	Project: GallitoAPI Contact: info@semantialab.es	Version: 2	Date: 04-01-2014
semantia LAB	Name: generalConceptsGallitoAPI_v3.docx	Document Type: User Manual	

GallitoAPI

General concepts

Information about this document

This document describes the protocol to be followed to prepare and load **GallitoAPI** with a thematic domain and its use in third-party applications. It also mentions what **GallitoAPI** is and how it works, and in what scenarios it might be used. In the last part of the document, its most frequent functions and applications are described.

Introduction

An API that receives and analyses texts GallitoAPI is an application that receives texts and analyses them in various ways. Its processes are mainly based on probabilistic techniques (n-dimensional spaces, stochastic models of language, information theory techniques, etc.), although it also has rule-based mechanisms, more specifically, deterministic grammars and some simple-search mechanisms. This document also describes its functionalities in depth.

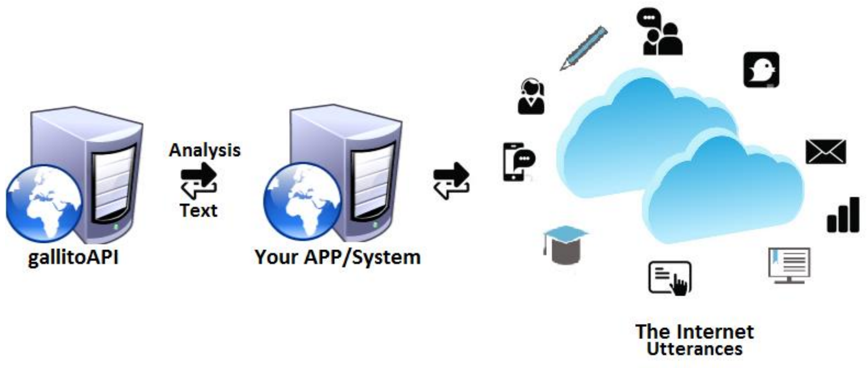
It is idiosyncratic. Even though **GallitoAPI** has pre-built generalist domains that can work efficiently, ideally, you should adapt it to your domain. In this way, **GallitoAPI's** analysis will have greater control and apply the restrictions of the thematic domain in question.

It does not consume too much implementation time. You don't need a lot of time or human resources to get **GallitoAPI** ready to work. You don't need to define domain ontologies or even label texts. You just need to know which theme is to be dealt with, check whether there is a pre-built domain about it, and, if not, provide a text sample to train using **Gallito Studio**. **Gallito Studio**'s training output will be used by **GallitoAPI** in its analysis.

EASY to use!

- Just a few steps.
- Progresive and incremental deployments.
- Running soon, better then.

Place it in your back-end. We should also say that **GallitoAPI** is basically a set of online services hosted in an Internet Information Server (IIS), ready to answer third-party system requests fast. When installed, it can serve as a cloud for third-party applications, or work as just another application in your ecosystem (e.g. a Data Base server). **GallitoAPI** will receive and analyse texts, with no interface other than those proper to the http protocol (SOAP), and will be invisible to third-party systems. Installation is easy, and just requires an IIS server. This software is included in all the Windows operative systems (https://www.iis.net/learn)



Create an API farm GallitoAPI can also be cloned, creating a GallitoAPI farm, in such a way that, for example, each API has a specific domain or specific settings, in order to cover different tasks and requirements. The IIS Server should simply be configured so that each **GallitoAPI** can work as an independent application (see the Cloning **GallitoAPI** section).

Potential scenarios It is hard to list the scenarios in which semantic metrics can be used by third-party systems. In general, it could be said that any system that requires semantic heuristics could use them.

A. Let's imagine **a robot that explores URLs** to compile website information (**crawler**). This robot could perform a more efficient search and not "get lost" if it used heuristics based on semantic metrics. In this way, the crawler would check in advance the best steps to follow and retrieve the contents pertaining to a given domain.

B. In the same we, you might want to classify, summarise, and highlight the main concepts in **the input from customers of a specific service (post, calls, claims, complaints, etc**.) and save them in your own database in a vectorial format, in order to perform efficient searches on this basis.

C. You could also classify **social media texts**, such as tweets. Because **GallitoAPI** and its techniques are low-cost, you can train corpora on a regular basis to include any new meanings.

D. You can also use it to create (extract) semantic neighbourhoods that help, simplify, or suggest ways to **navigate through a website**.

