



STORES



Nội dung

- 1. STORES và Kiến trúc hệ thống
- 2. Các chức năng của hệ thống
- 3. Các loại đối tượng trao đổi khi sử dụng các dịch vụ
- 4. Các nhóm dịch vụ
- 5. Bài tập



Ngữ cảnh

■ Hiện trạng:

- Dự án CNTT quy mô ngày càng lớn
 - →Bùng nổ về mã nguồn
- Các IDE thiếu công cụ tìm kiếm mã nguồn hiệu quả
 - →Khó khăn trong tái sử dụng, bảo trì phần mềm

■ Giải pháp:

- Xây dựng công cụ tìm kiếm mã nguồn, tích hợp vào các IDE
- Tìm kiếm trên:
 - Thuần túy quan hệ các thành phần Source code
 - Thông tin bổ sung



Tiếp cận xây dựng

- Hướng tiếp cận:

- Sử dụng ontology mô hình quan hệ các thành phần mã nguồn
- Tìm kiếm trên mô hình ngữ nghĩa

- Ưu điểm:

- Tìm kiếm ngữ nghĩa
 - Đáp ứng các tiêu chí phức tạp
 - Tích hợp thêm các luật suy diễn

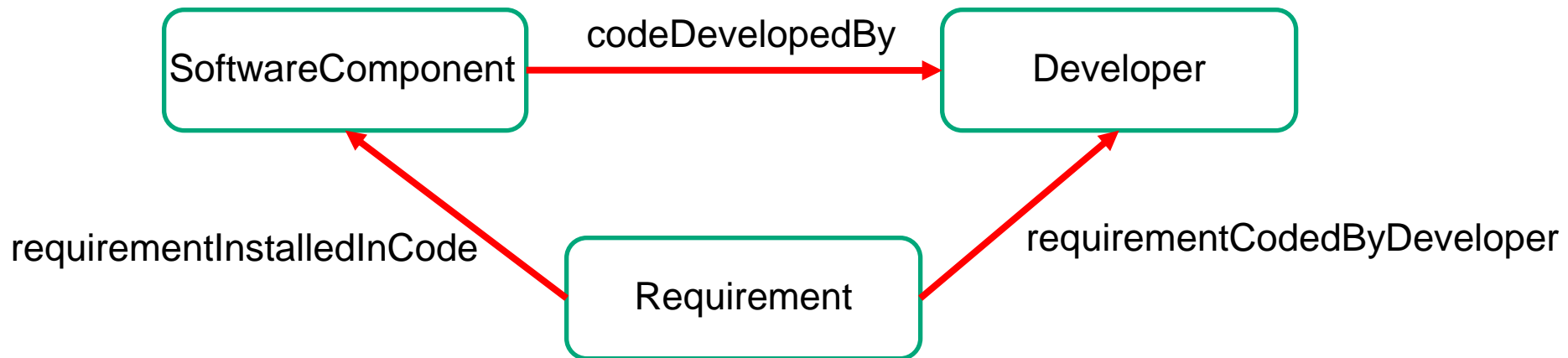


Ontology

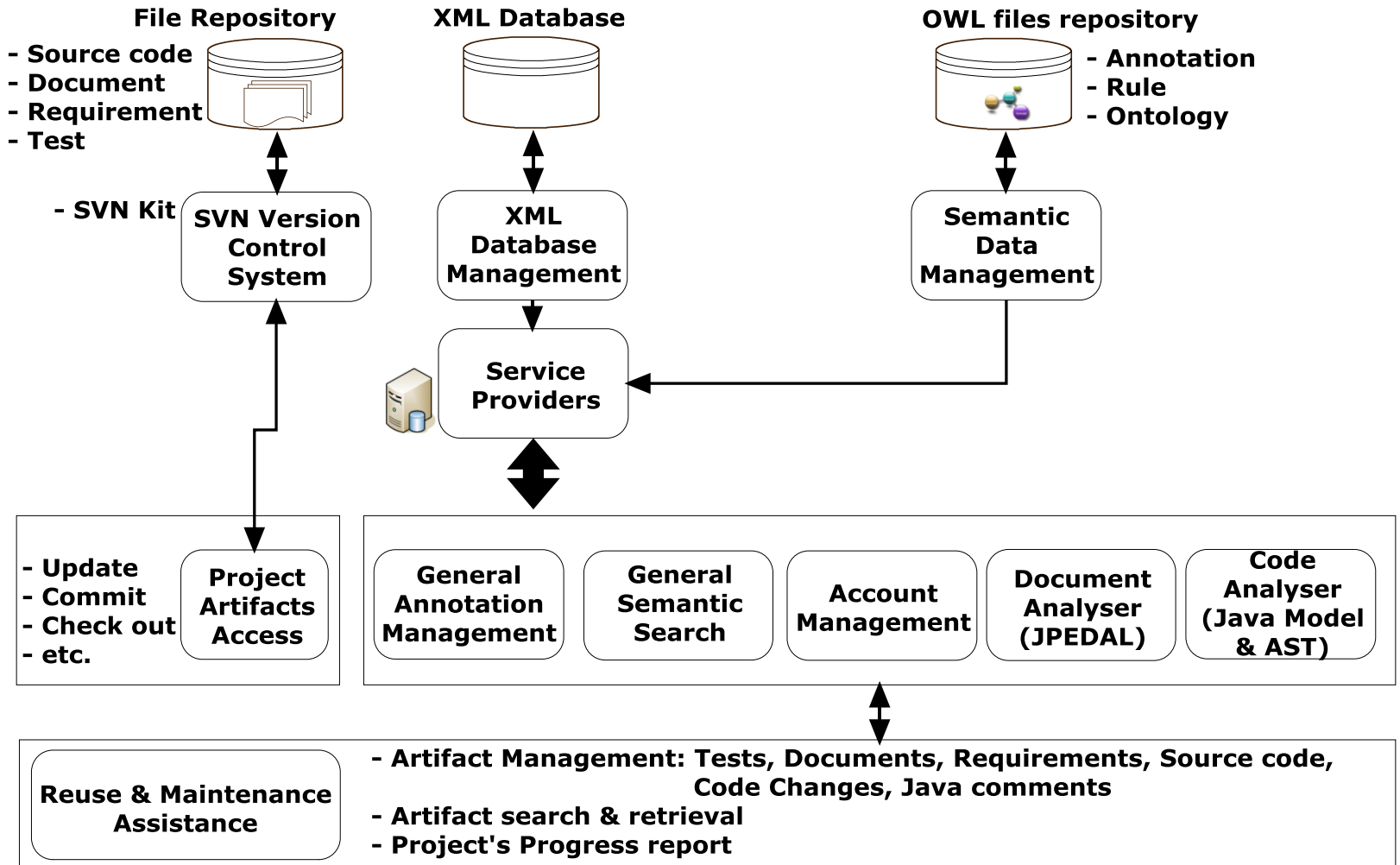
- Ngôn ngữ: OWL full
- Có tham khảo các nghiên cứu khác
 - Đảm bảo CL thiết kế & khả năng dễ chia sẻ
- 1 domain ontology:
 - Software Engineering
- 1 application ontology:
 - Software Maintenance & Software Reuse Ontology
- Các luật SWRL

Ontology

■ Ví dụ



1. Kiến trúc hệ thống





2. Các chức năng của hệ thống

- 6 views:
 - Account Management
 - Semantic Annotator
 - Semantic Search
 - Document & source code support
 - Software Artifact Management
 - Project's Progress

2.1. Account management

The screenshot displays a web application interface with two main sections: 'User's Tasks' and 'Admin's Tasks'. The browser's address bar shows the URL 'http://localhost:8080/'.

User's Tasks

Login

Server:

User name:

Password: [Log out](#)

Project Team:

Change Password

User name:

Current PW:

New PW:

Admin's Tasks

Create new Account

User name:

Password:

Account's Type:

Remove Account

User name:

