ONTOSIS: REPRESENTACIÓN ONTOLÓGICA DEL CONOCIMIENTO EN LOS CONTENIDOS PROGRAMÁTICOS DEL PROGRAMA DE INGENIERÍA DE SISTEMAS

MANUAL TÉCNICO

Autor(es):
Wendy Estefania Galindo Merchán
C.C 1001187365

Nicolás Steven Bermúdez Hernández
C.C 1000970794

UNIVERSIDAD EL BOSQUE
PROGRAMA DE INGENIERÍA DE SISTEMAS
FACULTAD DE INGENIERÍA
Bogotá, 2023

Contenido

1. Introducción

Este documento presenta una investigación sobre la problemática de la representación y actualización manual de los contenidos programáticos en la Universidad El Bosque, con un enfoque en las dimensiones biopsicosociales y culturales que influyen en este proceso. Se exploran antecedentes relevantes, se identifican áreas de mejora y se proponen soluciones para abordar los desafíos identificados. Además, se profundiza en la importancia de comprender cómo estas dimensiones interrelacionadas afectan la eficiencia y efectividad del diseño curricular en el contexto específico de la universidad. La estructura del documento de los contenidos programáticos incluye una introducción, antecedentes, metodología, resultados, discusión y conclusiones, seguidas de referencias bibliográficas.

A través de la exploración de antecedentes relevantes, se busca contextualizar la problemática y destacar la importancia de abordarla de manera integral, considerando tanto aspectos técnicos como sociales y culturales. Hemos identificado áreas de mejora específicas en el proceso de representación y actualización de los contenidos programáticos identificando las limitaciones y desafíos que la universidad conlleva en este sentido.

2. Especificación Técnica

El sistema de recomendación híbrido basado en ontologías consta de un código realizado en el lenguaje de Python y de OWL para el manejo de las ontologías. La integración con el sistema a través de un sheet de Google

Lenguaje de programación	Python 3.12.0

Librerías principales	Pandas Matplotlib Sklearn Gspread Google Networkx Owlready2 Os

Tabla 1. Características técnicas del software

3. Requerimientos de Hardware

A continuación, se detallan los requerimientos mínimos de hardware para poder cargar, depurar, compilar y probar el proyecto de código fuente anexo a este documento.

Esencialmente, se requiere contar con un computador de escritorio o portátil con las siguientes características.

- Arquitectura de CPU: x86_64; 2da generación (o superior) de Intel Core o AMD CPU con soporte para Windows Hypervisor.
- 4 GB de memoria RAM o más.
- 6 GB de espacio libre en disco duro
- Resolución mínima de 1280 x 800.

4. Requerimientos de Software

El proyecto puede ser cargado desde diferentes sistemas operativos, tales como Windows®, Mac® y Linux®. A continuación, se relacionan las características adecuadas en cada caso:

- **Windows:** 64-bit Microsoft® Windows® 8/10
- **Mac:** MacOS® 10.14 (Mojave) or higher
- **Linux:** Any 64-bit Linux distribution that supports Gnome, KDE, or Unity DE; GNU C Library (glibc) 2.31 or later.

Para abrir y compilar el proyecto, se requiere:

- Visual Studio Code
- Python instalado
- Navegador web compatible con los servicios de Google
- Cuenta Gmail
- Protégé

5. Partes del Sistema

El presente apartado será presentado en……

Creación de la ontología

Inicialmente, se realizó la creación de la ontología que está pensada para la verificación de las actividades y resultados de aprendizaje de los sílabos por cada dimensión de aprendizaje

CLASES
syllabus, bibliography, general_contents, subject, schedule, learning_dimensions, conclusion, student, syllabus_evaluation, teacher

