificación del esquema superior de forma que solo haya una única acción en la lista de acciones que sea deshacer y que la inferencia “modify” siga un esquema de búsqueda en el espacio de estados.

Los requerimientos son:

Máximo de dos horas de teoría por día. (Pasará a ser límite duro).

Las asignaturas trocales son las primeras se dispondrán consecutivamente cada día.(Límite duro)

Se prefiere que las asignaturas troncales se den lo más temprano posible. (Preferencia, límite suave)

Todas las clases de laboratorioson de dos horas (Pasará a ser límite duro).

Solo existen tres laboratorios. (Limite furo)

Solo se pueden simultanear las horas de prácticas.

Un profesor debe dar solo un grupo de teoría.

Se prefiere que se pongan las horas de teoría antes que las de práctica.

El diseño esquelético:

Es el horario con sus franjas horarias.

Las extensiones son:

Cada una de las versiones del horario con un conjunto de franjas ocupadas.

Las violaciones son las extensiones que no cumplan las restricciones.

El valor de verdad es una extensión que cumple las restricciones.

Las inferencias de esta tarea son:

“specify”/especificar: a partir de los requerimientos se genera el diseño esquelético.

“operationalize”:dividir los requerimientos en las clases suave y duro.

“propose”: genera una extensión a partir de los requerimientos suaves y el diseño esquelético.

“verify”:verificar que se cumplen los límites duros y genera un valor de verdad o un violación de restricciones.

“modify”: deshace la última extensión y prueba con otro valor.

El control de esta tarea es:

operationalize(requirements -> hard-reqs + soft-reqs);

specify(requirements -> skeletal-design);

while new-solution propose(skeletal-design + design +soft-reqs

-> extension) do

design := extension union design;

verify(design + hard-reqs -> truth-value or violation);

while violation != NULL then

modify(design + action -> design);

verify(design + hard-reqs -> truth-value or violation);

end while;

end while