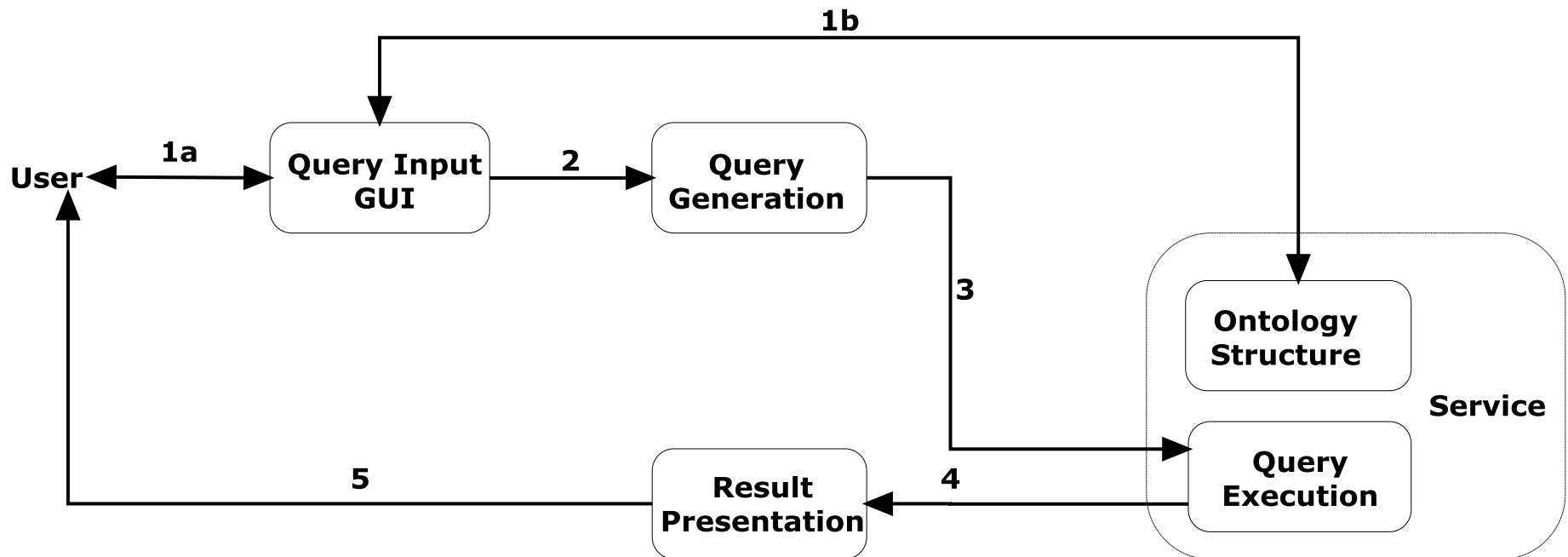
2.2. Semantic Annotator

The screenshot displays the Semantic Annotator application window, which includes a top menu bar with options like Account Management, Semantic Search, Document And Source, Manage Software Artifacts, Project's Progress Report, Tasks, and Problems. The main interface is divided into several panels:

- Class List (1):** A tree view on the left showing a hierarchy of classes starting with 'Thing', including 'Agent', 'CodeChange', 'Document', 'DocumentElement', 'FontFormat', 'Layer', 'Link', 'Metric', 'Requirement', 'SoftwareComponent', 'Test', 'Topic', and 'Type'.
- Option Panel (2):** A central panel containing a list of instances (e.g., 'Developer_duc07', 'ProjectTeam-new', 'Developer-Duc07', 'Tester-HienPT', 'Tester-Duc07', 'ProjectTeam-06_09_2010.16_43_57', 'ProjectManager-06_09_2010.16_44_16', 'ProjectManager-DungCT', 'Developer-HienPT') and a 'Filter by Name' input field.
- Annotation Panel (3-12):** A right-hand panel for editing an instance. It includes:
 - Buttons (3-5):** 'Add', 'Delete', and 'Save owl' buttons at the top.
 - Form Fields (6-8):** A 'Developer_duc07' label, a 'Filter by Name' field, and a 'Description' field with 'Save' and 'Full URI' options.
 - Property Table (9-11):** A table with 'Property' and 'Value' columns. It lists 'hasName' (value: 'Duc07') and 'memberOf' (value: 'ProjectTeam-new'). Each row has '+', 'x', and globe icons for editing.
 - authorOf Field (12):** A text input field at the bottom for the 'authorOf' property.

