Archimind User guide

Author: Klaas Andries de Graaf - https://github.com/kadevgraaf/

Version: V2

Date: 15-12-2016

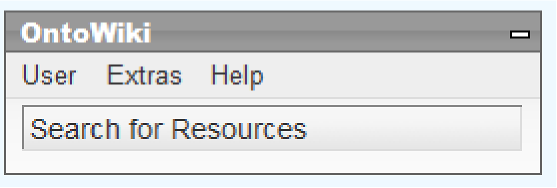
Introduction

This document contains instructions for using ArchiMind. Archimind is a semantic wiki for Software Architecture (SA) documentation management and retrieval and was adapted from OntoWiki (http://aksw.org/Projects/OntoWiki.html) version 0.9.5.

A concise overview of adaptations and their rationale is given in [1] Klaas Andries de Graaf - "**Annotating Software Documentation in Semantic Wikis**". Klaas Andries de Graaf, Peng Liang, Antony Tang, Hans van Vliet - "**How Organisation of Architecture Documentation Affects Architectural Knowledge Retrieval**" follows up on [1] in more detail and describes an experiment about Architecture Knowledge retrieval from the ArchiMind semantic wiki and traditional documentation.

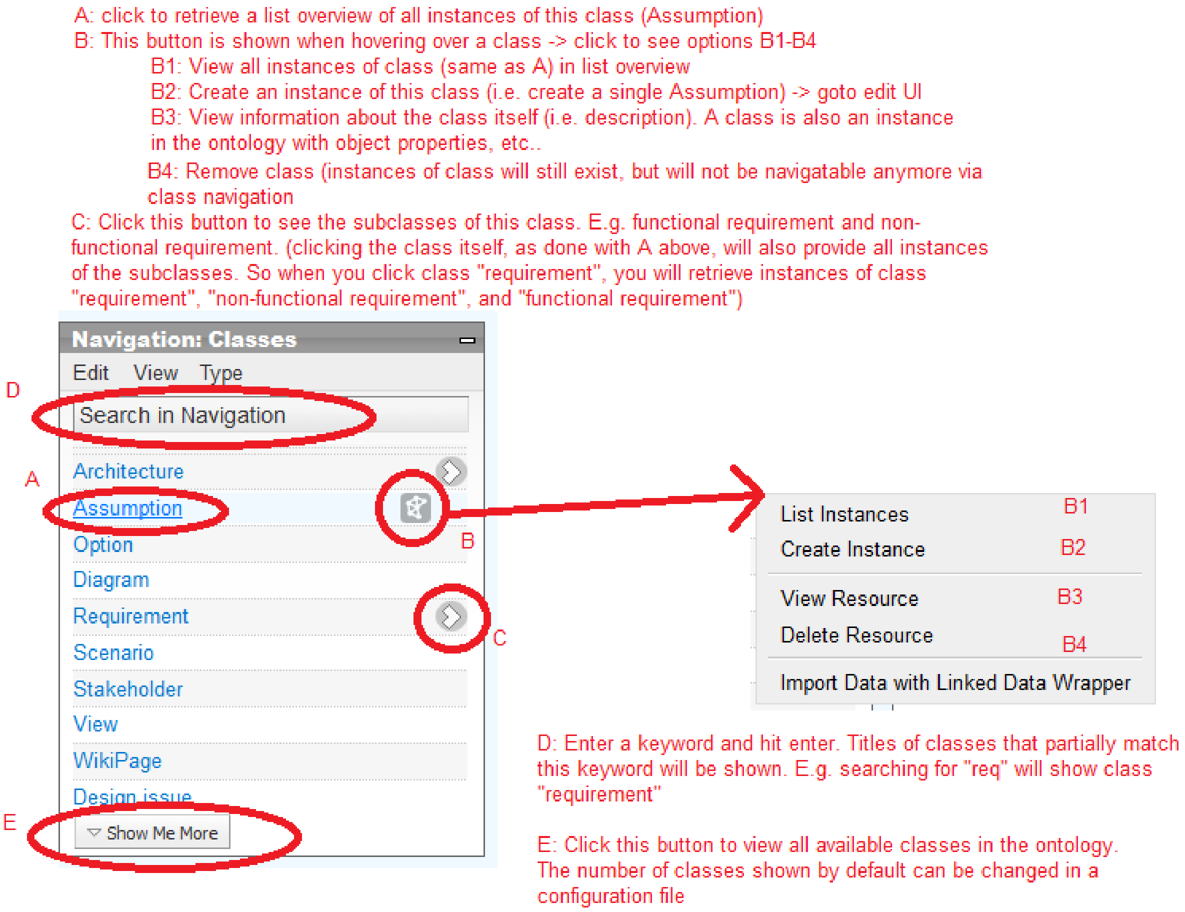
Keyword searching and Class navigation

One of the most straightforward navigation features of the ArchiMind UI is the keywords search panel. One can type a keyword in the field “Search for Resources” and hit enter to search the knowledge base for this keyword.