E. You can obtain **many auxiliary lexical indices**; the most useful ones would, for example, be related to the **significance** and **specificity** of words. The former provides information about the degree of significance of a given word in the thematic space being used, while the latter shows how specific or restricted it is, that is, its degree or level of "ambiguity" and how reliable its prediction is.

F. It could also **automatically evaluate the quality of the content of academic texts** (discursive texts) produced by students (open answers, reading summaries, short answers, etc.), and give numerical scores for certain key aspects of the evaluations.

G. You might also wish to evaluate the degree of **coherence**, use of **vocabulary**, informative force of words, and word frequency in the texts, and even their degree of adjustment to the target audience's ages, etc.

H. Likewise, you might be interested in **correcting** the hypothetical interpretations extracted by **voice recognition software (ASR) or OCR systems**, in order to include a new decision layer that weights semantic coherence, in addition to the **phonetic models and the n-gram models (SLMs).**

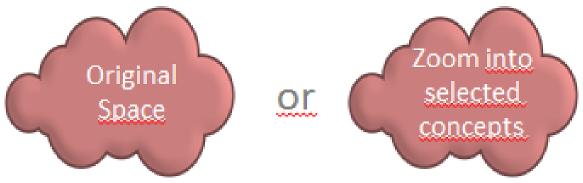
Text sample

(Collecting texts)

We have said that a text sample must be compiled and a thematic space must be generated to operate **GallitoAPI**, but… how are these thematic spaces which will be later loaded in **GallitoAPI** generated?

First of all, you must compile text samples in a text corpus (see the Gallito Studio 2.0 manual for the details of the corpus format). These can be text samples taken from the Internet, transcriptions from a contact centre, messages in forums, textbooks etc. You can even use applications that specialise in taking texts from the Internet and creating corpora, such as IMPORT.IO (https://import.io/). The better the sample, the better the thematic space on which the app functionalities are based will be. You can also label the samples, even though this is not strictly mandatory. This flexibility means that the project to be developed can be implemented early on and improved by monitoring its operation.

This domain can also have a general nature, that is, it need not represent a specific theme, but be able to assess any domain, even though its specificity levels will decrease and the ambiguity of some of the words represented will increase. Finally, it should be pointed out that generalist spaces are incredibly useful to classify social media texts and topics, for example. ¡You are the one who knows best how to decide which domain you need!.

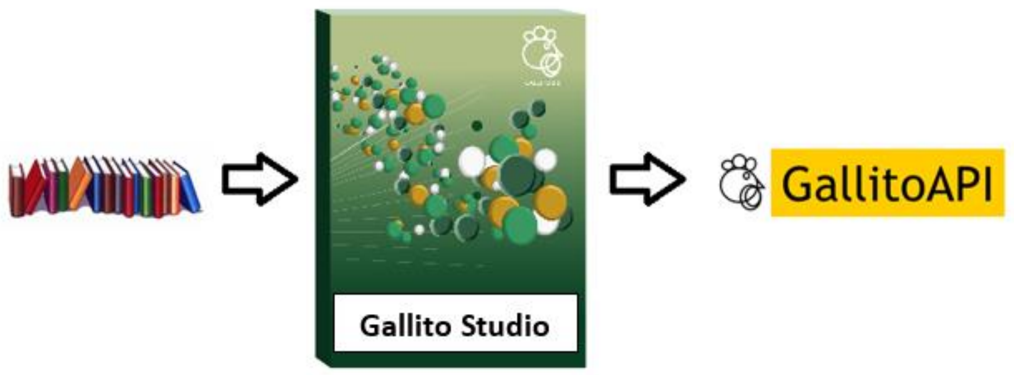


You can also create your own lemmatised samples, where words are grouped following specific criteria of interest for your activity or business line (e.g. grouping all telephone companies under a single label). The tool that processes text samples is **Gallito Studio,** which admits a huge range of processing settings.

Thematic space of reference

(Processing text using Gallito Studio)

Once you have the samples, you can process them using Gallito Studio in order to obtain a **thematic space** as output (see Gallito Studio 2.0 manual). This processing task can take minutes, hours, or sometimes even days, as it depends on the size of the sample (corpus) and the process settings. Only some of the files resulting from the text processing carried out should be set apart so that they can be loaded in GallitoAPI, as almost all the **GallitoAPI** linguistic analysis functions need to use them.



Gallito Studio can also be used to create a **labelled space**. A labelled space is a thematic space in which some of the coordinates are defined by concepts previously chosen on the basis of various interests. This alternative makes it possible to focus the analysis on those concepts to then "judge" any word or text on that basis. To have this possibility, you should follow a simple procedure, integrated in **Gallito Studio**, which includes the adjustment of the final model, in such a way that its operation and outcome are reliable (see change of base manual).