2.3. Semantic Search (1)

■ Mô hình xử lý



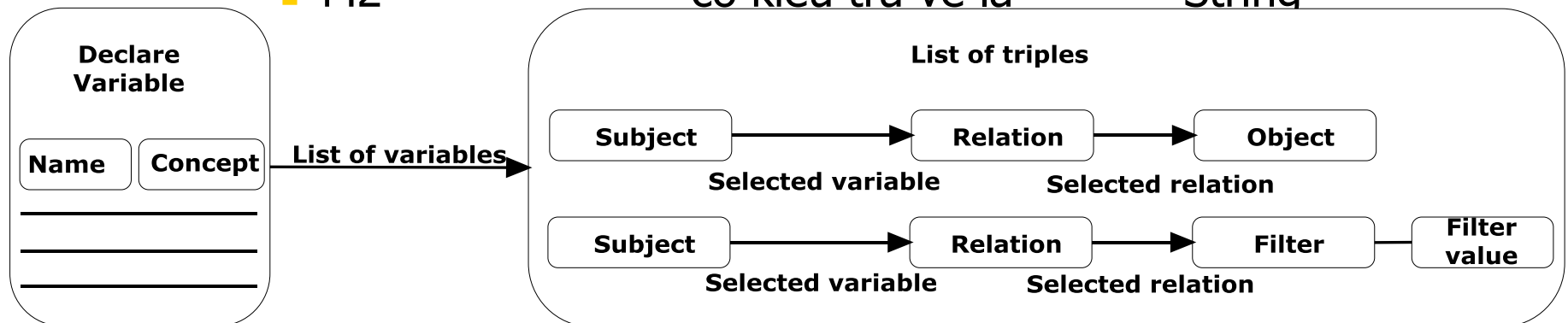
2.3. Semantic Search (2)

■ Mô hình thiết kế

- Ý tưởng: Phân rã các tiêu chí tìm kiếm thành các bộ 3.

- VD: Tìm 1 lớp java có 2 phương thức, 1 phương thức kiểu trả về là String

- Lớp Java C có phương thức M1
- Lớp Java C có phương thức M2
- M2 có kiểu trả về là String





2.3. Semantic Search (3)

- Ví dụ 1: Tìm các lớp Java có 2 method
 - `getDeveloperName()`
 - Có kiểu trả về là `String`
 - `getTesterName()`



2.3. Semantic Search (4)

- Các bộ 3:

(1) ?C	rdf:type	Class.
(2) ?M1	rdf:type	Method.
(3) ?M2	rdf:type	Method.
(4) ?P	rdf:type	JavaPrimaryType.

(5) ?C	hasMethod	?M1.
(6) ?C	hasMethod	?M2.
(7) ?M1	hasName	?x1 filter(fn:contains(?x1, `getDeveloperName`)).
(8) ?M1	returnType	?P.
(9) ?P	hasName	?x2 filter(fn:equals(?x2, `String`)).
(10) ?M2	hasName	?x3 filter(fn:contains(?x2, `getTesterName`)).

2.3. Semantic Search (5)

Variable Restriction Result

Name	Type	
C	Class	<input type="checkbox"/>
M1	Method	<input type="checkbox"/>
M2	Method	<input type="checkbox"/>
P	JavaPrimaryType	<input type="checkbox"/>
		<input type="checkbox"/>

Reset Arrange New variable Fast Choose

Variable Restriction Result

☐ + P(JavaPrimaryType)

☒ - C(Class)

- ☐ classIsExtendedBy
- ☒ classOf
- ☐ codeDesignedInDoc
- ☐ codeDevelopedBy
- ☐ codeInstallRequirement
- ☐ codeIllustratedInImage

Update Query No results found! 3 << < > >>

Account Semantic Semantic Document Manage So Project's P Tasks Problems

Variable Restriction Result

New Obj...	Subject	Predicate	Object/Filter	Filter Value	Optional	
<input type="checkbox"/>	C(Class)	hasMethod	M1(Method)		<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	C(Class)	hasMethod	M2(Method)		<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	M1(Method)	hasName	Contains	getDeveloperName	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	M1(Method)	returnType	P(JavaPrimaryType)		<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	P(JavaPrimaryType)	hasName	Equals	String	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	M2(Method)	hasName	Contains	getTesterName	<input type="checkbox"/>	<input type="checkbox"/>

Reset Arrange New Restriction



2.3. Semantic Search (4)

- Ví dụ 2: Tìm tất cả các lớp về Service nào đó, viết bởi Developer Peter
- Các bộ 3:
 - (1) ?D rdf:type Developer.
 - (2) ?C rdf:type Class.
 - (3) ?T rdf:type WebServiceTopic
 - (4) ?C hasTopic ?T.
 - (5) ?C codeDevelopedBy ?D.
 - (6) ?C hasTopic ?T.

