DialogueTech

Customization process overview

DialogueTech

Table of contents

DialogueTech

The challenge

Using your own language is perhaps the most natural way for people to interact intelligently with other people. However, it offers three major challenges when we communicate with a computer:

1. The number of ways of expressing a single question is very large.

2. The meaning of a question must be interpreted by a computer.

Consider the examples *We gave the monkeys bananas because they were over-ripe* and *We gave the monkeys bananas because they were hungry.* You need to teach the computer that monkeys may be hungry but not over-ripe!

An application must be easy to create, modify and maintain.

This breaks down into two main parts – managing the language interpreter itself and managing the user data. As an illustration consider an application where you search for information related to persons. This information may reside in e.g. address books, electronic calendars, mail, documents, etc. It can be referenced directly – *Find John Jones' phone number* or indirectly - *List Volvo contacts*. If the concept of a person is changed or removed it affects all applications, directly and indirectly. The challenge of managing user data is best solved using the original data and standard database technology.

DialogueTech

Development Process Overview

1. Define the domain.

A. Database-centric applications

All the information a user can ask for is held in a database

B. Corpus-centric applications

Users can ask any questions they like in a defined domain

2. Create a domain model

The domain model contains the information the Ergo engine needs to work out the intent of the question or command

3. Compile a run-time environment

Integrate the domain model with the core modules of the SDK and create an executable file

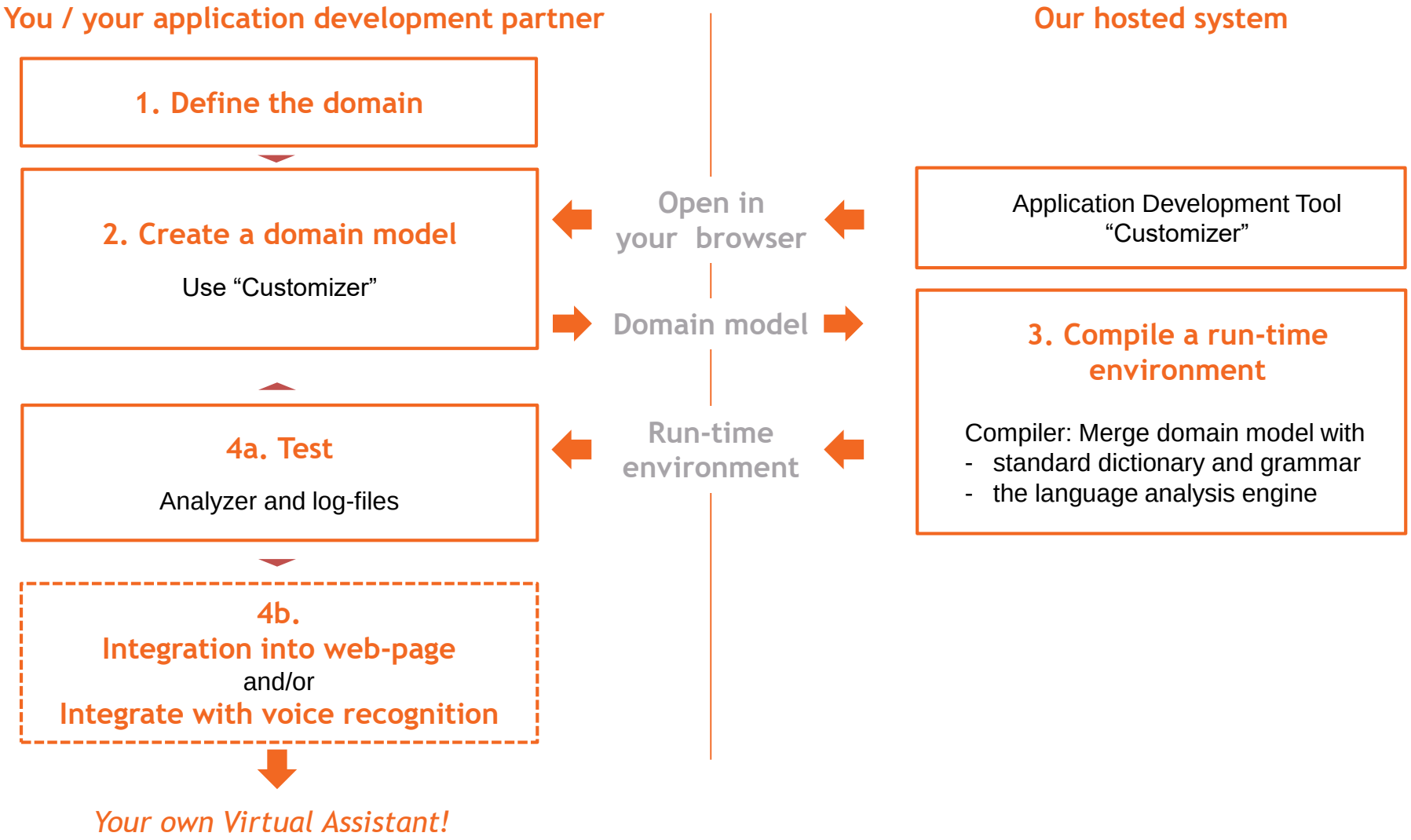
4. Test and integrate on target platform

Test and verify

Integrate into web-page and or voice recognition solution

DialogueTech

Development Process Overview



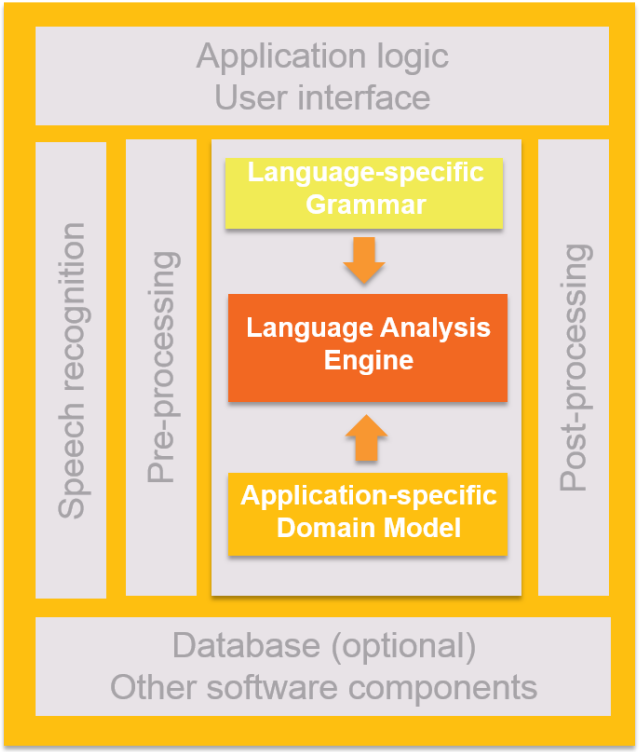
DialogueTech

Förvirrande bild?

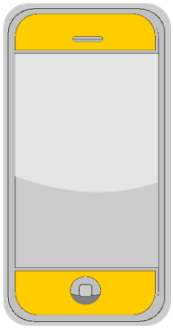
Packaged product

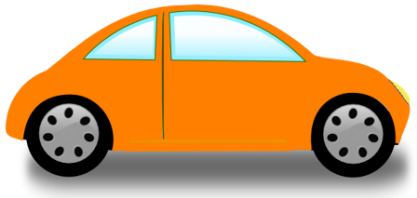
Speaker

Software package



Host system





DialogueTech

Table of contents

DialogueTech

Key concept: entity - relationship

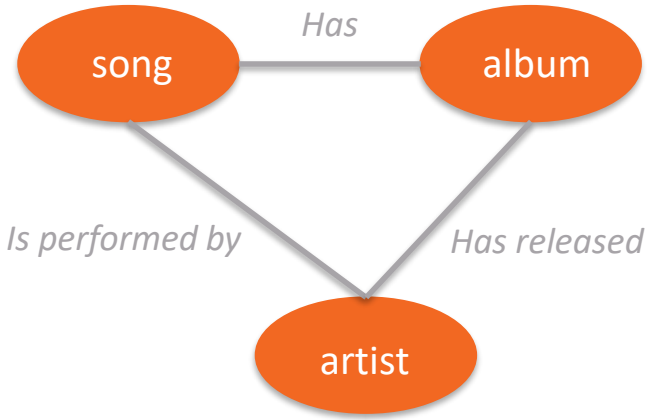
The logic in Ergo for interpreting the meaning of a question or statement is a binary entity-relationship model. It contains information about a few key-concepts that need to be identified and input into the application.

Entity: An entity may be defined as a thing which is recognized as being capable of an independent existence and which can be uniquely identified. An entity represents some aspect of the real world which can be distinguished from other aspects of the real world.

Term: A term describes how you refer to an entity. Consider the entity album which can be referred to with the terms album, CD, record, etc.

Relationship: A binary relationship captures how two entities are related to one another. All modeling facts can be described as binary relationships and are shown in an entity-relationship diagram.

Illustration: Consider an application to select music from a hard drive. Each song is characterized by things like the album it belongs to, the artist performing the song, the release date of the song, the genre it belongs to, etc. Artist, song, album, release date, genre etc. are entities and the corresponding relationships are a song is performed by an artist, a song has a release date, an artist has released a song etc.



DialogueTech

Key concepts

Database-centric applications

All information the user can ask for is, or can be, held in a database.

Corpus-centric applications

Users can ask for any information they like within a limited domain.

Output

- A list of entities and terms
- An entity-relationship diagram
- A set of test questions and commands
- A database diagram (optional)

DialogueTech

Database-centric applications

All information the user can ask for is, or can be, held in a database.

1. Identify the entities by inspecting the database from which we want to retrieve information. Often each column in the database corresponds to an entity.

2. Identify terms which can be associated with each entity. A thesaurus can be useful

3. Identify which entities have a relationship. You don't have to identify the type of relationship - only that it exists!

4. If the original data is not stored in a database, a relational database structure must be set up. This is a relatively straightforward procedure described in most database textbooks.

DialogueTech

Corpus-centric applications

Users can ask for any information they like within a limited domain.

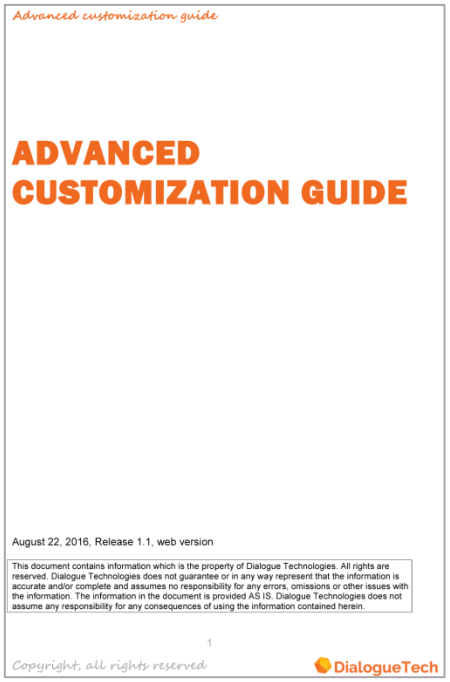
1. Develop set of typical user questions (a “corpus”). Use existing FAQs, interviews, manuals, etc.
2. Identify entities and terms by inspecting the corpus, extracting relevant entities.
3. Identify which entities have a relationship. You don’t have to identify the type of relationship - only that it exists!
4. Develop a relational database structure and link the database to the original data. This is a relatively straightforward procedure described in most database textbooks.