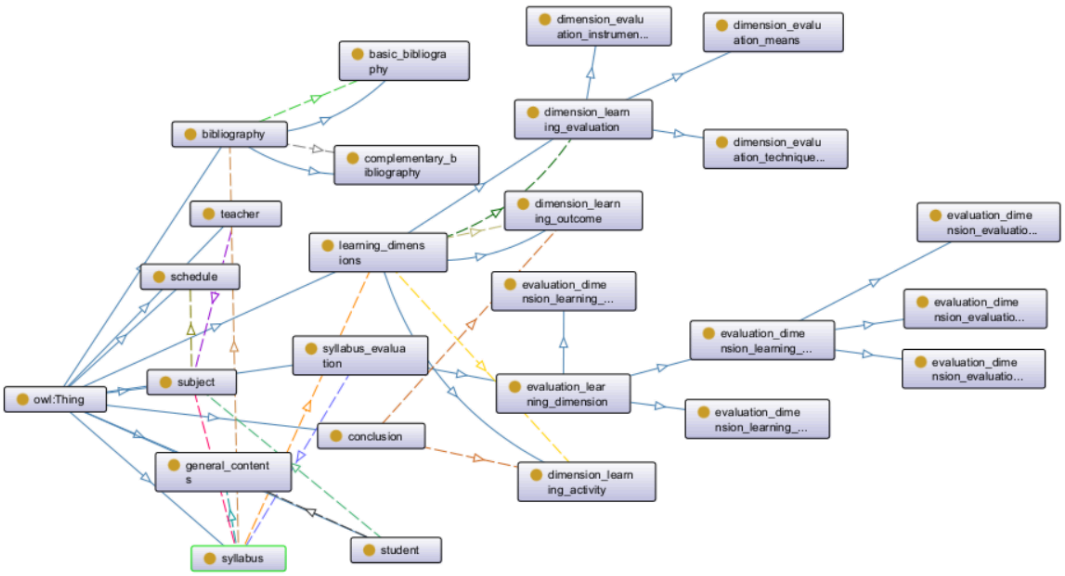
Tabla 2. Clases ontológicas

SUBCLASES

basic_bibliograhpy, complementary_bibliography, dimension_learning_activity, dimension_learning_outcome_dimension_learningevaluation, dimension_evaluation_instrument, dimension_evaluation_means, dimension_evaluation_techniques

Tabla 3. Subclases ontológicas

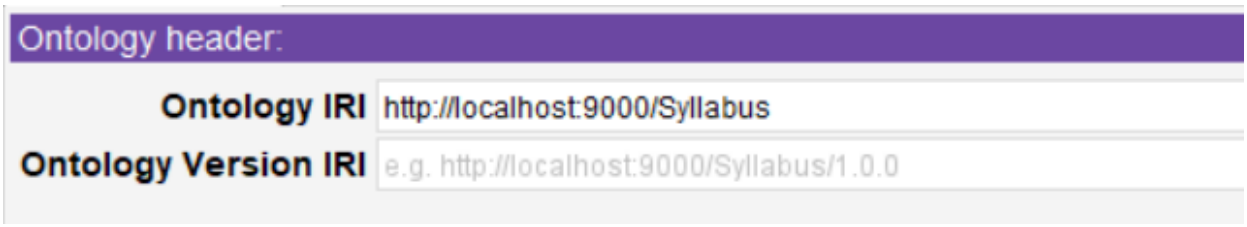
Se crean toda la ontología con su respectivas características.



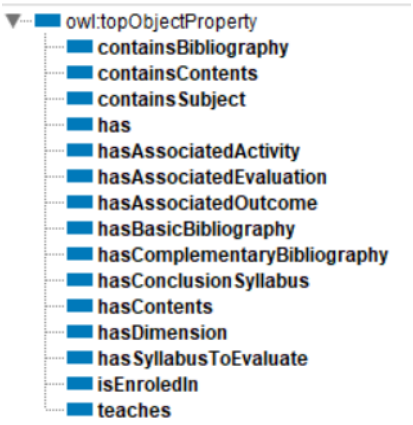
La ontología se puede subir a un servidor para poder relacionar entre sí. En este caso se subieron al puerto 9000 de LocalHost

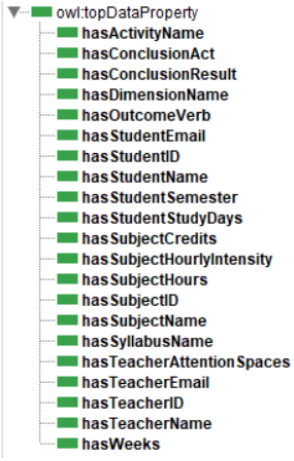
```
C:\Users\estef\Desktop\Ontosis>python -m http.server 9000
Serving HTTP on :: port 9000 (http://[::]:9000/) ...
```

Es importante definir el IRI (identificador único) que tendrá cada ontología para posteriormente realizar las reglas en Python



Ya una vez creada las ontología se procede a realizar todas las relaciones (Object Properties) y atributos (DataProperties)