2.3. Semantic Search (5)

This screenshot shows a window titled 'Semantic Search' with tabs for 'Variable', 'Restriction', and 'Result'. The 'Variable' tab is active, displaying a table with columns 'Name' and 'Type'. Below the table are buttons for 'Reset', 'Arrange', 'New variable', and 'Fast Choose'.

Name	Type
C	Class
D	Developer
T	WebServiceTopic

This screenshot shows a window titled 'Semantic Search' with tabs for 'Variable', 'Restriction', and 'Result'. The 'Restriction' tab is active, displaying a list of variables with checkboxes and plus signs. Below the list are buttons for 'Update' and 'Query', a numeric input field set to '3', and navigation arrows.

Project's Progress Report

- ☐ + D(Developer)
- ☐ + T(WebServiceTopic)
- ☐ + C(Class)

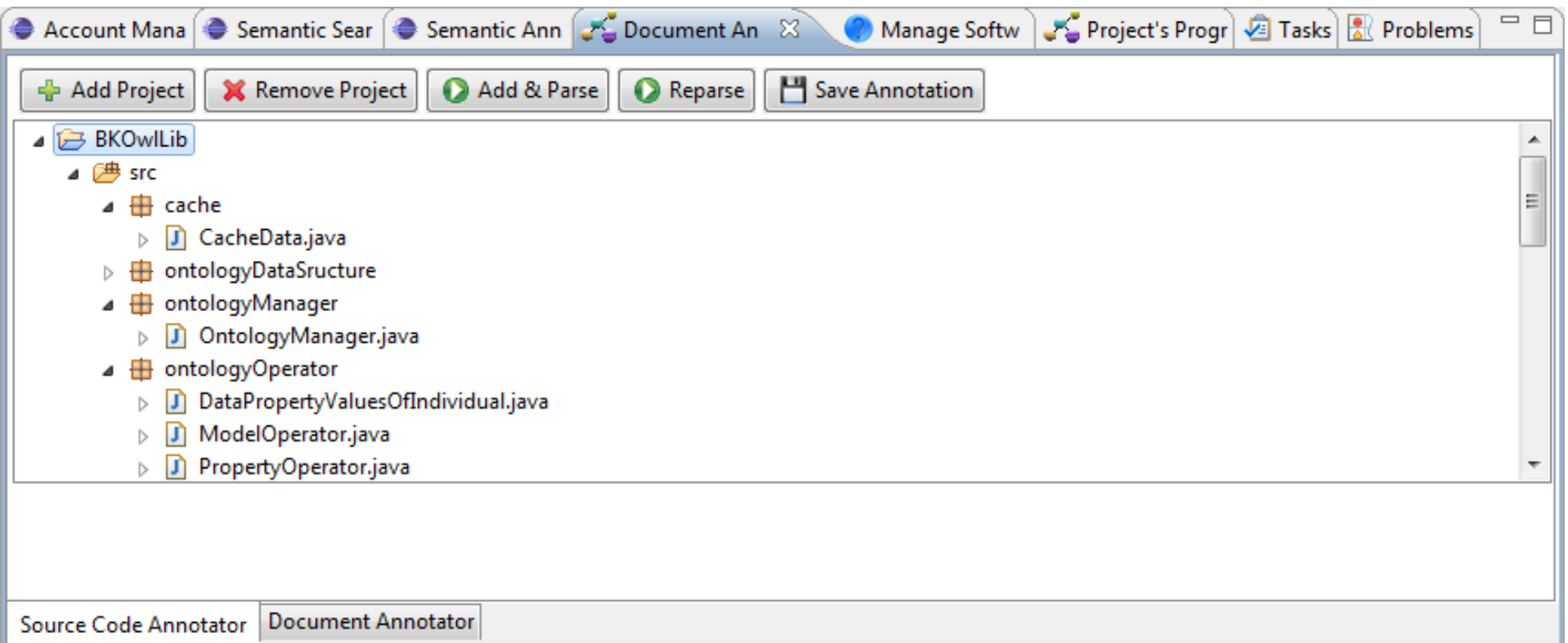
Update Query 3 << < > >>

This screenshot shows a window titled 'Semantic Search' with tabs for 'Variable', 'Restriction', and 'Result'. The 'Restriction' tab is active, displaying a table with columns: 'New Obj...', 'Subject', 'Predicate', 'Object/Filter', 'Filter Value', 'Optional', and a final column with checkboxes. Below the table are buttons for 'Reset', 'Arrange', and 'New Restriction'.