DialogueTech

Development tools and documentation

Documentation:
- Advanced Customization Guide
- Customization overview

Tools:
- Standard office tools



DialogueTech

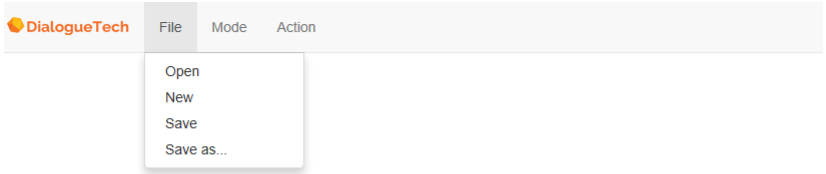
Table of contents

DialogueTech

Getting started

The domain model is a file with information used by the language analysis engine to interpret the query. It is created using a software tool, *Customizer*, based on a list of entities and corresponding terms together with an entity-relationship diagram. The *Customizer* runs in a standard browser

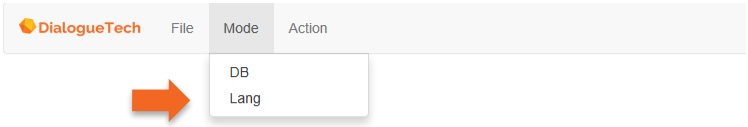
Continue on an existing domain-model



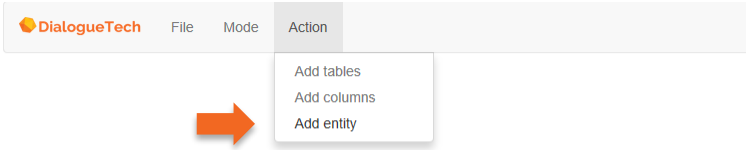
Press *File* ➔ *Open* to open the domain model

Note: Use the mouse scroll wheel to get the right size. Relocation of entities might also be necessary to get the right overview

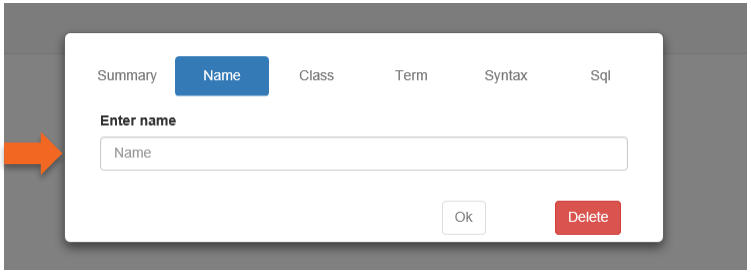
Starting a new domain-model



Press *Mode* ➔ *Lang* (language)

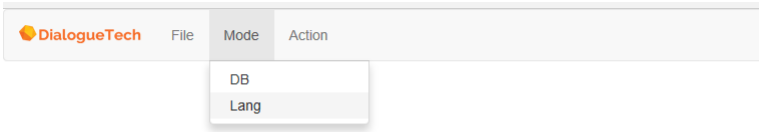


Press *Add entity* to open Entity window

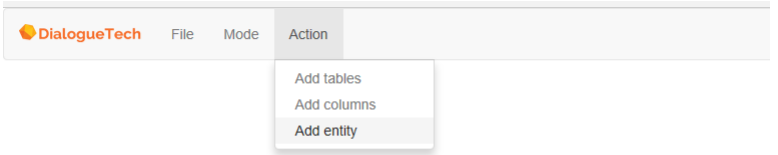


Create an entity - naming

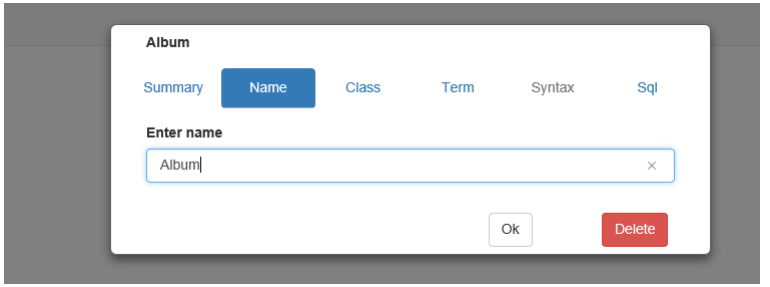
Select *Mode* -> *Lang*



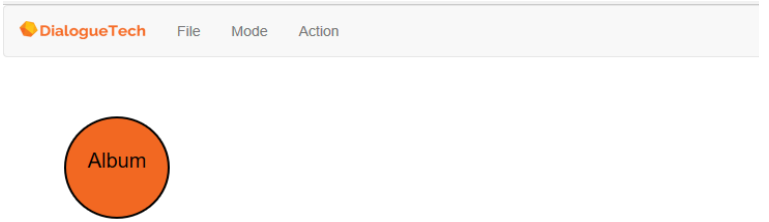
Select *Action* -> *Add entity*.



Give the entity a name, preferably describing the entity, and press *Ok*.

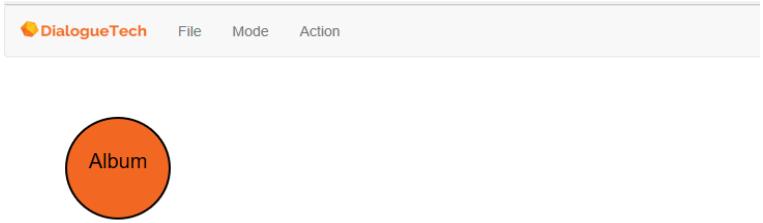


A first entity will now appear in your browser



4 types of entities

Until you have defined *the type* of the entity the symbol will be round



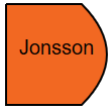
There are 4 types of entities. Once you define the type the symbol will change shape:

nouns

verbs

adjective

proper name



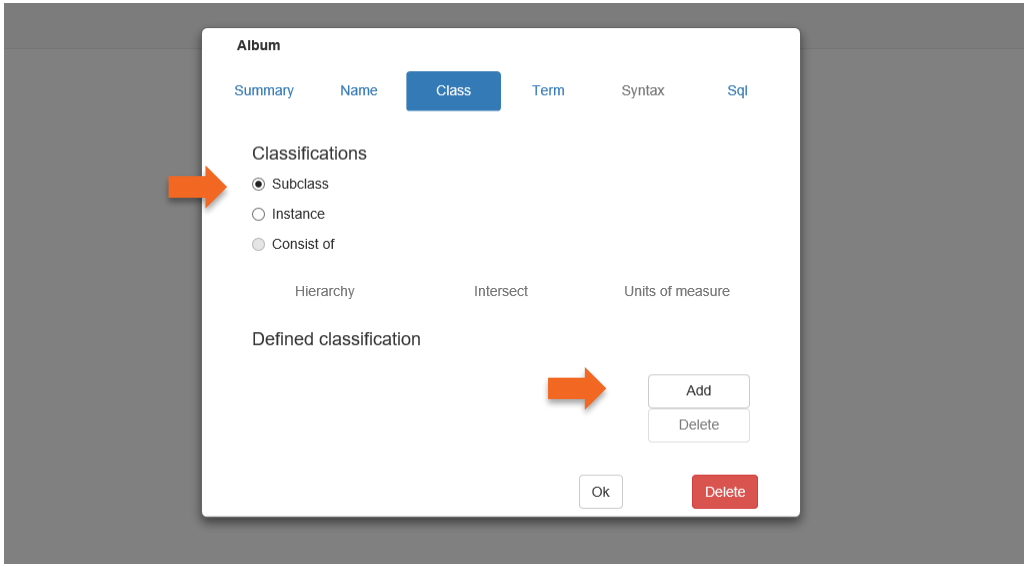
You select which type the entity is when you define terms for the entity.

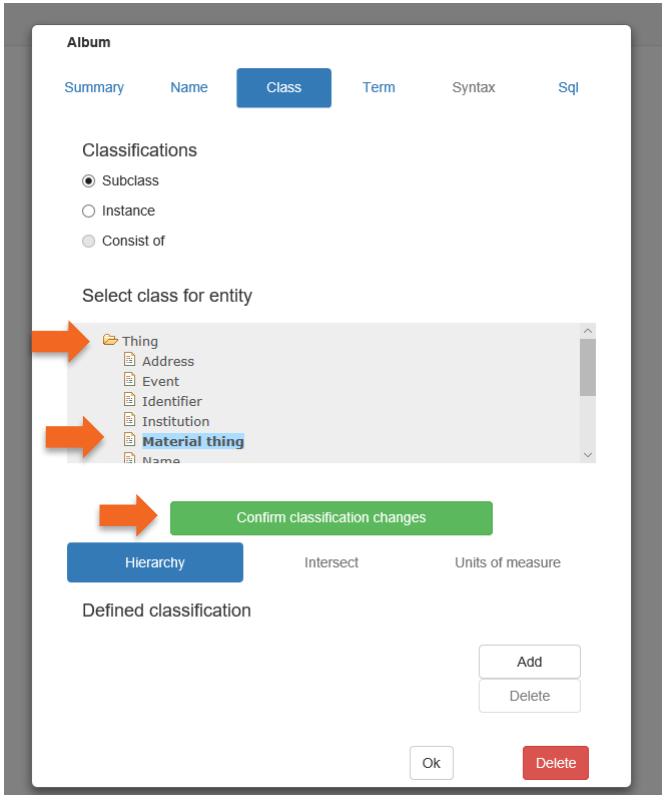
This manual gives a first introduction to how you create entities for all 4 types

Create a noun entity: classification

Double-click on an entity

Press *Class*, *Subclass* and *Add*





Open the scroll down menu by pressing *Thing*.

Classify the entity by selecting an option. If you don't find an option which feels like a 100% match, select *material thing*. Most nouns are classified as *material thing*.

Press Material thing and Confirm classification changes

Create a noun entity

defining terms

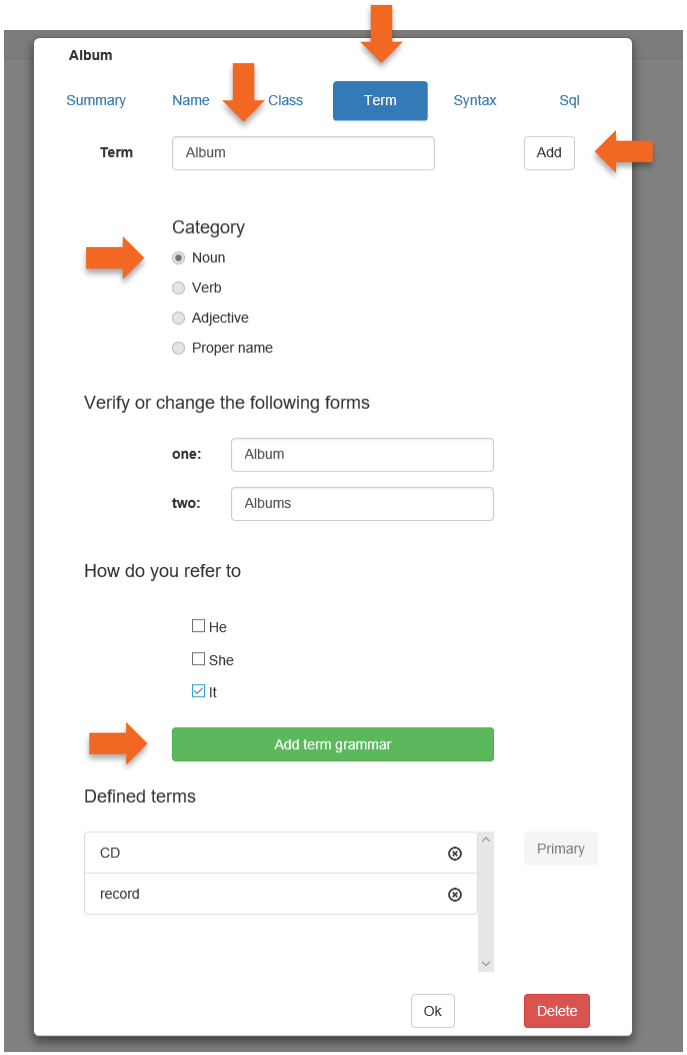
Press *Term* and enter the terms which will be used for the entity and select Category (Noun) – press *Add*.

The *Customizer* proposes noun forms and pronouns which are used in relation to the term selected. Edit if necessary.

Select “How you refer to” the noun Pronomen?

Press *Add term grammar*.

Repeat if you want to add more terms.



Noun entity

define syntax

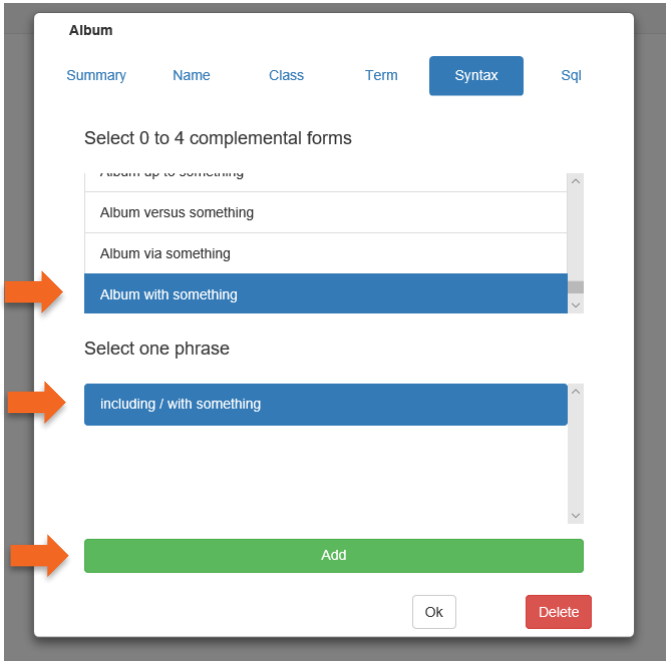
Press *Syntax* and select the syntactical environment (optional) in which the entity occurs.