Creación de razonador

Primeramente, se creó una tabla con datos de los sílabos y a su vez los datos de los resultados y actividades de aprendizaje por dimensión

Materia	Facultad	Programa	Nombre Asignatura	Semestre	Periodo Academico	Código Asignatura
Proyecto de grado 1	FACULTAD DE INGENIERIA	ING. SISTEMAS DIURNA	PROYECTO DE GRADO 1	8	20231	13013

dimension	Resultado 1	Resultado 2	Resultado 3	Resultado 4	Resultado 5	Resultado 6	Resultado 7	Resultado 8	Resultado 9
APRENDERAAPRENDER	CREARPLAN	RECOPILAR	INDAGAR	REFLEXIONAR	AUTOEVALUAR	AUTOREGULAR	DESARROLARPLAN		
COMPROMISO	ENTUSIASMARSE	PREPARARSE	INTERESARSE	VALORAR	COMPROMETERSE	DECIDIRSE	ACORDAR		
DIMENSIONHUMANA	VERSEASIMISMO	RELACIONARSE	EMPATIZAR	DECIDIRCONVERTIRSEEN	LIDERAR	RESPETAR	CONCILIAR	NEGOCIAR	AYUDAR
CONOCIMIENTOFUNDAMENTA	CLASIFICAR	COMPRENDER	DEFINIR	DESCRIBIR	ELEGIR	ENUMERAR	IDENTIFICAR	LISTAR	NOMBRAR
APLICACION	ADAPTAR	ANALIZAR	ARGUMENTAR	CALCULAR	COORDINAR	CREAR	CONFIGURAR	DECIDIR	DESARROLLAR
INTEGRACION	CONECTAR	IDENTIFICARLAINTERACCIONENTRE	IDENTIFICARSIMILITUDESENTRE	RELACIONAR	DIFERENCIAR	COMPARAR	INTEGRAR	ASOCIAR	COMBINAR

dimension	Actividad 1	Actividad 2	Actividad 3	Actividad 4	Actividad 5	Actividad 6
APRENDERAAPRENDER	EVALUACIONDELAPRENDIZAJE	REFLEXIONESPERSONALES	PORTAFOLIODEAPRENDIZAJE	DESEMPEÑODEAPRENDIZAJEBASADOENPROBLEMAS	RUBRICAS	LISTADODEMETAS
COMPROMISO	REFLEXIONESPERSONALES	CUESTIONARIOSESTANDARIZADOS	PORTAFOLIODEAPRENDIZAJE	GRUPOSDEDISCUSION	ANALISISDESITUACIONES	DEBATES
DIMENSIONHUMANA	REFLEXIONESPERSONALES	CUESTIONARIOSESTANDARIZADOS	PORTAFOLIODEAPRENDIZAJE	DISCUSIONESGRUPALES	JUEGOSDEROLES	DILEMASETICOS
CONOCIMIENTOFUNDAMENTAL	EXAMEN	EJECICIOSPREGUNTASORALES	MAPACONSEPTUAL	RESUMENES	QUIZ	CUESTIONARIOS
APLICACION	SIMULACIONES	DEMOSTRACIONES	PROYECTOSENEQUIPO	ESTUDIOSDECASO	ACTIVIDADESPARAEXPLICACIONES	ESCRITURA
INTEGRACION	ESCRITURAREFLEXIVA	CASOSPROGRESIVOSINCOMPLETOS	MAPASCONSEPTUALES	APRENDIZAJEBASADOENPROBLEMAS	PROBLEMASPRACTICOSDOCUMENTADOSENAMBIENTESREALES	TRABAJOSCONEEJEMPLODELAVIDAREAL

Finalmente se hace un análisis de los sílabos y se categorizan según las tablas

anteriormente mencionadas

Resultados:

MATERIA	APRENDERAAPRENDER	COMPROMISO	DIMENSIONHUMANA	CONOCIMIENTOFUNDAMENTAL	APLICACION	INTEGRACION
PG1_PROYECTODEGRADO1	CREARPLAN	COMPROMETERSE	RELACIONARSE	COMPRENDER	DESARROLLAR	INTEGRAR
PG2_PROYECTODEGRADO2	CREARPLAN	COMPROMETERSE	RELACIONARSE	COMPRENDER	DESARROLLAR	INTEGRAR
LS_LABORSOCIAL	INDAGAR	INTERESARSE	DEMOSTRAR	PLANIFICAR	DESARROLLAR	INTEGRAR
TDP_TALLERDEPRACTICAPROFESIONAL	CREARPLAN	ESTARLISTOPARA	VERSEASIMISMO	RECONOCER	CREAR	CONECTAR
PP_PRACTICAPROFECIONAL	PROFUNDIZAR	PROMOVER	DEMOSTRAR	COMPRENDER	RESOLVERPROBLEMASCRITICAMENTE	INTEGRAR
FDP_FUNDAMENTOSDEPROGRAMACION	DESARROLARPLAN	ACORDAR	VERSEASIMISMO	COMPRENDER	PROGRAMAR	INTEGRAR
P1_PROGRAMACION1	FORMULAR	PLANTEAR	RESOLVER	EXTENDER	DESARROLLAR	INTEGRAR
P2_PROGRAMACION2	RECOPILAR	VALORAR	RELACIONARSE	COMPRENDER	DESARROLLAR	RELACIONAR
BD1_BASEDEDATOS1	INDAGAR	INTERESARSE	AYUDAR	CONSTRUIRACTUALIZARYMANTENER	ELABORAR	CONTRASTAR
BD2_BASEDEDATOS2	RECOPILAR	ACORDAR	RELACIONARSE	COMPRENDER	PROGRAMAR	INTEGRAR
IS1_INGENIERIADESOFTWARE1	INDAGAR	PLANTEAR	DECIDIRCONVERTIRSEEN	COMPRENDER	DESARROLLAR	INTEGRAR
IS2_INGENIERIADESOFTWARE2	INDAGAR	VALORAR	RELACIONARSE	COMPRENDER	ADAPTAR	RELACIONAR
IS3_INGENIERIADESOFTWARE3	ACTUALIZAR	PLANTEAR	CONCILIAR	COMPRENDER	GESTIONAR	INTEGRAR
PN1_PROYECTONUCLEO1	RECOPILAR	ACORDAR	RELACIONARSE	IDENTIFICAR	UTILIZAR	RELACIONAR
PN2_PROYECTONUCLEO2	REFLEXIONAR	INTERESARSE	RELACIONARSE	DESCRIBIR	UTILIZAR	CONSTRUIR
SI_SISTEMASINTELIGENTES	RECOPILAR	INTERESARSE	RELACIONARSE	RECONOCER	DESARROLLAR	CONECTAR

Actividades:

MATERIA	APRENDERAAPRENDER	APRENDERAAPRENDER	APRENDERAAPRENDER	APRENDERAAPRENDER	APRENDERAAPRENDER	APRENDERAAPRENDER	COMPROMISO	COMPROMISO	COMPROMISO	COMPROMISO	COMPROMISO
PG1_PROYECTODEGRADO1	DIARIO	REFLEXIONESPERSONALES	CHARLASEXTERNAS	BITACORA			REFLEXIONESPERSONALES	BITACORA	ANALISISDESITUACIONES		
PG2_PROYECTODEGRADO2	REFLEXIONESPERSONALES	CHARLASEXTERNAS	BITACORA	PROYECTODEEQUIPO			REFLEXIONESPERSONALES	GRUPOSDEDISCUSION	BITACORA	PROYECTODEEQUIPO	
LS_LABORSOCIAL	REFLEXIONESPERSONALES	BITACORA					PLANDETRABAJO				
TDP_TALLERDEPRACTICAPROFESIONAL	CLASEMAGISTRAL	REFLEXIONESPERSONALES	RUBRICA				CUESTIONARIOSESTANDARIZADOS	REFLEXIONESPERSONALES	CLASEMAGISTRAL		
PP_PRACTICAPROFECIONAL	REFLEXIONESPERSONALES	PORTAFOLIODEAPRENDIZAJE	DIARIO	LISTADODEDOMINIOS	AMBIENTESDEAPRENDIZAJEPERSONAL		PORTAFOLIODEAPRENDIZAJE				
FDP_FUNDAMENTOSDEPROGRAMACION	LISTADODEMETAS	PROYECTO					GRUPOSDEDISCUSION	ANALISISDESITUACIONES	CASOSDEESTUDIO	PROYECTODEEQUIPO	
P1_PROGRAMACION1	INVESTIGACION	ACTIVIDADESPRACTICAS	EVALUACIONDELAPRENDIZAJE				GRUPOSDEDISCUSION	REALIMENTACION			
P2_PROGRAMACION2	DESEMPEÑODEAPRENDIZAJEBASADOENPROBLEMAS	LISTADODEMETAS					ANALISISDESITUACIONES	CASOSDEESTUDIO			
BD1_BASEDEDATOS1	TALLERESGRUPALES	PROYECTO					TALLERESGRUPALES	PORTAFOLIODEAPRENDIZAJE	ANALISISDESITUACIONES		
BD2_BASEDEDATOS2	CLASEMAGISTRAL	EXPOSICIONES	ACTIVIDADESPRACTICAS				CLASEMAGISTRAL	EXPOSICIONES	ACTIVIDADESPRACTICAS		
IS1_INGENIERIADESOFTWARE1	DESEMPEÑODEAPRENDIZAJEBASADOENPROBLEMAS	MODELAMIENTO					TALLERES	ANALISISDESITUACIONES			
IS2_INGENIERIADESOFTWARE2	DEBATES	LISTADODEMETAS	ACTIVIDADESGRUPALES				GRUPOSDEDISCUSION	DEBATES	CASOSDEESTUDIO	ACTIVIDADESGRUPALES	
IS3_INGENIERIADESOFTWARE3	DESEMPEÑODEAPRENDIZAJEBASADOENPROBLEMAS						PRESENTACIONESDELOSTEMAS	SESIONESTEORICASDELOSTEMAS	RETROALIMENTACIONES		
PN1_PROYECTONUCLEO1	CLASEMAGISTRAL	AMBIENTESDEAPRENDIZAJEPERSONAL					CLASEMAGISTRAL				
PN2_PROYECTONUCLEO2	CLASEMAGISTRAL	DESEMPEÑODEAPRENDIZAJEBASADOENPROBLEMAS	ACTIVIDADES	PROYECTO	AMBIENTESDEAPRENDIZAJEPERSONAL	CHARLASEXTERNAS	CLASEMAGISTRAL	TALLERESDIRIGIDOS	PORTAFOLIODEAPRENDIZAJE	BUSQUEDADEINFORMACION	TUTORIAS
SI_SISTEMASINTELIGENTES	AMBIENTESDEAPRENDIZAJEPERSONAL						TALLER	PROYECTODEEQUIPO			

Se crean los individuos primeramente llamando la ontología en el código por medio de

```
onto = get_ontology("http://localhost:9000/Syllabus.owx")


onto.load()
```

Luego se carga la información mostrada en las tablas

```
file_path = "https://docs.google.com/spreadsheets/d/e/2PACX-1vS-IXRrHrgGskrXGxwSyBJJ5-h-M-YHIiPdmtEz_UYnHIk1KhqW-lJDZAOxTAsgsTwngrbgXix3RNmJ/pub?output=xlsx"
resultados = pd.read_excel(file_path, sheet_name='Resultados')
actividades = pd.read_excel(file_path, sheet_name='Actividades')
categorizadaResult = pd.read_excel(file_path, sheet_name='Tabla categorizada Res. por dim')
categorizadaAct = pd.read_excel(file_path, sheet_name='Tabla categorizada Act. por dim')
```