New Obj...	Subject	Predicate	Object/Filter	Filter Value	Optional	
<input type="checkbox"/>	C(Class)	codeDevelopedBy	D(Developer)		<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	C(Class)	hasTopic	T(WebServiceTopic)		<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>					<input type="checkbox"/>	<input type="checkbox"/>

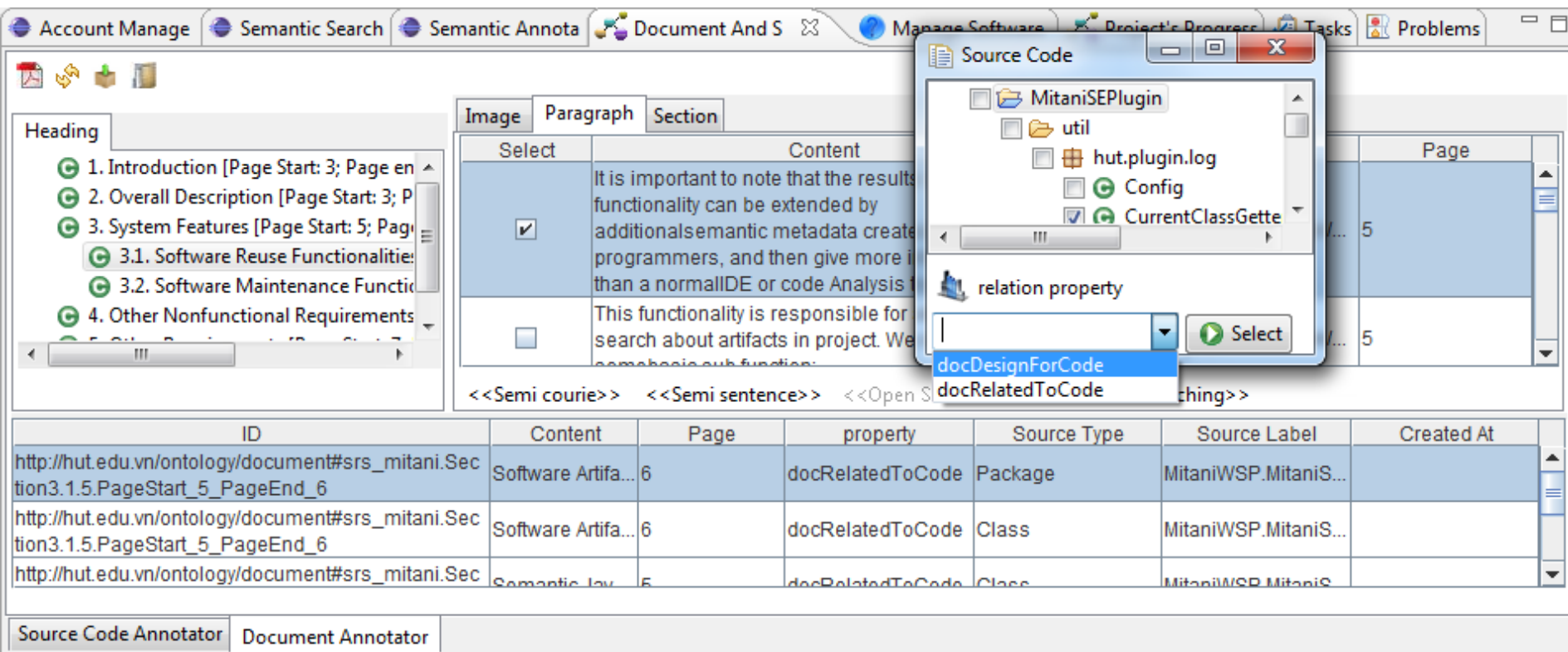
2.4. Document & Source code supports

- Bộ phân tích mã nguồn



2.4. Document & Source code supports (2)

■ Bộ phân tích tài liệu



The screenshot displays a software development environment with the following components:

- Heading Panel:** Lists document sections such as "1. Introduction", "2. Overall Description", "3. System Features", and "3.1. Software Reuse Functionalities".
- Document And S Panel:** Contains a table with document metadata.
- Source Code Panel:** Shows a tree view of the project structure, including "MitaniSEPlugin", "util", "hut.plugin.log", "Config", and "CurrentClassGetter".
- relation property Dialog:** A dropdown menu is open, showing "docDesignForCode" and "docRelatedToCode".