When searching for a keyword, a list overview (see below) will be returned with class instances (one on each row) that contain this keyword in their title, description or any other property that they have (e.g. author, note, label, documentation content, etc..).

Another UI element, or UI panel, is the class navigation. The picture below describes features of the class navigation panel.



A: click to retrieve a list overview of all instances of this class (Assumption)

B: This button is shown when hovering over a class -> click to see options B1-B4

B1: View all instances of class (same as A) in list overview

B2: Create an instance of this class (i.e. create a single Assumption) -> goto edit UI

B3: View information about the class itself (i.e. description). A class is also an instance in the ontology with object properties, etc..

B4: Remove class (instances of class will still exist, but will not be navigatable anymore via class navigation

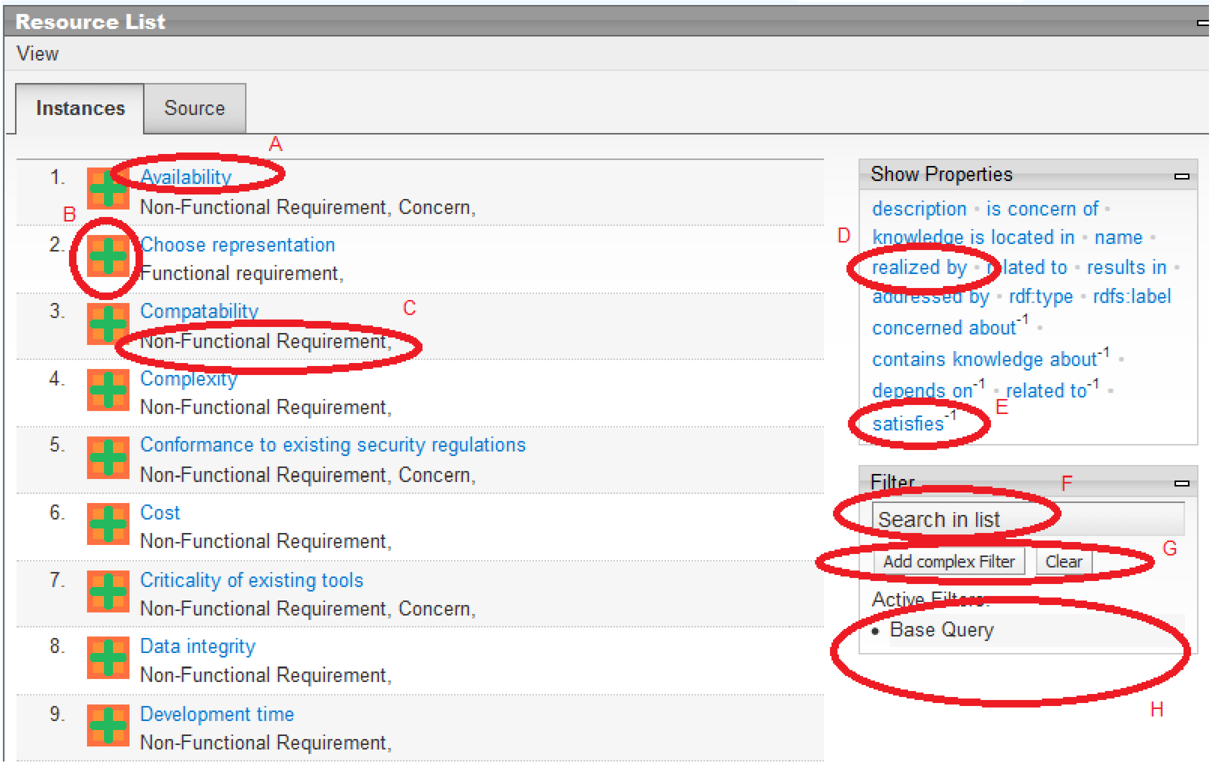
C: Click this button to see the subclasses of this class. E.g. functional requirement and non-functional requirement. (clicking the class itself, as done with A above, will also provide all instances of the subclasses. So when you click class "requirement", you will retrieve instances of class "requirement", "non-functional requirement", and "functional requirement")