The *Customizer* proposes a set of options and you can select 0-4 alternatives. Click on the selected options (selected options turns blue)

If you choose more than one option then you are asked to select one phrase

Press *Add*.

Note that generally you don't need to model spatial or temporal prepositional complements, e.g.
courses during spring semester
courses in London

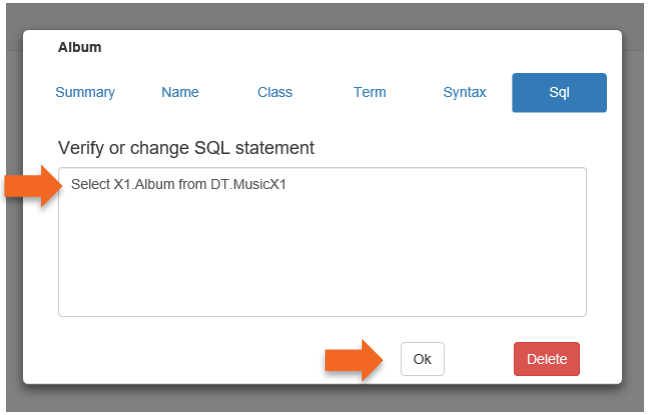


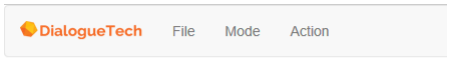
Noun entity

define SQL

Press *SQL* and type in the location in the database where information related to the entity is stored.

Press *OK*.



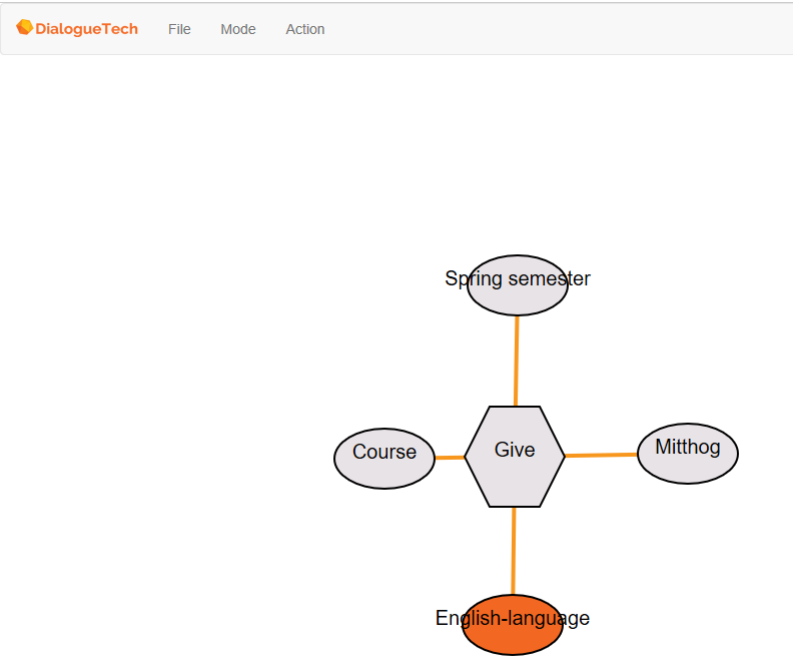


The entity has now been defined and appears as a symbol in the *Customizer*.

Create a verb entity

general

Domain modelling is based on the primary verb in the user's question. It is the base to which other entities are attached.



Create a verb entity

naming

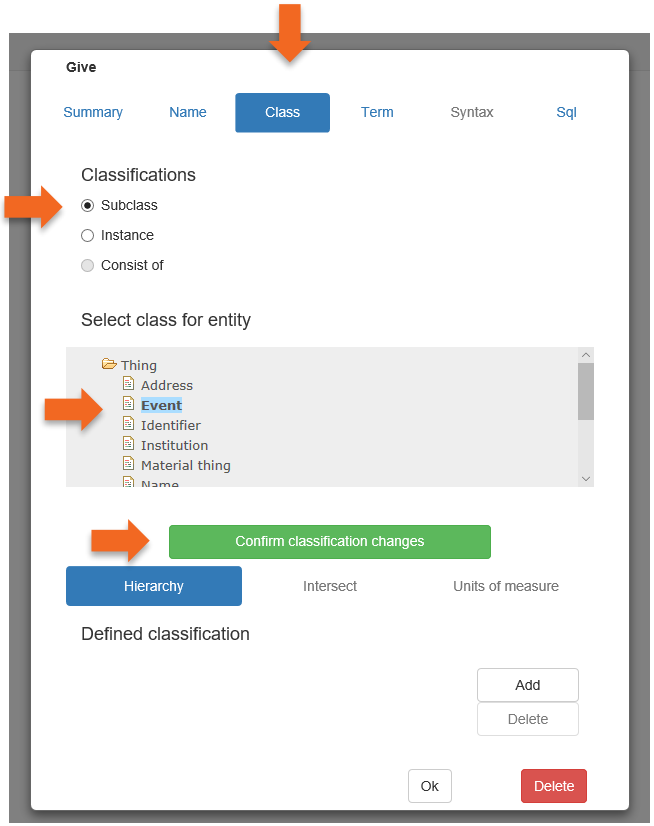
Same procedure as noun

- *Mode* ➔ *Lang*
- *Action* ➔ *Add entity*
- *Enter name* ➔ *OK*

classification

Same procedure as noun

- Double-click on an entity
- Press *Class*, *Subclass* and *Add*
- Open the scroll down menu by pressing *Thing*.
- Classify the verb entity as ***Event***. Verbs are always classified as events.
- Press *Confirm classification changes*.



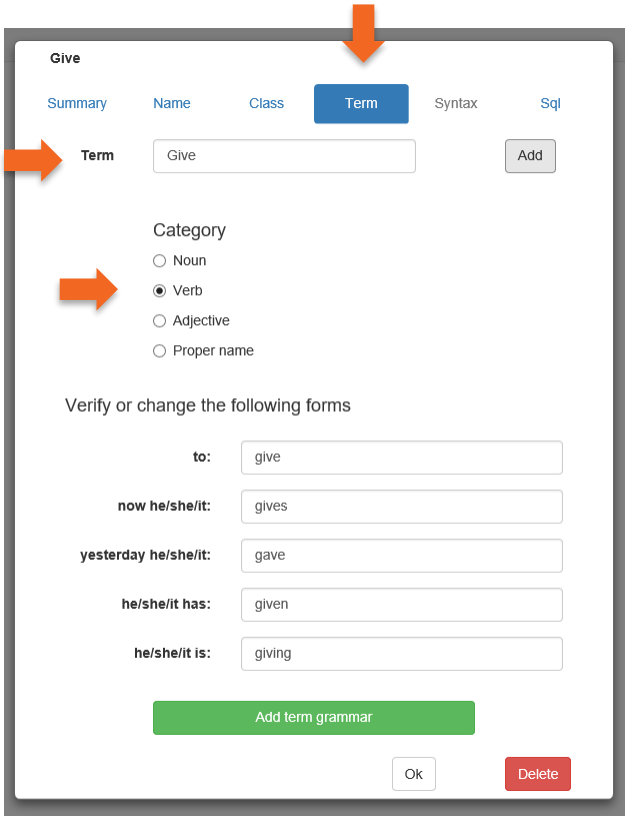
Create a verb entity

defining terms

Press *Term* and specify the terms which will be used for the entity.

For each verb term the Customizer suggests conjugations - correct these if they are incorrect.

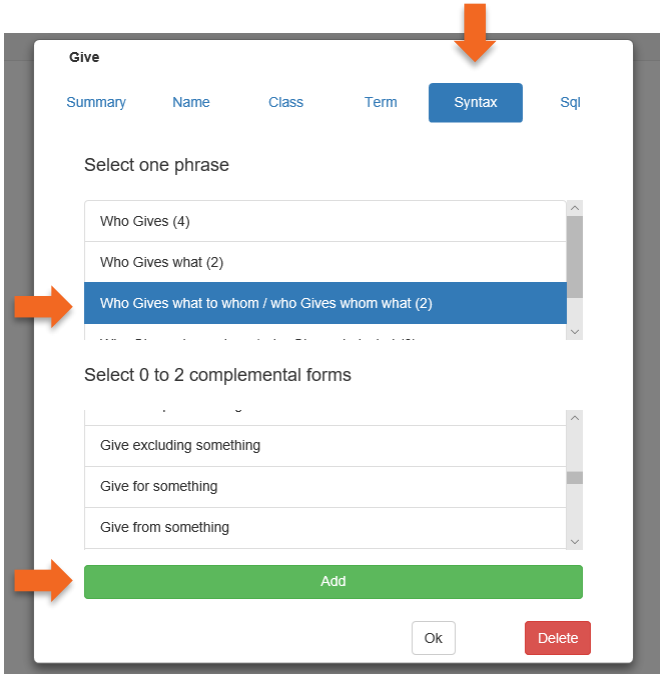
Press *Add term grammar*.



Create a verb entity

Defining syntax

For all verbs a verb complement needs to be defined, describing how the verb relates to other entities. Select one alternative and press *Add*.



DialogueTech

Create a verb entity

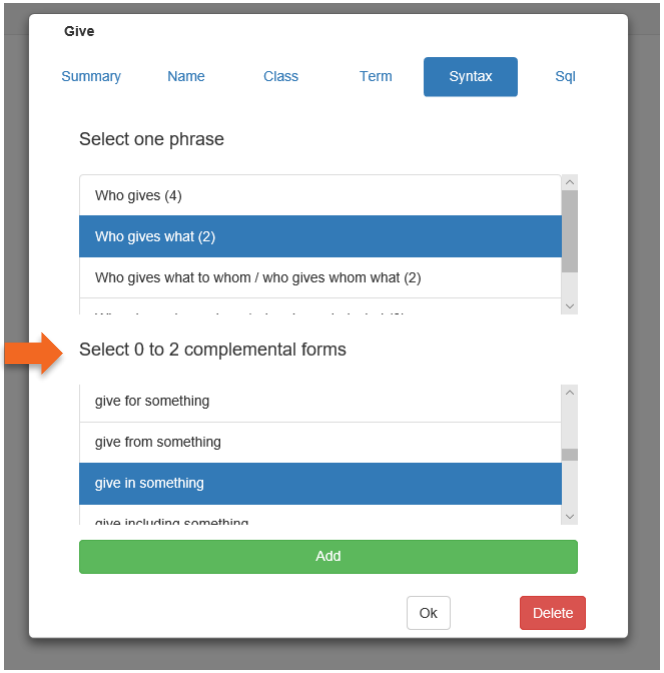
select prepositional complement (optional)

For all verbs a prepositional complement can be defined specifying which prepositional complements which are made available to the verb.

Note that spatial and temporal complements does not need to be defined. These types of relationships will be available anyway, e.g.:

What courses do Uppsala University give in English during the spring semester?

Select 0-2 alternatives and press *Add* ➔ *OK*.



you do not specify any SQL for verbs 1(2)

Currently there is no support for specifying SQL for verbs. This may cause additional work. Consider e.g.

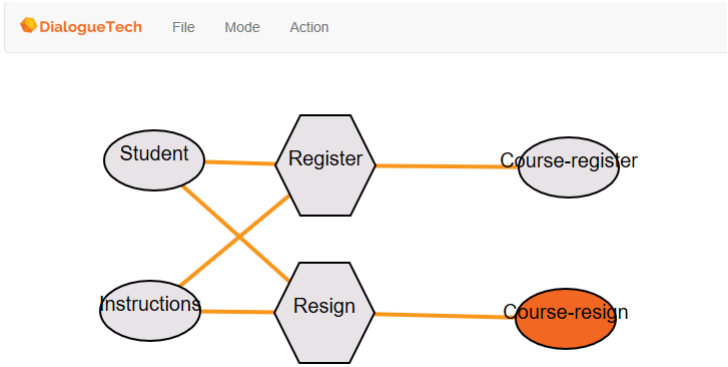
How do I register for a course?