y se crean los métodos para la creación de individuos

```python
with onto:
    for index, row in resultados.iterrows():

        # Creacion de las dimensiones en evaluacion y los resultados de aprendizaje en evaluacion
        for column_name, value in row.items():
            if not pd.isnull(value):  # Verificar si el valor de la celda no está vacío
                if column_name == 'dimension':
                    value = "E_" + value
                    dimE = onto.evaluation_learning_dimension(value)
                    dimE.hasDimensionName = [value]
                else:
                    div2 = value.split()[0]
                    resultE = onto.evaluation_dimension_learning_outcome(div2)
                    resultE.hasOutcomeVerb = [div2]
                    dimE.hasAssociatedOutcome.append(resultE)

    for index, row in actividades.iterrows():

        # Creación de actividades y asociación con las dimensiones existentes en evaluación
        for column_name, value in row.items():
            if not pd.isnull(value):  # Verificar si el valor de la celda no está vacío
                if column_name == 'dimension':
                    # Buscar la dimensión existente por su nombre
                    value = "E_" + value
                    dim = onto.search_one(hasDimensionName=[value], is_a=onto.evaluation_learning_dimension)
                else:
                    div2 = value.split()[0]
                    act = onto.evaluation_dimension_learning_activity(div2)
                    act.hasActivityName = [div2]
                    dim.hasAssociatedActivity.append(act)
```

```
for index, row in categorizadaResult.iterrows():

    for column_name in categorizadaResult.columns:
        value = row[column_name]
        if not pd.isnull(value):  # Check if the value of the cell is not empty
            if column_name == 'MATERIA':
                div = value.split("_")
                pref = div[0]
                name = div[1]
                subj = onto.subject(name)
                aux = "S_" + name
                syll = onto.syllabus(aux)
                subj.hasSubjectName = [name]
                syll.hasSyllabusName = [aux]
            elif column_name in ['APRENDERAAPRENDER', 'COMPROMISO', 'CONOCIMIENTOFUNDAMENTAL', 'APLICACION',
                                 'INTEGRACION', 'DIMENSIONHUMANA']:
                aux2 = pref + "_" + column_name
                aux2 = aux2.split(".")[0]
                aux3 = "C_" + aux2
                aux3 = aux3.split(".")[0]
                con = onto.conclusion(aux3)
                dimS = onto.learning_dimensions(aux2)
                dimS.hasDimensionName = [aux2]
                result = onto.dimension_learning_outcome(value)
                result.hasOutcomeVerb = [value]
                dimS.hasAssociatedOutcome.append(result)
                syll.hasDimension.append(dim)
                con.hasConclusionSyllabus.append(result)
```

```python
for index, row in categorizadaAct.iterrows():
    for column_name, value in row.items():
        if not pd.isnull(value):  # Verificar si el valor de la celda no está vacío
            if column_name == 'MATERIA':
                div = value.split("_")
                pref = div[0]
                name = div[1]
                subj = onto.search_one(hasSubjectName=[name], is_a=onto.subject)
                aux = "S_" + name
                syll = onto.search_one(hasSyllabusName=[aux], is_a=onto.syllabus)
            else:

                aux2 = pref + "_" + column_name
                aux2 = aux2.split(".")[0]
                aux3 = "C_" + aux2

                dimA = onto.search_one(hasDimensionName=[aux2], is_a=onto.learning_dimensions)
                if dimA is not None:
                    con = onto.conclusion(aux3)
                    # Crear la actividad
                    div2 = value.split()[0]
                    act = onto.dimension_learning_activity(div2)
                    act.hasActivityName = [div2]
                    # Agregar la actividad a la dimensión
                    dimA.hasAssociatedActivity.append(act)
                    con.hasConclusionSyllabus.append(act)
```

Con la ayuda del lenguaje swrl se crearon reglas de inferencia que dicen si las actividades o resultados por silabo y dimensión son consistentes o inconsistentes.

```python
rules = [
    """
    evaluation_learning_dimension(?evalDim) ^ evaluation_dimension_learning_activity(?evalAct) ^ hasAssociatedActivity(?evalDim, ?evalAct) ^ learning_dimensions
    """,
    """
    evaluation_learning_dimension(?evalDim) ^ evaluation_dimension_learning_outcome(?evalRest) ^ hasAssociatedActivity(?evalDim, ?evalAct) ^ learning_dimensions
    """,
    """
    evaluation_learning_dimension(?evalDim) ^ evaluation_dimension_learning_outcome(?evalRest) ^ hasAssociatedActivity(?evalDim, ?evalAct) ^ learning_dimensions
    """,
    """
    evaluation_learning_dimension(?evalDim) ^
    evaluation_dimension_learning_activity(?evalAct) ^
    hasAssociatedActivity(?evalDim, ?evalAct) ^
    learning_dimensions(?learnDim) ^
    dimension_learning_activity(?learnAct) ^
    hasAssociatedActivity(?learnDim, ?learnAct) ^
    differentFrom(?evalDim, ?learnDim) ^
    differentFrom(?evalAct, ?learnAct) ^
    hasConclusionSyllabus(?c, ?learnAct) ->
    hasConclusionAct(?c, "INCONSISTENTE")
    """
```

y se pasaron por el razonador pellet por medio del siguiente código

```
with onto:
    for i, rule in enumerate(rules, start=1):
        imp = Imp()
        imp.set_as_rule(rule)

    sync_reasoner_pellet(infer_property_values=True, infer_data_property_values=True)
```

Finalmente se guarda la ontología en el archivo llamado OntosisFinal

```
onto.save("OntosisFinal.owl")
```