D: Enter a keyword and hit enter. Titles of classes that partially match this keyword will be shown. E.g. searching for "req" will show class "requirement"

E: Click this button to view all available classes in the ontology. The number of classes shown by default can be changed in a configuration file

List overview

After clicking on a class or doing a keyword search, ArchiMind provides a list overview of the results. This is depicted and explained in the figure below:



1. Availability
Non-Functional Requirement, Concern,
2. Choose representation
Functional requirement,
3. Compatability
Non-Functional Requirement,
4. Complexity
Non-Functional Requirement,
5. Conformance to existing security regulations
Non-Functional Requirement, Concern,
6. Cost
Non-Functional Requirement,
7. Criticality of existing tools
Non-Functional Requirement, Concern,
8. Data integrity
Non-Functional Requirement,
9. Development time
Non-Functional Requirement,

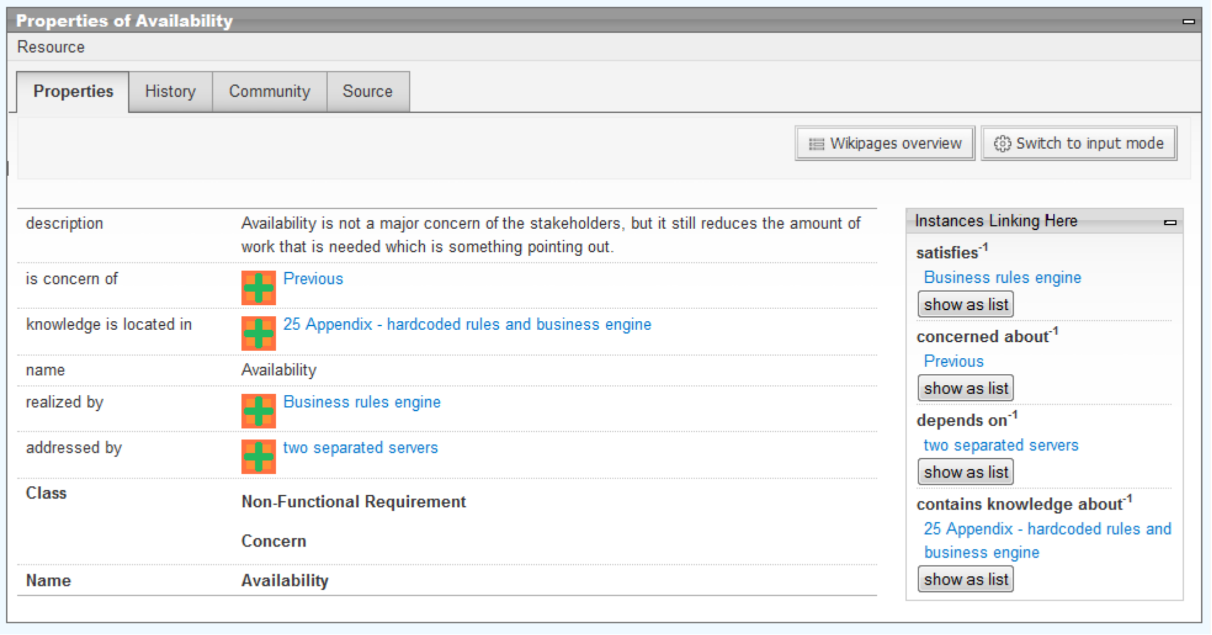
description • is concern of • knowledge is located in • name • realized by • related to • results in • addressed by • rdf:type • rdfs:label concerned about^{-1} • contains knowledge about^{-1} • depends on^{-1} • related to^{-1} • satisfies^{-1}