And

How do I resign from a course?

which differ only in the verb.

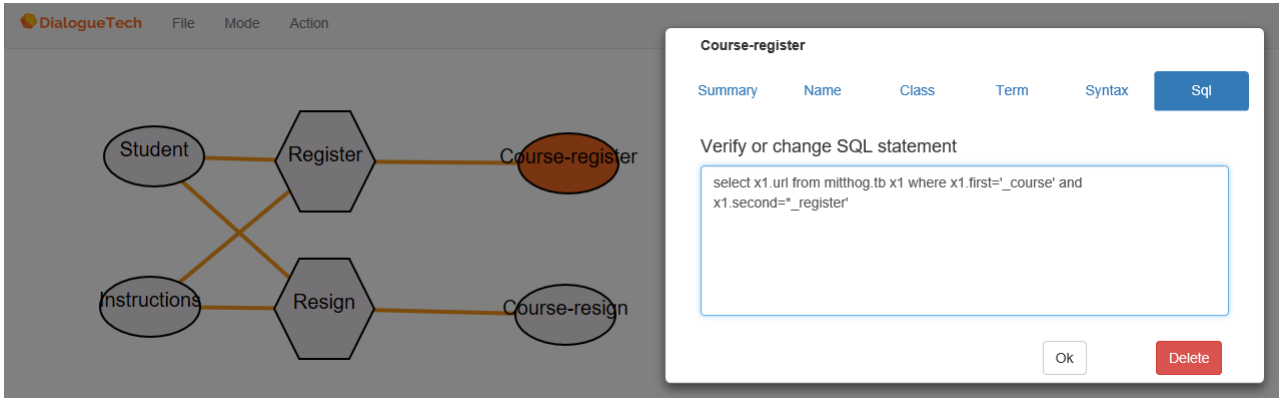
To solve this you define another noun entity which is uniquely connected to the verb.



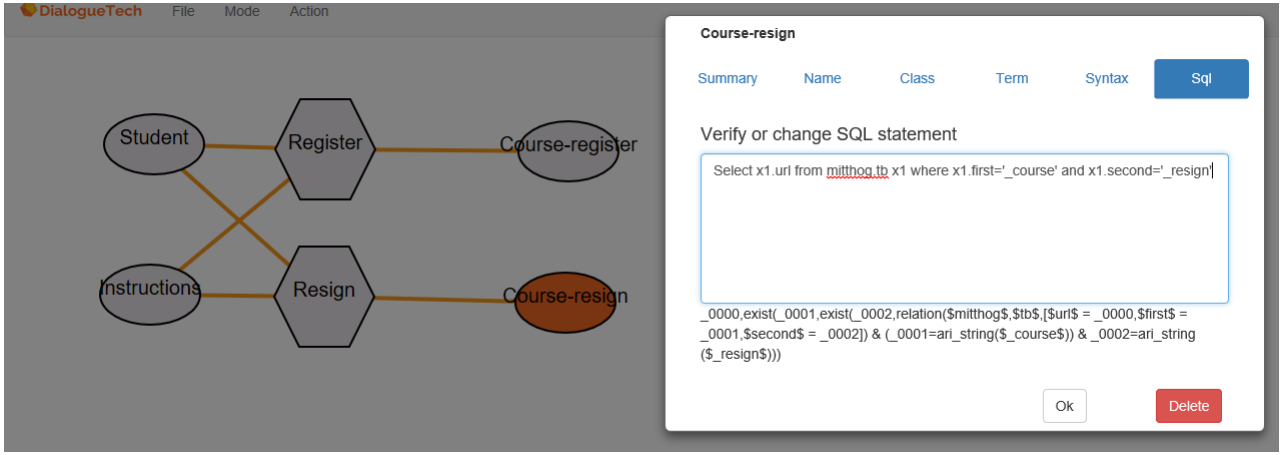
DialogueTech Create a verb entity

you do not specify any SQL for verbs 2(2)

The entity COURSE-REGISTER has the SQL for the verb entity REGISTER and contains the term course.



The entity COURSE-RESIGN has the SQL for the verb entity RESIGN and contains the term course.



If you ask: *What is a course?* two SQL queries will be generated capturing:

- the concept of registering on a course *and*
- the concept of resigning.

Create an adjective entity

Ny sida

naming

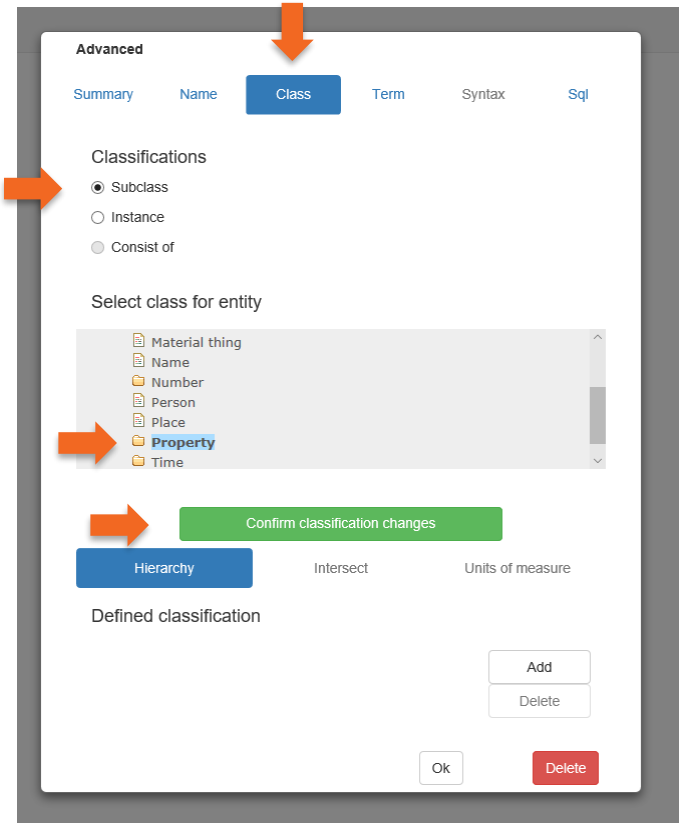
Same procedure as noun

- *Mode* ➔ *Lang*
- *Action* ➔ *Add entity*
- *Enter name* ➔ *OK*

classification

Same procedure as noun

- Double-click on an entity
- Press *Class*, *Subclass* and *Add*
- Open the scroll down menu by pressing *Thing*.
- Classify the adjective entity e.g. as Property. Press *Confirm classification changes*.



Bättre förslag??

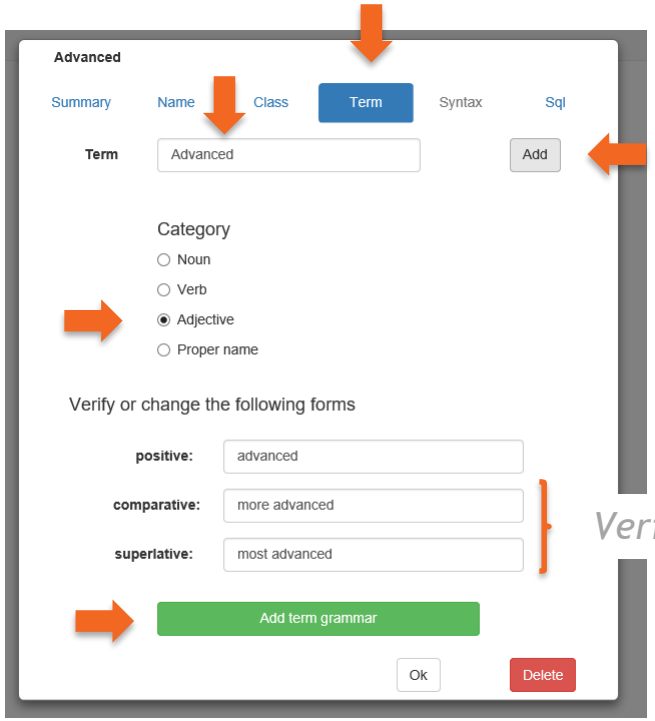
Create an adjective entity

defining terms

Press *Term* and enter the terms which will be used for the entity – press *Add*. The *Customizer* proposes comparative and superlative forms for the term selected. Edit if necessary.

Repeat if you want to add more terms.

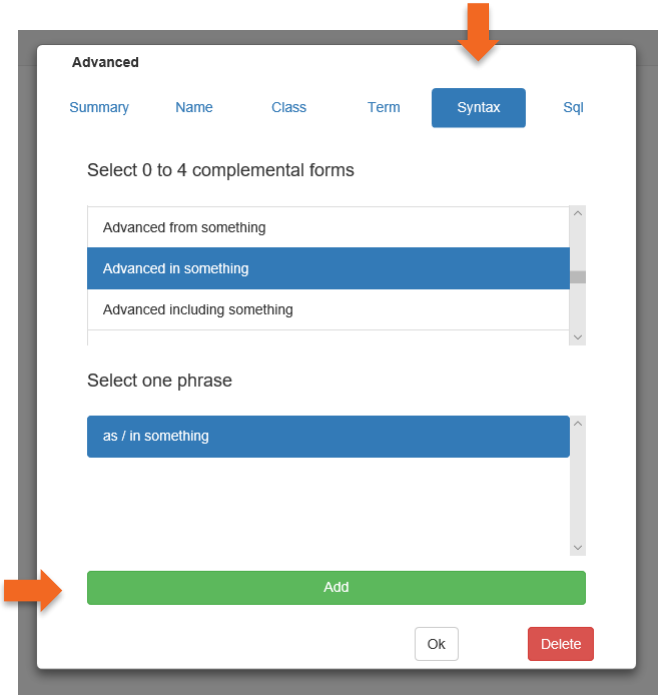
Finally press *Add term grammar*.



Create an adjective entity

defining syntax

Lämplig text



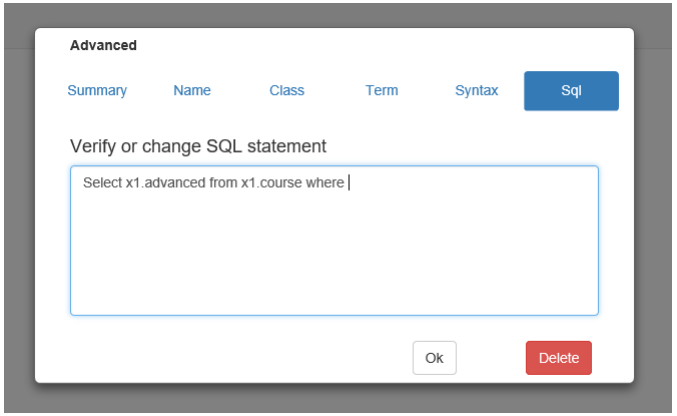
DialogueTech

Create an adjective entity

defining SQL

Lämplig text

Bättre exempel



DialogueTech

Create an adjective entity – option 1, mandatory part of a noun

There are two ways to model adjectives which are unique to a particular noun:

1. Like a mandatory part of a noun.
2. Like a subclass to a noun.

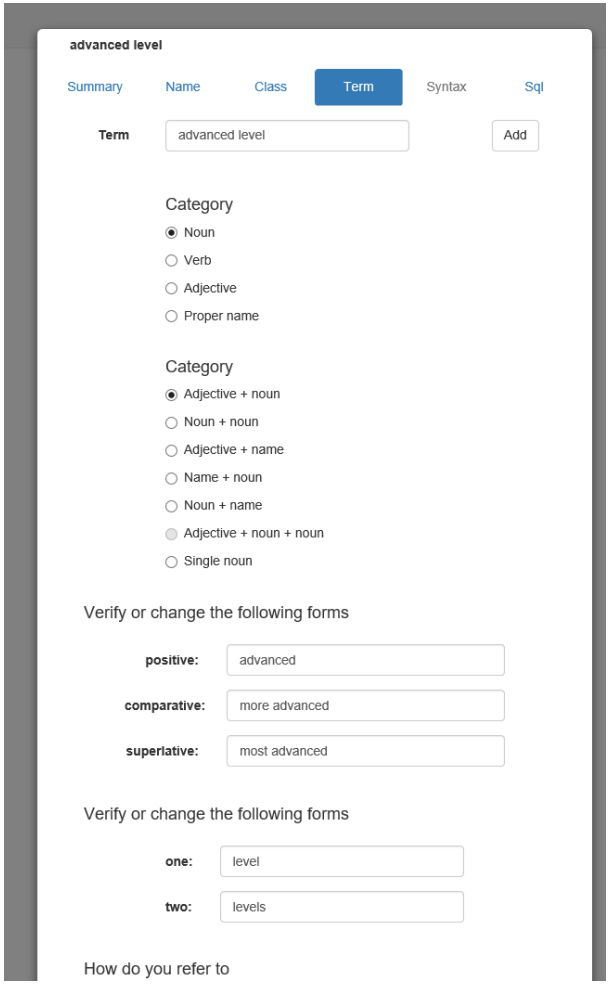
When the adjective is made a mandatory part of an noun, e.g. *advanced level*, the grammar will make sure that it is part of the term, e.g.:

What courses are on an advanced level?

will work, whereas:

What courses are on a level?