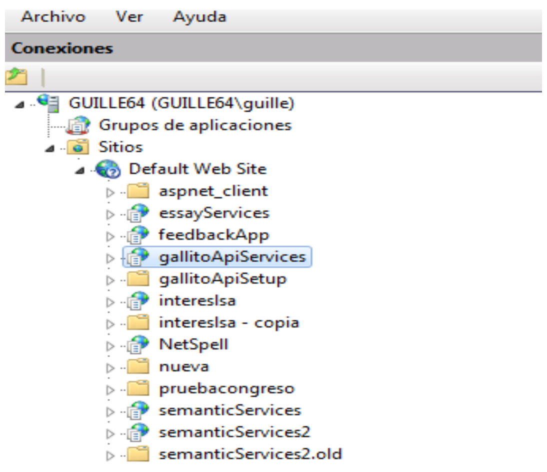
Some of **GallitoAPI**'s functions use **labelled spaces**, even though the possibility of creating an initial labelled space to be used by all functions is also envisaged. Another work scenario is based on cloning two **GallitoAPIs** in a server, so that one of them works with a **labelled space**, and the other one with an **ordinary space** (note that **GallitoAPI** cloning is unlimited: see the Cloning **GallitoAPI** section).

Once the thematic space has been created, you just have to "place" it in a **GallitoAPI**. A thematic space is a set of files with the ".bnl" extension that must be kept in a safe place to be used whenever required (see the important files document).

Using GallitoAPI

(Soldiers, en garde)

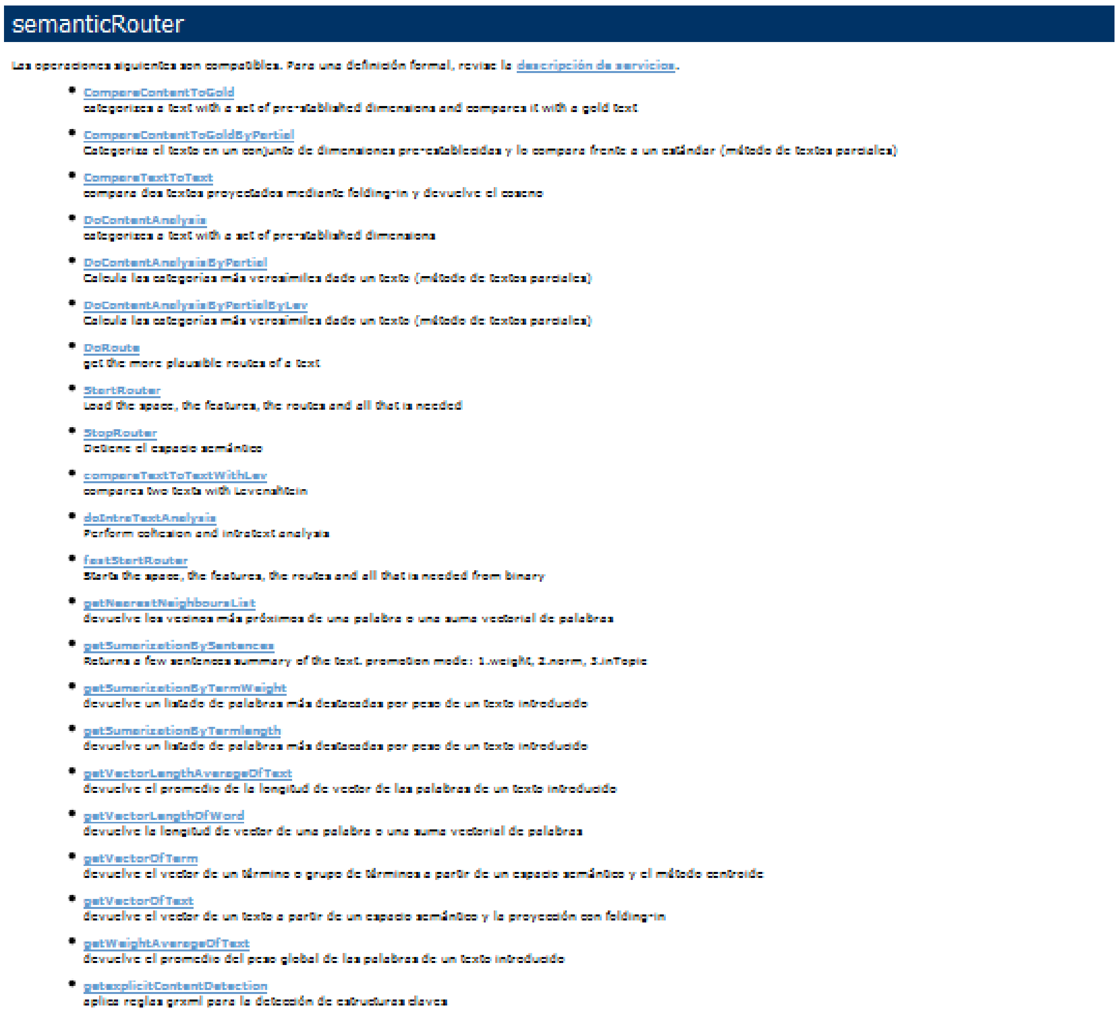
Once GallitoAPI has been installed in your machine, as well as your own licence serving program (see **GallitoAPI** installation manual), everything is ready, waiting for the thematic space to be loaded, that is, all the ".bnl" files provided by Gallito Studio. Depending on the functionality to be used, you must even use a text sample (file), (see the section Important files in **GallitoAPI**).