- Base Query

A: Title of ontology class instance, click to view this instance
B: Expand ontology class instance in list
C: Class(es) of ontology class instance (Multiple classes can be assinged to an instance, see Result 1 in the list
D: Show an extra column (or "facet") in the list which contains a property (or shows a relationship). By clicking on "realized by" and extra column will show how the components, subsystems, UI's, decisions, etc… that realize each of the requirements in the list.
E: "-1" means that this is an "incoming" or "inverse" relationship. Clicking on satisfies will show an extra column that contains e.g. architectural elements that satisfy the requirement.
F: Keyword filter: filter the instances in the list based on a keyword that they have in their title or one of their properties (e.g. description, author, content, etc…)
G Add a filter that uses object properties (e.g. "description" and "semantic relationships") to filter the list.
H: view active filters for the list. However over filters to show a button "X". Clicking the "X" button removes an active list filters.

Viewing an instance

After clicking on an instance. The UI below is displayed:



description	Availability is not a major concern of the stakeholders, but it still reduces the amount of work that is needed which is something pointing out.
is concern of	+ Previous
knowledge is located in	+ 25 Appendix - hardcoded rules and business engine
name	Availability
realized by	+ Business rules engine
addressed by	+ two separated servers
Class	**Non-Functional Requirement** **Concern**
Name	**Availability**

Instances Linking Here

satisfies^{-1}
Business rules engine
show as list

concerned about^{-1}
Previous
show as list

depends on^{-1}
two separated servers
show as list

contains knowledge about^{-1}
25 Appendix - hardcoded rules and business engine
show as list

The “+” button will, just like in the list overview, expand a property of the instance and show its details.

The button “wikipages overview” is a shorthand button for click on class “wikipage”.

The button “Switch to input mode” allows a user to edit the instance (see section below)

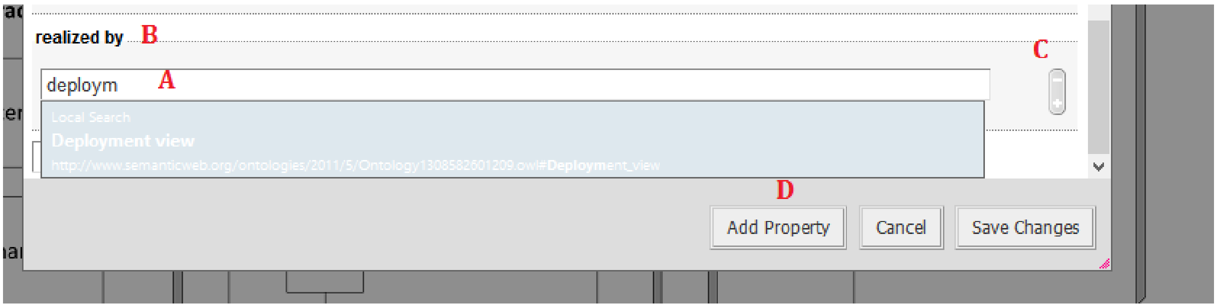
The “instances linking here” panel allows one to navigate to other instances via an “incoming” or “inverse” relationship. I.e. relationships that originate from other instances and link to this instance. Clicking on “show as list” will show a list of (one or) multiple instances that are inversely related via the same relationship.

A good improvement for this UI is that the class of related instances is shown. (e.g. “*two separated servers <class: requirement>*” in above figure

Editing instances

To start editing instances, click on button "Switch to input mode" (to go to input mode) and subsequently on button "Edit Properties".