ID	Content	Page	property	Source Type	Source Label	Created At
http://hut.edu.vn/ontology/document#srs_mitani.Section3.1.5.PageStart_5_PageEnd_6	Software Artifa...	6	docRelatedToCode	Package	MitaniWSP.MitaniS...	
http://hut.edu.vn/ontology/document#srs_mitani.Section3.1.5.PageStart_5_PageEnd_6	Software Artifa...	6	docRelatedToCode	Class	MitaniWSP.MitaniS...	
http://hut.edu.vn/ontology/document#srs_mitani.Sec	Semantic law...	5	docRelatedToCode	Class	MitaniWSP.MitaniS...	

Source Code Annotator | Document Annotator

2.5. Software artifact management

The screenshot displays the 'Manage Software Artifacts' window, which includes a toolbar with icons for refresh, new requirement, and delete requirement. Below the toolbar is a table with columns: Delete, Name, Content, Described in D..., Developer, Tested In, Is Installed, Start Date, and End Date. The table contains three rows of requirements. A red box highlights the table area, and a red circle with the number 5 is placed below it. A red box highlights the 'Filter by Developer' dropdown menu, and a red circle with the number 6 is placed next to it. A red box highlights the 'View All' button, and a red circle with the number 7 is placed next to it. The 'Manage Comment' tab is selected in the top navigation bar.

Account Manage Semantic Search Semantic Annotatio Document And Sour Project's Progress Re Manage Software Arti Tasks Problems

Manage Requirements Manage Tests Manage Code Code Changed Manage Comment

refresh New Requirement Delete Requirement

1 2 3 4

Delete	Name	Content	Described in D...	Developer	Tested In	Is Installed	Start Date	End Date
<input type="checkbox"/>	Requirement for Query Creator Modul	Requirement for Query Creator Modul	srs_mitani	Trinh Tuan Dat				
<input type="checkbox"/>	Requirement for Semantic Javadoc	Requirement for Semantic Javadoc	srs_mitani			Yes		
<input type="checkbox"/>	Document Code Link Annotator Modul Requirement	Document Code Link Annotator Modul Requirement	srs_mitani	Trinh Tuan Dat		Yes		

5

Filter by Developer

6 7

View All Finished Requirement Processing Requirement

2.6. Project progress report

Account Manage Semantic Search Semantic Annot Document And S Manage Software Project's Progress Tasks Problems

Rate of process

choose a project MitaniProject refresh rate of process

General information

ID	name	Description	Developer team	Tester team	Requirement	Developers	Testers
http://hut.edu.vn/...	Mitani SOICT Collaboration Project	Mitani SOICT Collaboration Project	1	1	2	3	2

Detail

Requirements detail Developers detail Testers detail

ID	Name	Requirements	completed	installing	not install
http://hut.edu.vn/ontology/docu...	Phan Thanh Hien	1	1	0	0
http://hut.edu.vn/ontology/docu...	Trinh Tuan Dat	3	2	0	1



3. Các loại đối tượng trao đổi

■ Đối tượng thường

- Bắt buộc phải có mã nguồn java cho:
 - Lớp tương ứng với đối tượng truyền
 - Các lớp mà các phương thức/thuộc tính của đối tượng dùng đến (Tham số, giá trị trả về của phương thức, kiểu của thuộc tính)
 - Trừ các lớp Java nguyên thủy (String, Integer, ...)

■ Đối tượng kiểu List

- Với 1 số api có thể gặp khó khăn (ví dụ: JAX-WS)



4. Các nhóm dịch vụ

- 4.1. Tương tác trực tiếp với ontology
 - Load/reload/backup, ...
- 4.2. Tương tác với class trong ontology
- 4.3. Tương tác với property trong ontology
- 4.4. Tương tác với individual trong ontology
- 4.5. Nhóm dịch vụ khác



4.1. Tạo model từ ontology

- //Lưu model hiện tại ra 1 file owl
- `public Message backupOWL(String owlFileName)`
- //Xem một ontology đã được load chưa
- `public boolean isInitited(String owlFileName)`
- //Load vào 1 ontology (nếu đã tồn tại thì load lại)
- `public boolean reloadOntology(String owlFileName)`
- //Ghi ontology ra một file
- `public void writeToOWL(String owlFileName, String fileName)`