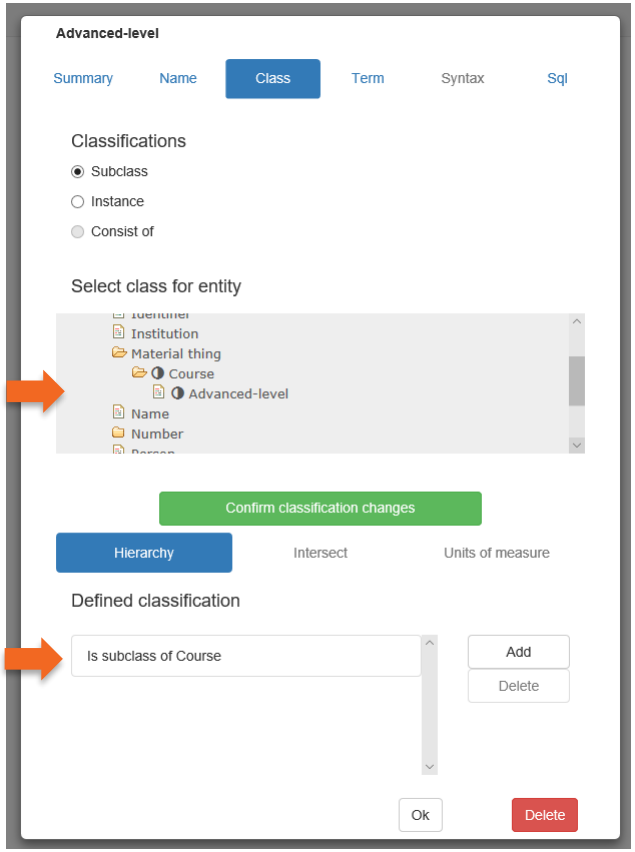
will not work.



DialogueTech

Create an adjective entity – option 2, subclass to a noun

Define an adjective as a subclass to a noun. You do this under the *Class* option.



DialogueTech

Create an adjective entity – option 2, subclass to a noun 2(2)

Select *Term* and define the term(s) you want to use. Select *Category* and select the Adjective option. Take away comparative forms.

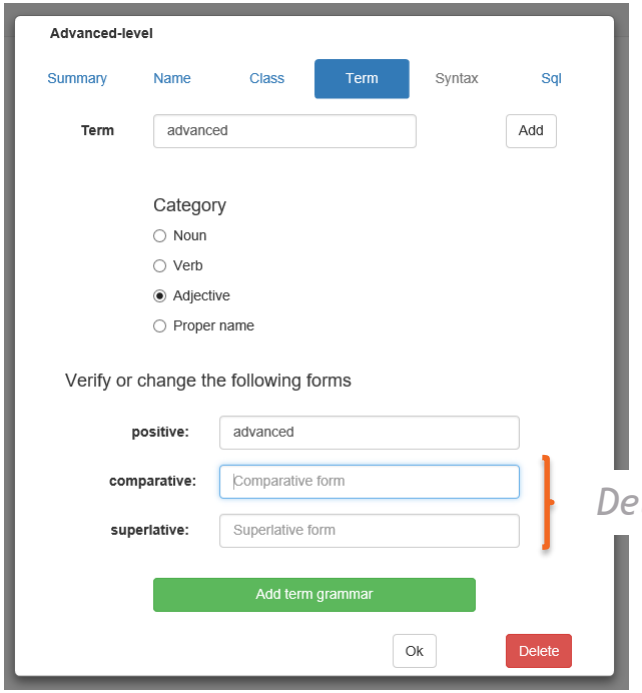
Unlike option 1 both:

Are there any advanced courses?

And

Are there any courses?

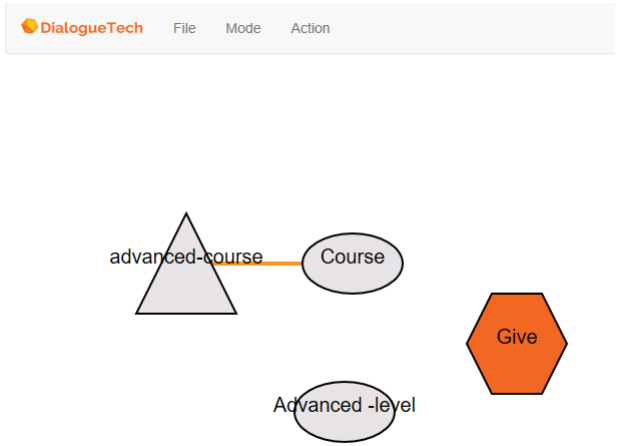
will now work.



DialogueTech

Create an adjective entity – done!

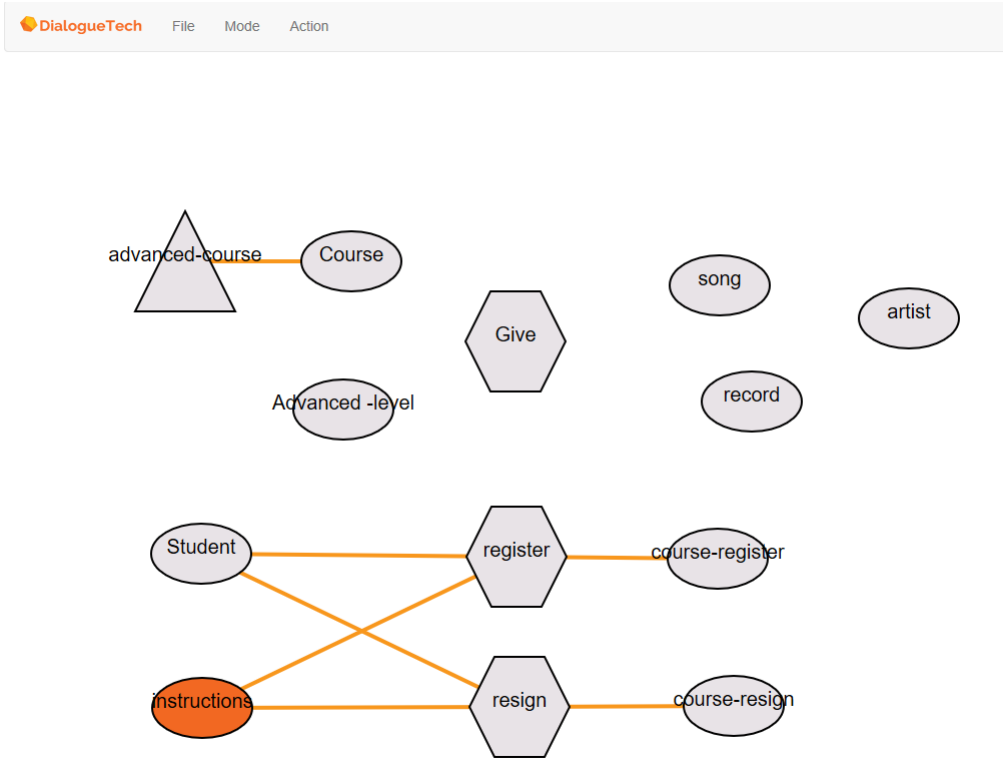
Adjective have triangular symbols. In the example option 1 is represented by the ADVANCED-LEVEL entity and option 2 by the adjective entity ADVANCED-COURSE. In option 1 the compound term *advanced level* is defined for the noun entity ADVANCED-LEVEL.



DialogueTech

Create an entity – repeat the process until all entities have been defined

Using the list of entities from the first phase this process is repeated until all entities have been defined.

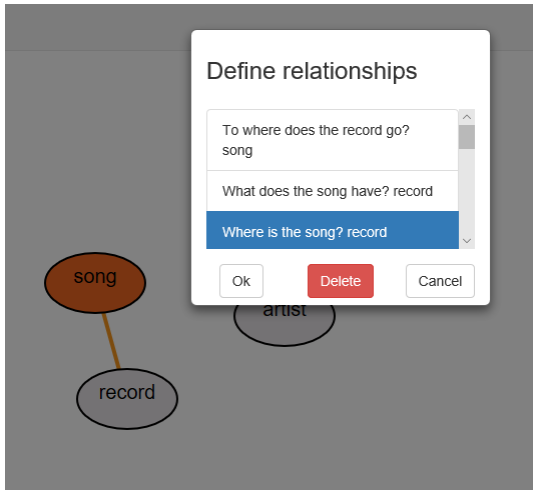


DialogueTech

Define relationships

Using the entity-relationship model from the first phase start defining relationships between entities. Drag and drop the entities you are working with on top of each other and select one or several relationships suggested by the *Customizer*.

Click *OK* and continue until all relationships have been defined.



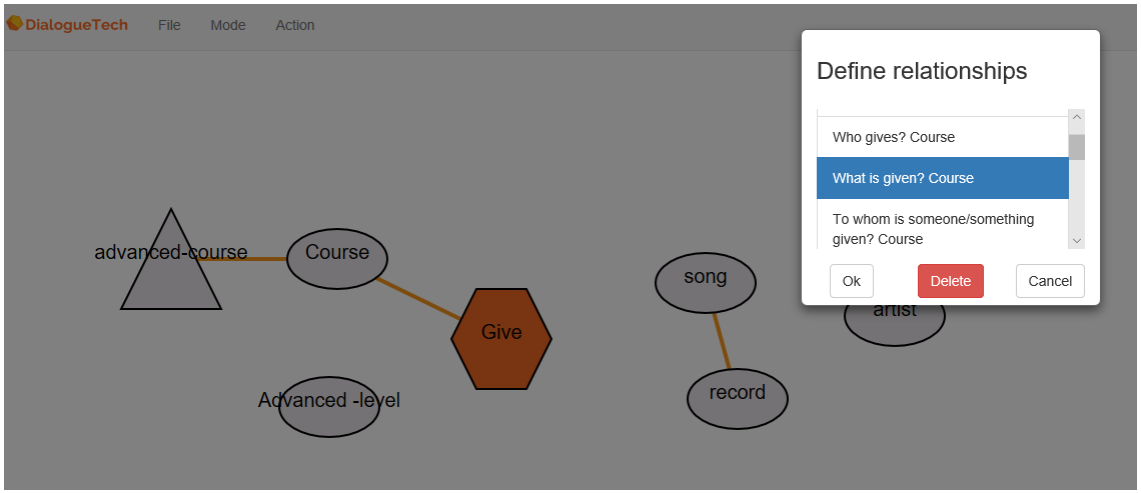
Verb relationships – What relationship (direct object)

The relationships are invoked by dragging (right-click) and dropping a verb entity onto a noun entity. A number of relationships are suggested.

Select the one you want and click *OK*.

The *What* relationship is one of the most frequently used relationships.

A relationship between two entities is illustrated by a line. If you double-click the line the relationship window appears.