Integración con la interfaz

Una vez se tiene el archivo final de la ontología, se procede a crear un programa con la interfaz, primero se tiene que volver a hacer el llamado a la ontología

```
onto = get_ontology("http://localhost:9000/OntosisFinal.owl").load()
```

Luego, por medio de flask se empieza a crear la página la cúal nos va a mostrar la ontología con las reglas aplicadas, se crean 3 métodos de visualización, uno para la ontología general, otro para sus clases y otro para los individuos

```
app = Flask(__name__)
```

```
@app.route('/')
def ontology_page():
    classes = [(Class.name, url_for( endpoint: "class_page", iri=Class.iri)) for Class in Thing.subclasses()]
    return render_template( template_name_or_list: "ontology.html", classes=classes)


@app.route('/class/<path:iri>')
def class_page(iri):
    Class = IRIS[iri]
    subclasses = [(SubClass.name, url_for( endpoint: "class_page", iri=SubClass.iri)) for SubClass in Class.subclasses()]
    individuals = [(individual2.name, url_for( endpoint: "individual_page", iri=individual2.iri)) for individual2 in
                   Class.instances()]
    return render_template( template_name_or_list: "class.html", class_name=Class.name, subclasses=subclasses, individuals=individuals)
```

```python
@app.route('/individual/<path:iri>')
def individual_page(iri):
    individual = IRIS[iri]
    classes = [(cls.name, url_for( endpoint: "class_page", iri=cls.iri)) for cls in individual.is_a]
    object_properties = {}
    data_properties = {}
    individual_name = str(individual).split(".")[1]

    for prop in individual.get_properties():
        values = []
        if isinstance(prop, ObjectPropertyClass):
            values = [obj.name for obj in getattr(individual, prop.name)]
            object_properties[prop.name] = values
        else:
            values = [str(getattr(individual, prop.name))]
            data_properties[prop.name] = values

    return render_template( template_name_or_list: 'individual.html', individual=individual, classes=classes,
                           object_properties=object_properties, data_properties=data_properties,
                           individual_name=individual_name)
```

A la vez se crean 3 archivos .html con lo necesario para la visualización
Ontología:

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ontosis</title>
    <style...>
</head>
<body>
    <div class="container">
        <h2>Ontosis</h2>
        {% for class_name, class_url in classes %}
            <a href="{{ class_url }}" class="class-link">{{ class_name }}</a>
        {% endfor %}
    </div>
</body>
</html>
```

Clases:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{{ class_name }} class</title>
    <style...>
</head>
<body>
    <div class="container">
        <h2>'{{ class_name }}' class</h2>
        <a href="{{ url_for('ontology_page') }}" class="back-link">Volver</a><br><br>
        {% for subclass_name, subclass_url in subclasses %}
            <a href="{{ subclass_url }}" class="class-link">{{ subclass_name }}</a>
        {% endfor %}
        <h3>Individuals</h3>
        {% for individual_name, individual_url in individuals %}
            <a href="{{ individual_url }}" class="individual-link">{{ individual_name }}</a>
        {% endfor %}
    </div>
</body>
</html>
```

Individuos:

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{{ individual_name }} individual</title>
    <style...>
</head>
<body>
    <div class="container">
        <h2>{{ individual_name }} individual</h2>
        <a href="{{ url_for('ontology_page') }}" class="back-link">Volver a la página principal</a><br><br>
        <h3>Classes</h3>
        {% for class_name, class_url in classes %}
            <a href="{{ class_url }}" class="class-link">{{ class_name }}</a>
        {% endfor %}
        <h3>Object Properties</h3>
        {% if object_properties  %}
            {% for object_properties , values in object_properties .items() %}
                <p class="relation">{{ object_properties }}:
                {% for value in values %}
                    {{ value }},
                {% endfor %}
                </p>
            {% endfor %}
        {% else %}
            <p class="relation">No object properties associated with this individual.</p>
        {% endif %}
        <h3>Data Properties</h3>
        {% for data_property, value in data_properties.items() %}
            <p class="relation">{{ data_property }}: {{ value }}</p>
        {% endfor %}
    </div>
```

Finalmente se despliega la página con el siguiente código

```
if __name__ == "__main__":
    app.run(debug=True)
```

6. Control de cambios

Fecha	Descripción	Responsable
23/04/2024	Versión 1 del documento	Wendy Estefania Galindo Merchán