At this point, you need only follow these steps (see the **GallitoAPI** installation manual to find all the details):

1. Rename the Gallito Studio files (with the .bnl extension)
2. Review the configuration file (cfg.txt)
3. Invoke the service to launch it (Start Router)

Once you have done this, if no mistakes were made, you will have loaded your thematic space in the server and all the GallitoAPI functionalities will be ready.


semanticRouter

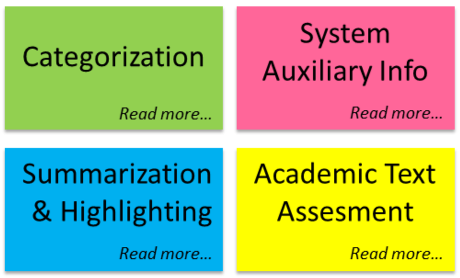
Las operaciones siguientes son compatibles. Para una definición formal, revise la descripción de servicios.

- CompareContentToGold
  categorizes a text with a set of pre-stablished dimensions and compares it with a gold text
- CompareContentToGoldByPartial
  Categoriza el texto en un conjunto de dimensiones pre-establecidas y lo compara frente a un estándar (método de textos parciales)
- CompareTextToText
  compara dos textos proyectados mediante folding-in y devuelve el coseno
- DoContentAnalysis
  categorizes a text with a set of pre-stablished dimensions
- DoContentAnalysisByPartial
  Calcula las categorías más verosímiles dado un texto (método de textos parciales)
- DoContentAnalysisByPartialByLev
  Calcula las categorías más verosímiles dado un texto (método de textos parciales)
- DoRoute
  get the more plausible routes of a text
- StartRouter
  Load the space, the features, the routes and all that is needed
- StopRouter
  Detiene el espacio semántico
- compareTextToTextWithLev
  compares two texts with Levenshtein
- doIntraTextAnalysis
  Perform cohesion and intratext analysis
- fastStartRouter
  Starts the space, the features, the routes and all that is needed from binary
- getNearestNeighboursList
  devuelve los vecinos más próximos de una palabra o una suma vectorial de palabras
- getSumarizationBySentences
  Returns a few sentences summary of the text. promotion mode: 1.weight, 2.norm, 3.inTopic
- getSumarizationByTermWeight
  devuelve un listado de palabras más destacadas por peso de un texto introducido
- getSumarizationByTermlength
  devuelve un listado de palabras más destacadas por peso de un texto introducido
- getVectorLengthAverageOfText
  devuelve el promedio de la longitud de vector de las palabras de un texto introducido
- getVectorLengthOfWord
  devuelve la longitud de vector de una palabra o una suma vectorial de palabras
- getVectorOfTerm
  devuelve el vector de un término o grupo de términos a partir de un espacio semántico y el método centroide
- getVectorOfText
  devuelve el vector de un texto a partir de un espacio semántico y la proyección con folding-in
- getWeightAverageOfText
  devuelve el promedio del peso global de las palabras de un texto introducido
- getexplicitContentDetection
  aplica reglas grxml para la detección de estructuras claves

GallitoAPI functions

(Preparations for action)

GallitoAPI's functions can be used to solve a wide range of needs or problems, each of which is aimed at a specific task. For this reason, each description of a function is accompanied by an example of a task for which it can be useful. Thus, there are functionalities pertaining to information categorisation, other functionalities are designed to evaluate academic texts, and yet other functionalities provide information that helps to make decisions in more complex tasks. For clarity's sake, there are four main categories.