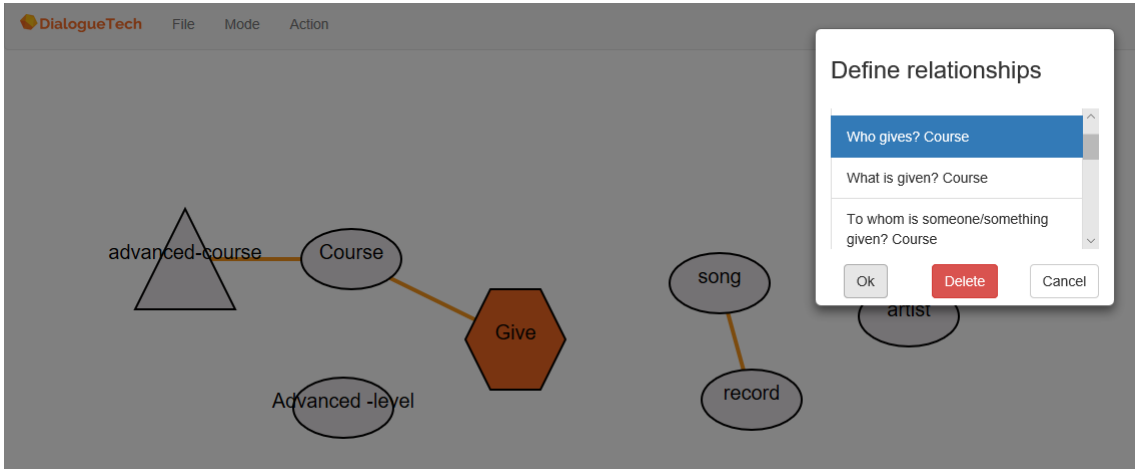
DialogueTech

Verb relationships – Who relationship

The *Who* relationship allow you to ask questions like:

- *Who gives courses?*
- *Which courses does Uppsala University give?*

Etc.



DialogueTech

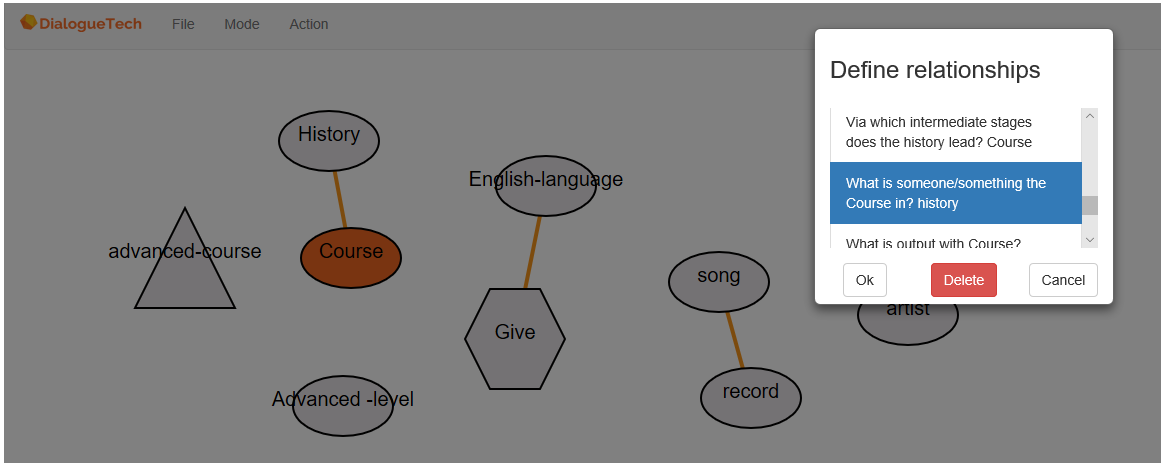
Verb relationships – Prepositional relationships

Selecting a prepositional relationship, such as

What does someone/something give?,

allows you to ask questions like:

Are any courses given in English?



DialogueTech

Verb relationships – Other relationships

Causal:	Why	***Why*** *do Uppsala University give History A this year?*
Instructive:	How	***How*** *can I apply to History A?*
Temporal:	When	***When*** *do the spring semester start?*
		What courses are given ***in*** *the spring semester?*
	Until when	*How many courses run* ***to*** *December 22?*
	Since when	***From*** *which date can I apply to History A?*
		Can I apply to History A ***from*** *October 15?*
Spatial:	Where	***Where*** *is Uppsala University located?*
		What kind of accommodation is there ***in*** *Härnösand*
	From where	*What is the distance* ***from*** *Stockholm?*
	To where	*What is the distance* ***to*** *Härnösand from Stockholm*

Via which intermediate stages...

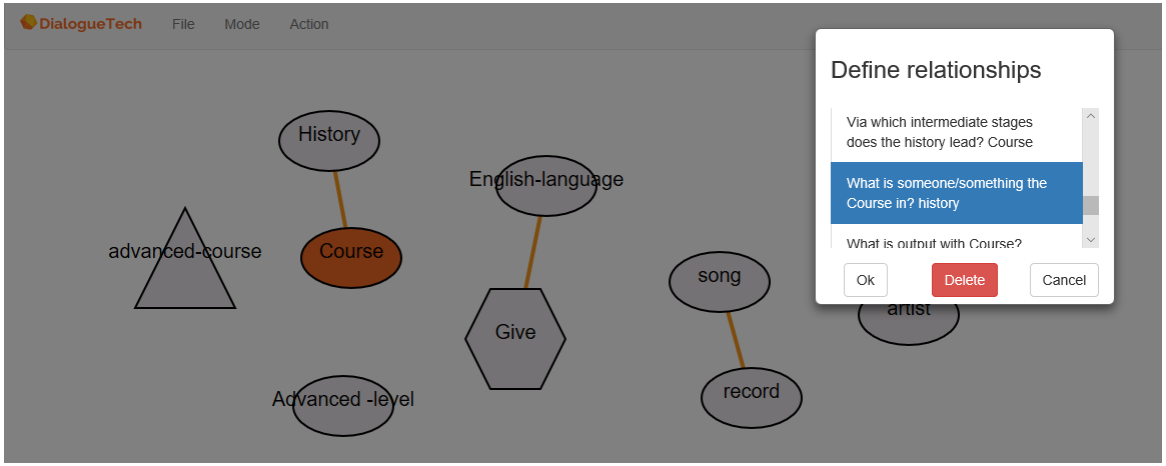
Can I apply to History A ***through*** *the ordinary application form*

DialogueTech

Noun relationships – Prepositional relationships

Assume that you have defined two noun relationships, COURSE and HISTORY. Selecting a prepositional relationship allows you to ask questions like:

Is there a course in History A?



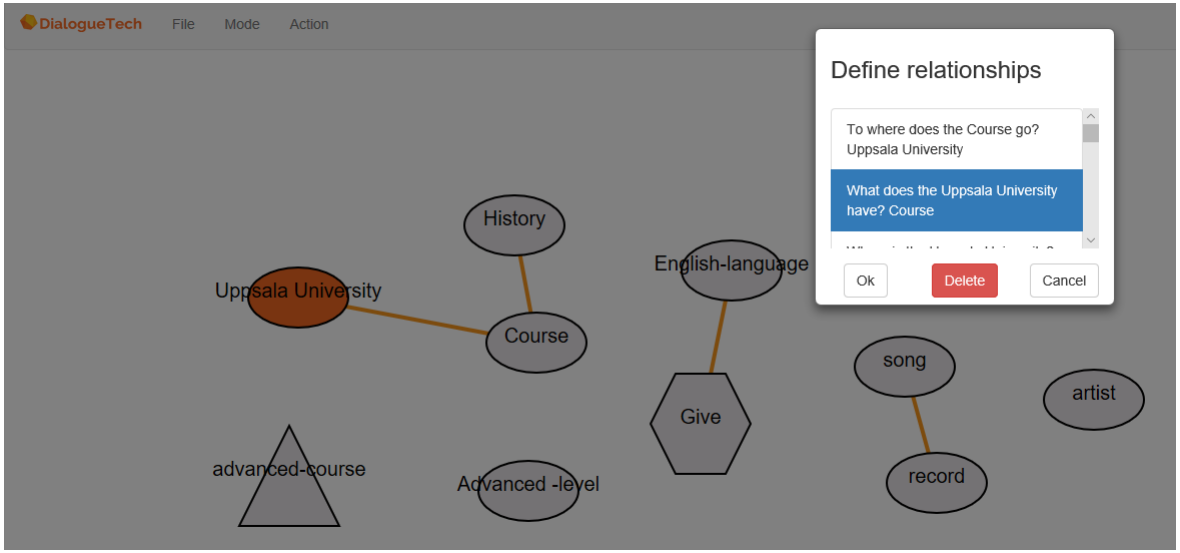
Noun relationships – Possessive relationships

By selecting e.g. the possessive relationship

What does the Uppsala University have?

you can ask questions like

- *What courses do Uppsala University have?*
- *Show me Uppsala University courses?*



DialogueTech

Noun relationships – Temporal and spatial relationships

Like in the case of verb entities noun entities have spatial and temporal relationships. Temporal relationships are only available if one of the entities has been categorized as *Time*.

Spatial	*Where is the Uppsala University?*
	To where is the Uppsala University?
	From where does the Uppsala University come?
	To where does the Uppsala University go?
Temporal	*When is the Uppsala University?*
	Since when is the Uppsala University?
	Till when is the Uppsala University?

DialogueTech

Quantifiable properties

Ergo supports the following quantifiable properties

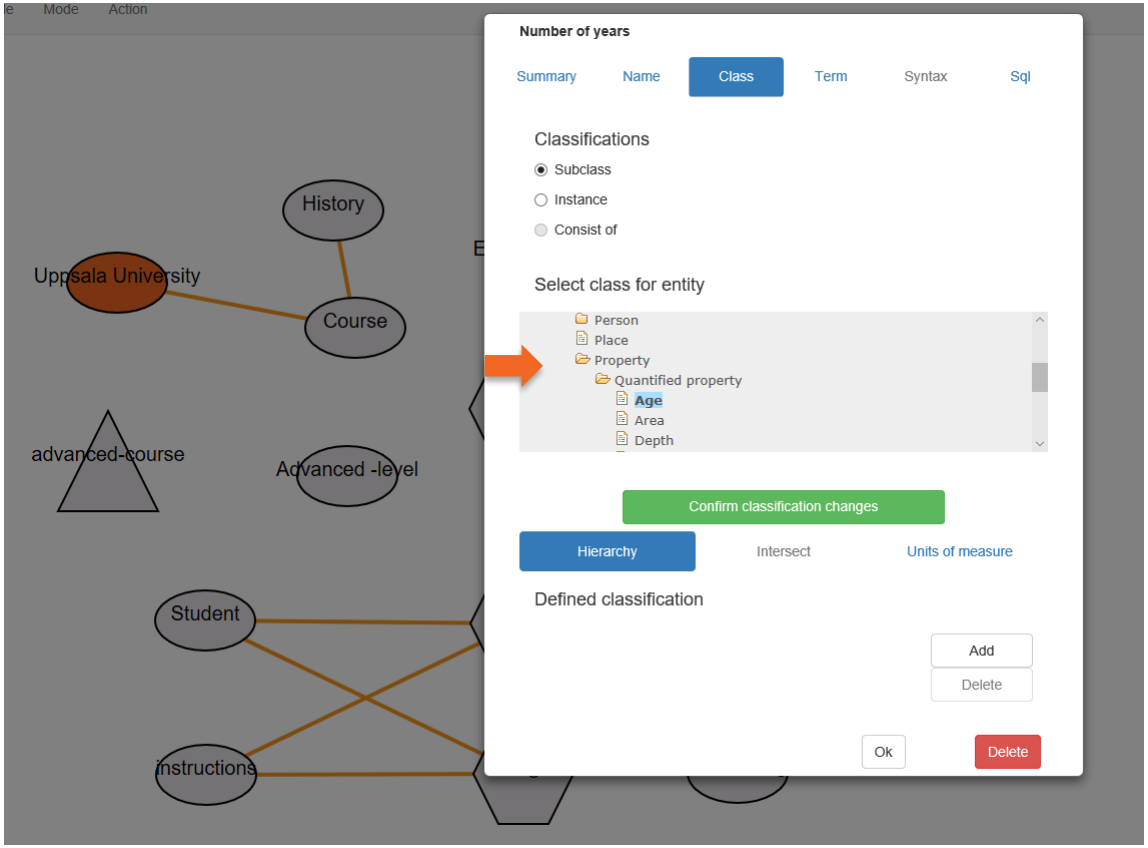
- -Age
- -Area
- -Depth
- -Duration
- -Height
- -Length
- -Price
- -Size
- -Value
- -Volume
- -Weight
- -Width

You select quantifiable property when you classify the entity.