A dialog will show up. The picture below illustrates part of this dialog.



realized by
B
deploym
A
Local Search
Deployment view
http://www.semanticweb.org/ontologies/2011/5/Ontology1308582601209.owl#Deployment_view
C
D
Add Property
Cancel
Save Changes

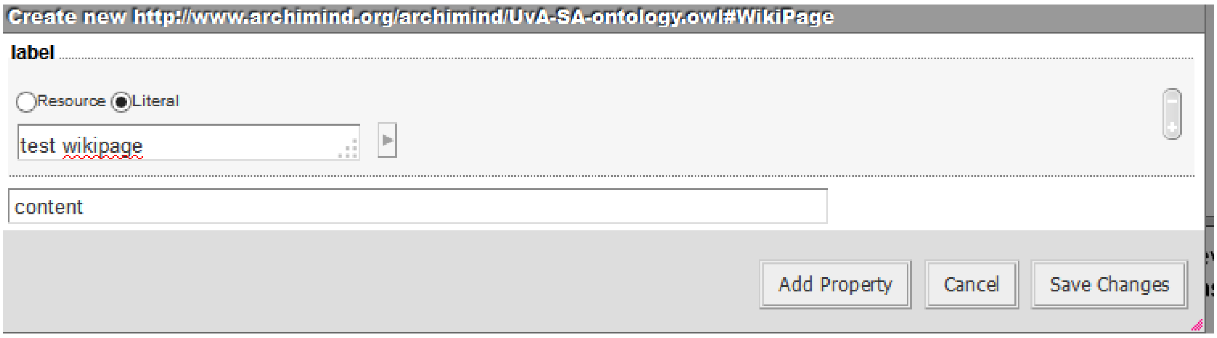
Click button "add property" (Label D) to add a new type of property (e.g. description, relationship to other AK, etc..) A blank input box will show up in which you can type the name of the property you want to add. The system will show suggestions for properties (together with their URI, label A) which need to be clicked in order to add the property.

Once a property is selected it will be shown above a text input box (Label B)

Click the "+" button at the right of the input box to add more information or relationships and "-" to remove it (Label C).

Adding documentation text as wikipage content

To add content from documentation (e.g. copied from MS word/OpenOffice) you have to follow the steps above described in section "editing instances" with the exception of typing "content" in the property input box. This is depicted below.

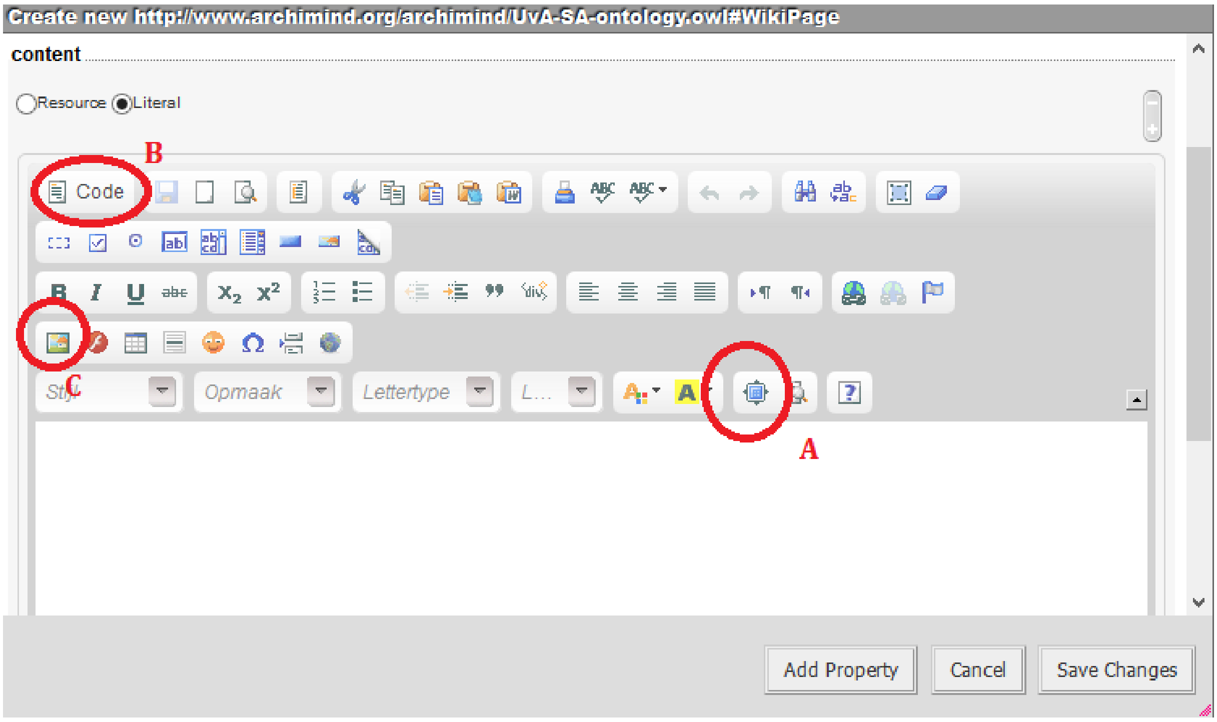


Create new http://www.archimind.org/archimind/UvA-SA-ontology.owl#WikiPage
label
○Resource ●Literal
test wikipage
content
Add Property
Cancel
Save Changes