The **GallitoAPI** analysis engine offers these functions:

- **CompareContentToGold** [Academic text assessment]

  It needs a **labelled space** that defines the relevant concepts to be identified (see change of base manual) and assessed. It also needs a model text or "golden text" that serves as the standard reference for comparison (see "Golden text" in the document on the main **GallitoAPI** files). Once these components are ready, the task consists in evaluating the presence/absence of the labelled concepts by comparison to their presence/absence in the standard text.

- **CompareContentToGoldByPartial** [Academic text assessment]

  It needs a set of partial texts representing the concepts to be identified (see "Partial texts" and "Golden text" in the document on the main **GallitoAPI** files). Once these components are ready, the task consists in evaluating the presence/absence of these concepts by comparison to their presence/absence in the standard text.

- **DoContentAnalysis** [categorization]

  It needs **a labelled space** that defines the relevant concepts to be identified (see change of base manual) and assessed. The task consists in estimating the amount or ratio of each concept in a text.

- **DoContentAnalysisByPartial** [categorization]

  It needs a set of partial texts representing the concepts to be identified (see the section on the main **GallitoAPI** files). The task consists in estimating the amount of each concept in a text.

- **DoContentAnalysisByPartialByLev** [Categorization & Academic text assessment]

  It needs a set of partial texts representing the concepts to be identified (see "Partial texts" in the document on the main **GallitoAPI** files). The task consists in calculating the extent to which the analysed text (short text) matches the partial texts proposed. This function is specially aimed at assessing questions with a brief answer, and complements exclusively semantic assessments.

- **DoRoute** [Categorization]

  It needs a set of partial texts representing the concepts to be categorised (see "Partial texts" in the document on the main **GallitoAPI** files). The task consists in categorising the text under any of those conpcepts.

- **doIntraTextAnalysis** [Academic text assessment]

  Returns information (data) on textual coherence and some surface data:

  - Number of paragraphs
  - Average number of words per sentence
  - Average number of words per paragraph
  - Average number of sentences per paragraph
  - Average global weight
  - Average sentence-sentence coherence
  - Average paragraph-sentence coherence
  - Paragraph-paragraph coherence

- **getNearestNeighboursList** [System auxiliary info & Categorization]

  Returns an ordered list thematic neighbours together with their similarity and significance values.

- **getSumarizationBySentences** [Summarization & Highlighting]

  Returns an eligible number of sentences representing the full text. This function can be configured so that its outcome is promoted on the basis of:

  - Global weight: word specificity

    - Frequency: word significance
    - Topic: pre-established concepts (This parameter requires a labelled space)

- **getSumarizationByTermlength** [Summarization & Highlighting]

  Returns the list of words that best represent a text. The outcome of this function can be promoted on the basis of the word significance index and word frequency. It can also be promoted on the basis of a pre-established concept or topic. This function requires a labelled space.

- **getSumarizationByTermWeight** [Summarization & Highlighting]

  Returns the list of words that best represent a text. The result of this function is promoted on the basis of the specificity index. It can also be promoted on the basis of a pre-established concept or topic. This function requires a labelled space (zoom space).

- **getVectorLengthAverageOfText** [System auxiliary info]

  It returns the average significance of the words in a text.

- **getVectorLengthOfWord** [System auxiliary info]

  Returns the significance of a word or word chain.

- **getVectorOfTerm** [System auxiliary info]

  Returns the vectorial representation of a word or word chain in the thematic space.

- **getVectorOfText** [System auxiliary info]

  Returns the vectorial representation of a text in the thematic space. This representation is very useful to translate a text into numerical coordinates, making its retrieval less costly. It is also useful to have a labelled thematic space in which some dimensions of the vector represent specific concepts.

- **getWeightAverageOfText** [System auxiliary info]

  Returns the average specificity of the words in a text.

- **getexplicitContentDetection** [Categorization]

  Identifies specific structures in sentences in the analysed texts. These structures are defined in grxml grammars.

- **getMaturationOfterm** [Academic text assessment]

Returns an index that estimates when a specific word will mature. The index gives a specific age, indirectly providing an estimate of its acquisition/comprehension difficulty.

- **getMaturationOfText** [Academic text assessment]

  Returns the average maturity index of the words in a text. It can be used as a partial index (together with other text properties) for text comprehension difficulty.