DialogueTech

Quantifiable properties - classification

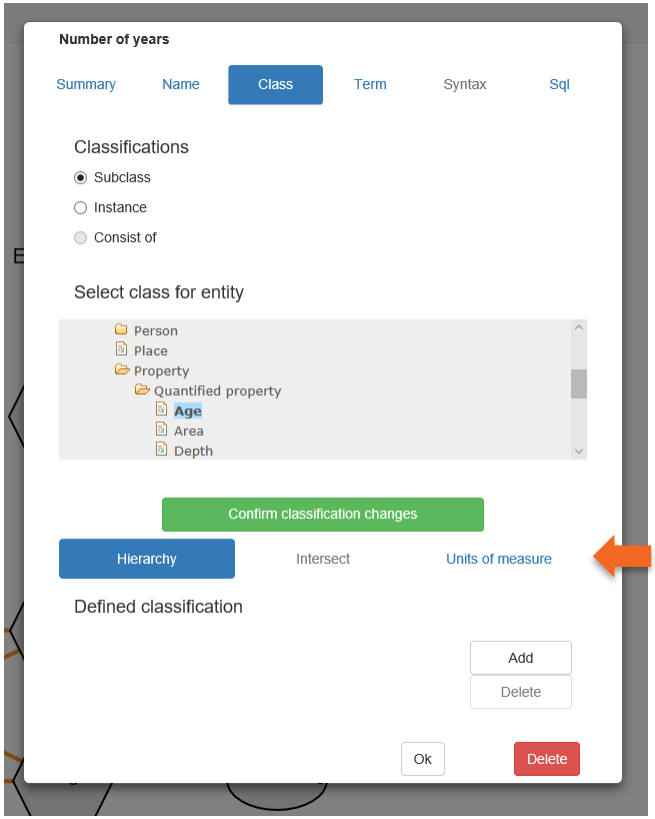
When you define a quantifiable entity, like age, you classify it as a *Property* ➔ *Quantified_property* ➔ *Age* under the *Class* tab.

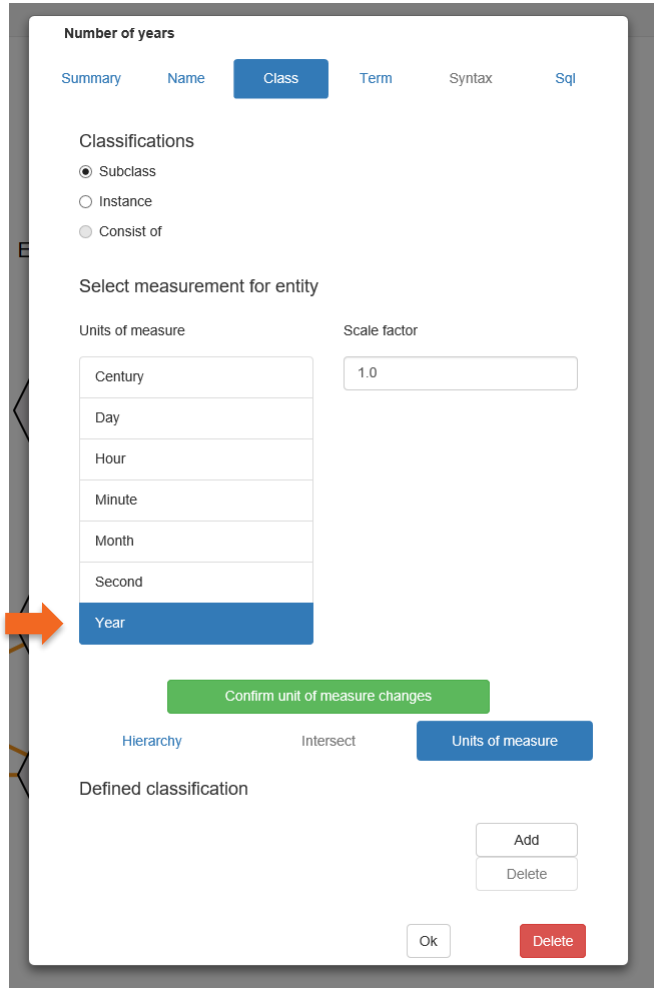


DialogueTech

Quantifiable properties – unit of measure

Press the tab *Unit of measure* and select the proper unit of measure, e.g. *Year* in the case of age.





Quantifiable properties – relationships

After having defined term(s) and SQL you can drag and drop an entity which is classified as a quantified property onto a noun entity. Selecting relationships like

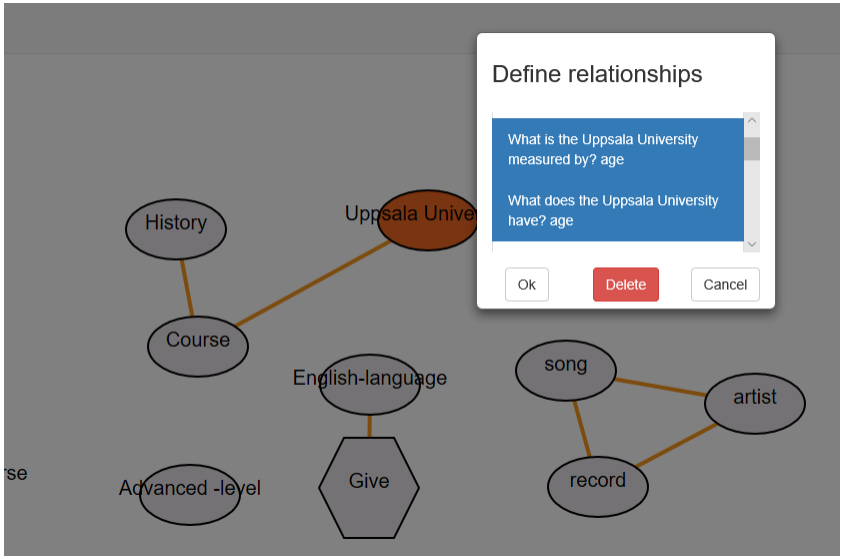
What is Uppsala University measured by?

and

What does Uppsala University have?

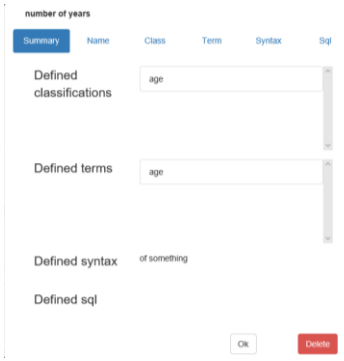
allows you to ask questions like

- *How old is Uppsala University?*
- *What is the age of Uppsala University?*



Adding a new entity

- "*number of years*"
- classified as *Age*
- term *age*
- syntax *age of something*

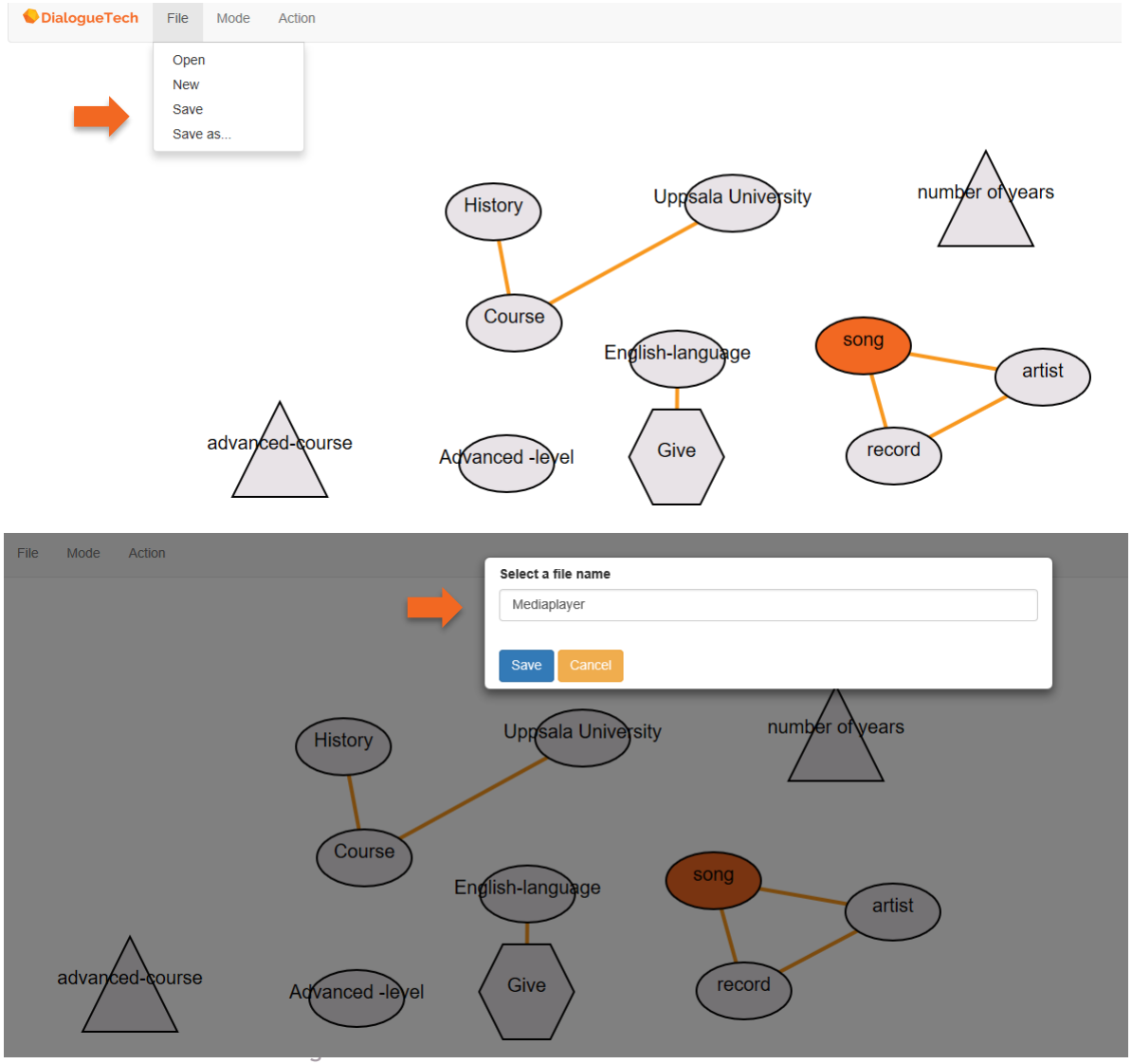


DialogueTech

Done - Save the Domain Model

When all relationships in the entity-relationship model have been defined you save the result in a file. Press *File Save* or *Save as…* to open the *Select a file name window* – from there you can save.

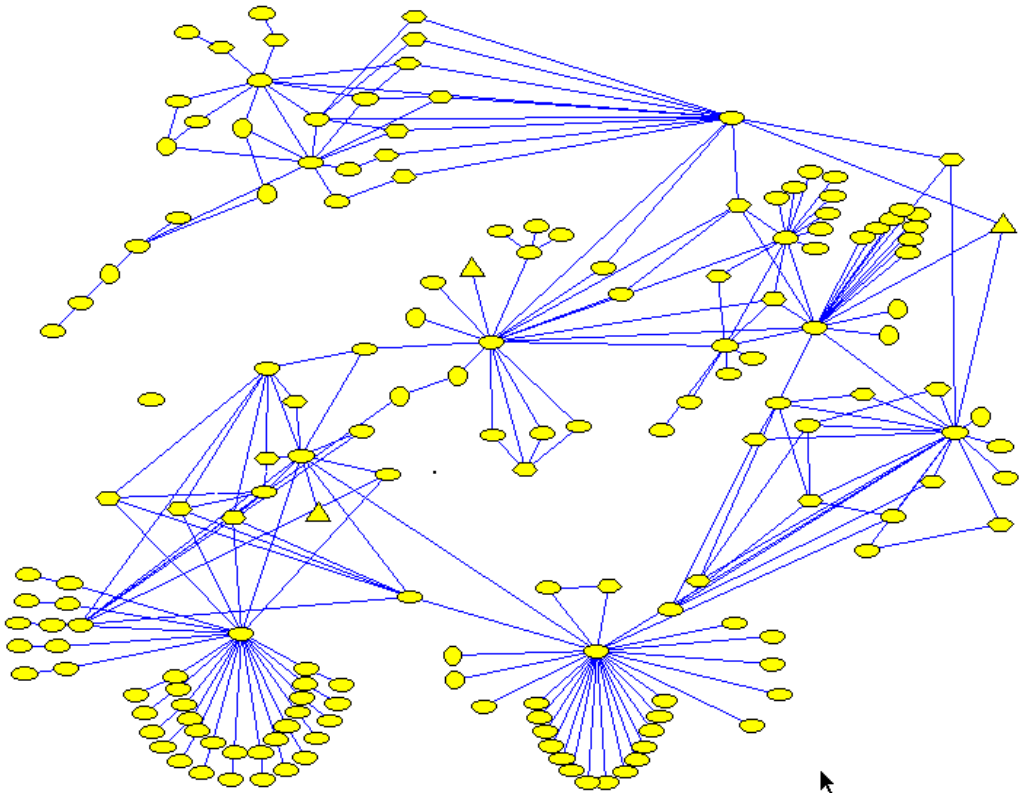
This file is the output of the *Customizer* .