After clicking "add property" and typing content in the input box, hit the enter button.

A WYSIWYG editor wil show up in which documentation content can by pasted (e.g. using CTRL-V) that was copied (e.g. using CTRL-V) from popular text editors. lay-out should be preserved (and stored as HTML) when doing this. The WYSIWYG editor is depicte below. Label A higlights a button for full-screen editing mode. If the SA documentation text in the WYSIWYG editor gets corrupted or is hard to correct somehow, it can be fixed by editing it in HTML by clicking button "Code" (label B).

Pictures cannot be copy-pasted, however, they can be added to documentation text in the WYSIWYG editor by clicking the image button under red Label C . Note that there are multiple possibilities to upload pictures, two upload dialogs are available after clicking the image button. You could also upload pictures via FTP to the correct directory and only select the pictures using the image button under red Label C (this may be more practical).



Create new http://www.archimind.org/archimind/UvA-SA-ontology.owl#WikiPage

content

○Resource ●Literal

Using ArchiMind for experimentation purposes

Remember to check whether a users is still logged in when starting an experiment. The session timeout may cause a user to log-out, which causes some entropy when using ArchiMind/OntoWiki for experimentation.

References

OntoWiki:

http://aksw.org/Projects/OntoWiki.html

http://ontowiki.net/

https://github.com/AKSW/OntoWiki

https://github.com/AKSW/OntoWiki/releases

ArchiMind:

https://github.com/kadevgraaf/ - http://kadegraaf.nl/

- [1] Klaas Andries de Graaf - "**Annotating Software Documentation in Semantic Wikis**" - In *Proceedings of the fourth workshop on Exploiting semantic annotations in information retrieval (ESAIR '11)*, pages 5-6., ACM, 2011. [View paper on ACM Digital Library]
-
- [2] My PhD Thesis - "**Ontology-based Software Architecture Documentation**" (cover)

- [3] Klaas Andries de Graaf, Peng Liang, Antony Tang, Hans van Vliet - "**How Organisation of Architecture Documentation Affects Architectural Knowledge Retrieval**" - *Science of Computer Programming*, Volume 121, Pages 75-99 Elsevier, June 2016. [View paper on ScienceDirect]

- Klaas Andries de Graaf, Peng Liang, Antony Tang, Hans van Vliet - "**Supporting Architecture Documentation: A Comparison of Two Ontologies for Knowledge Retrieval**" - In *International Conference on Evaluation and Assessment in Software Engineering (EASE)*, pages 3:1--3:10, ACM, 2015. [View paper on ACM Digital Library]

- Klaas Andries de Graaf, Peng Liang, Antony Tang, Hans van Vliet - "**The Impact of Prior Knowledge on Searching in Software Documentation**" - In *Proceedings of the 2014 ACM symposium on Document engineering (DocEng)*, pages 189-198, ACM, 2014. [See paper on ACM Digital Library]

- Klaas Andries de Graaf, Peng Liang, Antony Tang, Willem van Hage, Hans van Vliet - "**An Exploratory Study on Ontology Engineering for Software Architecture Documentation**" - *Computers in Industry*, Vol. 65, nr. 7, pages 1053-1064, Elsevier, 2014. [View paper on ScienceDirect]

- Klaas Andries de Graaf, Antony Tang, Peng Liang, Hans van Vliet - "**Ontology-based Software Architecture Documentation**" - In *Proceedings of the Joint 10th Working IEEE/IFIP Conference on Software Architecture & 6th European Conference on Software Architecture (WICSA/ECSA)*, pages 121-130, IEEE Computer Society, 2012. [View paper on IEEE Xplore]