4.2. Tương tác với class trong ontology

- //Liệt kê các Property của class
- `public ArrayList<PropertyData> getAllClassProperties(String owlFileName, String classname)`
- //Liệt kê các subclass của một class
- `public ArrayList<ClassData> getSubClasses(String owlFileName, String classname, Boolean direct)`
- //Liệt kê các superclass của một class
- `public ArrayList<ClassData> getSuperClasses(String owlFileName, String classname)`
- //Liệt kê các Class của một ontology
- `public ArrayList<ClassData> listClasses(String owlFileName)`
- //Lấy một Class trong ontology theo tên
- `public ClassData getClassByName(String owlFileName, String className)`



4.3. Tương tác với property trong ontology

- //Liệt kê các Property của một ontology
- `public ArrayList<PropertyData> listProperties(String owlFileName)`

- //Lấy một Property trong ontology theo tên
- `public PropertyData getPropertyByName(String owlFileName, String propertyName)`

- //Liệt kê các SuperProperty của một Property
- `public ArrayList<PropertyData> getSuperProperties(String owlFileName, String propertyname)`

- //Liệt kê các SubProperty của một Property
- `public ArrayList<PropertyData> getSubProperties(String owlFileName, String propertyname)`



4.3. Tương tác với property trong ontology (2)

- //Lấy kiểu dữ liệu của một DatatypeProperty
- `public String getPropertySpecificDataType(String owlFileName, String propertyname)`

- //Lấy Range của một ObjectProperty
- `public ArrayList<String> getObjectPropertyRanges(String owlFileName, String propertyname)`

- //Lấy Domain của một ObjectProperty
- `public ArrayList<String> getObjectPropertyDomains(String owlFileName, String propertyname)`

- //Lấy về Domain của một property
- `public ResourceData getDomain(String owlFileName, String propertyname)`



4.4. Tương tác với individual trong ontology

- //Tạo individual cho một class
- `public boolean createInstance(String owlFileName, String instancename, String classname)`
- //Kiểm tra 1 individual có tồn tại
- `public Boolean checkExistIndividual(String owlFileName, String individualName)`
- //Lấy 1 individual theo tên
- `public IndividualData getIndividualByName(String owlFileName, String individualName)`
- //Liệt kê các instance của 1 class trong ontology
- `public ArrayList<String> listClassInstance(String owlFileName, String classname)`



4.4. Tương tác với individual trong ontology (2)

- //Liệt kê các instance của một class và cả instance của các subclass
- `public ArrayList<String> listAllRelatedInstance(String owlFileName, String classname)`
- //Thêm một giá trị thuộc tính kiểu Datatype cho Individual
- `public void addDatatypeProperty(String owlFileName, String propertyname, String value, String individualname)`
- //Thêm một ObjectProperty cho Individual
- `public void addObjectProperty(String owlFileName, String property, String individualValue, String individualname)`
- //Lấy về giá trị 1 thuộc tính cụ thể của 1 individual cụ thể
- `public ArrayList<String> getValueOfSpecificPropertyForIndividual(String owlFileName, String instanceID, String propertyName)`

4.4. Tương tác với individual trong ontology (3)

- //Lấy về tất cả các giá trị các thuộc tính của 1 individual
- `public ArrayList<BKIndividualProperty> getValuesOfIndividual(String owlFileName, String instanceID)`
- //Lưu tất cả cục dữ liệu liên quan tới 1 individual
- `public void saveValuesOfIndividual(String owlFileName, ArrayList<InstanceData> instanceDataList, Boolean isBig)`
- //Xóa một individual trong ontology
- `public void removeIndividual(String owlFileName, String individualName)`
- //Lấy ra lớp của 1 individual
- `public String getClassOfIndividual(String owlFileName, String individualName)`



4.5. Nhóm dịch vụ khác

- //Thực hiện câu truy vấn Sparql
- `public ArrayList<ArrayListData>
SparqlResultList(String owlFileName, String
querystring)`
- //Quản lý người dùng
- ...



Bài tập

- Bài tập 1: thiết kế các dịch vụ cần thiết cho giao diện truy vấn tìm kiếm
- Bài tập2: thiết kế các dịch vụ cần thiết cho giao diện annotator