DialogueTech

A typical Domain Model

A typical domain model contains anything from a few tens of entities to a few hundred entities.



DialogueTech

Known bugs

Problem: Some older operating systems may cause problems in the Customization Tool with special characters such as åäö.

Solution: Model these using another unique ascii-combination. Before the model is converted to an .exe file this combination is replaced by the original characters.

Problem: If you have modelled a complex term using the combination adjective + noun you can not attach a prepositional phrase.

Example
Works: *advanced courses*
Does nor work: *advanced courses on Uppsala University*
Solution: Model *advanced courses* as a single noun

DialogueTech

Limitations

- Entity – SQL are in a one-to-one relationship
- Each entity can have 1-n terms associated with it which must:
  have the same category (Noun, Verb, Adjective)
  have the same syntax. Ergo does not perform any error check for this.
- Each term can be used in an arbitrary number of entities. For queries which only refer to several noun entities a new interpretation is added for each new entity where the term is used. As an illustration if the term *course* is used in five entities the query *What courses are there?* Will be given five interpretations of which the first one will be used.
- Currently SQL is not supported for verb entities.
- Do not use the category Proper name. If you want to define a proper name categorize it as a noun and erase the plural form.

DialogueTech

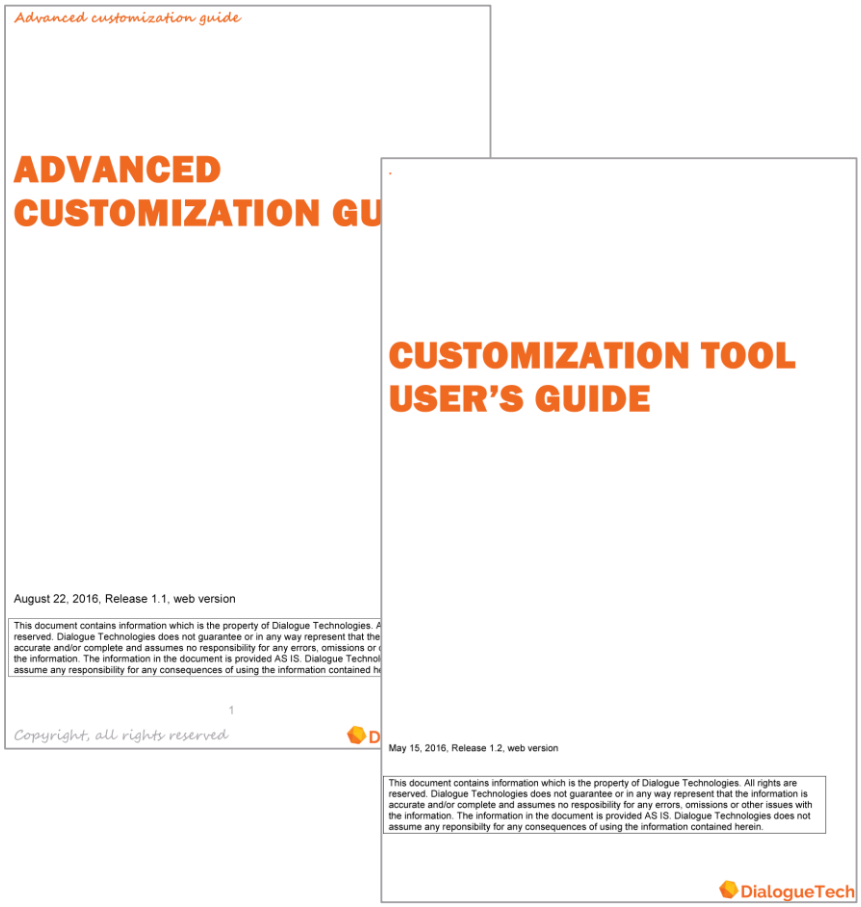
Development tools and documentation

Documentation:

- Advanced Customization Guide
- Customizer Tool Users Guide
- Customization Process Overview

Tools:

- Customizer



DialogueTech

Table of contents

DialogueTech

Verify the model

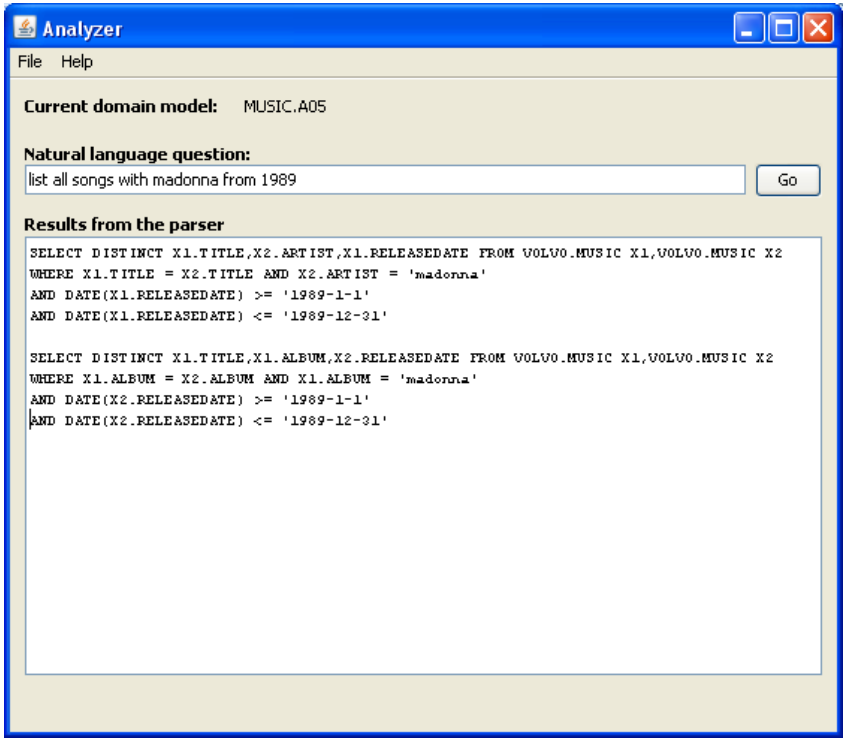
Start the test program *Analyzer*.

Press *File* and select the file to test. Use the test questions defined in stage 1 and verify that the resulting SQL phrases are correct.

Note how the fragments of SQL queries defined in stage 2, e.g.
SELECT X1 SONG FROM VOLVO.MUSIC X1,
are used to build complex SQL queries like

```
SELECT DISTINCT X1.TITLE,X2.ARTIST,X1.RELEASEDATE FROM
VOLVO.MUSIC X1,VOLVO.MUSIC X2
WHERE X1.TITLE = X2.TITLE AND X2.ARTIST = 'madonna'
AND DATE(X1.RELEASEDATE) >= '1989-1-1'
AND DATE(X1.RELEASEDATE) <= '1989-12-31'

SELECT DISTINCT X1.TITLE,X1.ALBUM,X2.RELEASEDATE FROM
VOLVO.MUSIC X1,VOLVO.MUSIC X2
WHERE X1.ALBUM = X2.ALBUM AND X1.ALBUM = 'madonna'
AND DATE(X2.RELEASEDATE) >= '1989-1-1'
AND DATE(X2.RELEASEDATE) <= '1989-12-31'
```

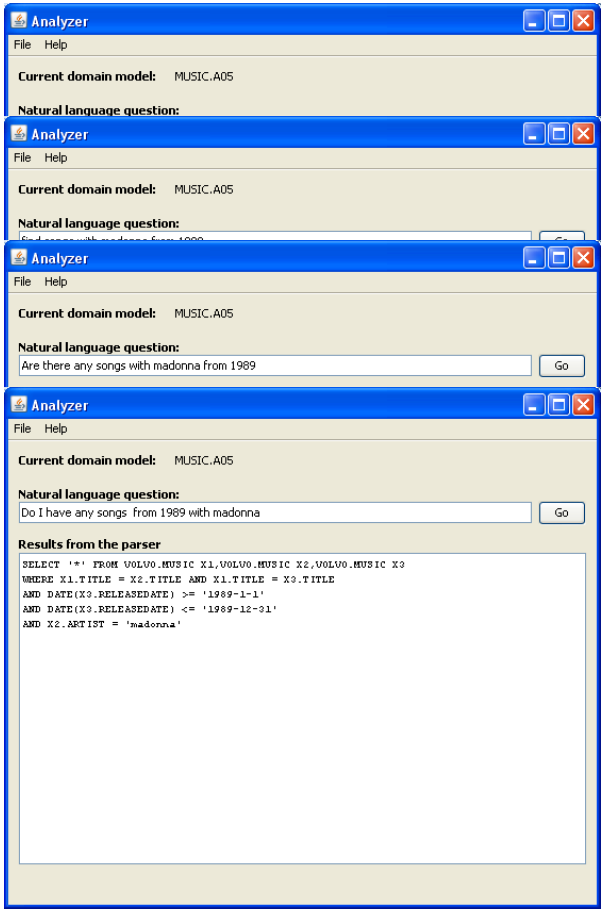


DialogueTech

Verify various formulations

The language engine is designed to be able to analyze a multitude of various ways of asking a question by defining the relationships between entities. Verify that several ways of asking the same question retrieves the same information from the database, e.g.

List all songs with madonna from 1989
Find songs with Madonna from 1989
Are there any songs with madonna from 1989
Do I have any songs from 1989 with Madonna
etc.



DialogueTech

Done!

When the domain model has been tested the process of building a domain model is completed. To complete the development you:

1. Compile the domain model from the analyzer together with the grammar for the language in question.

2. Integrate in the target environment

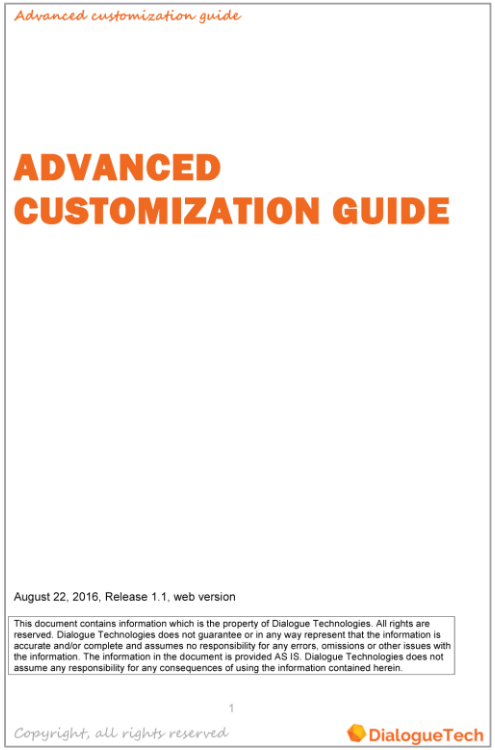
3. Test on target.

DialogueTech

Development tools and documentation

Documentation:
- Advanced Customization Guide
- Customization Process Overview

Tools:
- Analyzer
- Standard development tools, e.g. .net, java, Android Studio, etc.


Advanced customization guide

ADVANCED CUSTOMIZATION GUIDE

August 22, 2016, Release 1.1, web version

This document contains information which is the property of Dialogue Technologies. All rights are reserved. Dialogue Technologies does not guarantee or in any way represent that the information is accurate and/or complete and assumes no responsibility for any errors, omissions or other issues with the information. The information in the document is provided AS IS. Dialogue Technologies does not assume any responsibility for any consequences of using the information contained herein.

1

Copyright, all rights reserved DialogueTech