POLITECNICO DI TORINO

Corso di Laurea Magistrale in
Ingegneria del Cinema e dei Mezzi di Comunicazione

Tesi di Laurea Magistrale

Conceptual analysis of learning scenarios through Educational Data Mining: the "eSIT4SIP Knowledge Base" project

Relatrice
Prof.ssa Laura Farinetti

Candidata
Selene Di Viesti

Correlatore
Prof. Wolfgang Müller - Pädagogische Hochschule Weingarten

Aprile 2019

Index

Introduction 6

Research Objectives 7

Structure of thesis 9

1. State of Art 10

1.1 Data Mining and Educational Data Mining 10

1.2 The Semantic Web 16

1.3 Web Scraping 18

1.4 Concept and Ontological Maps 20

2. Concept and Design 29

2.1 eSIT4SIP Project 29

2.2 Programming languages 32

2.3 Editors 33

2.4 XWiki 33

2.5 CMap 33

2.6 D3.js 34

2.7 Additional information 35

3. Implementation 36

3.1 *screeping.rb* 39

3.2 *jsonToTextConverter.rb* 41

3.3 *matchesResearch.rb* 42

3.4 *mapsWriter.rb* 43

3.5 *mapAppending.rb* 47

3.6 *mapForEachDomain.rb* and *mapForEachDevice.rb* 48

3.7 *mapForEachScenario.rb* 50

3.8 Interactive maps building 52

4. Results 56

4.1 Scraping XWiki 56

4.2 Comparison between the ontology and the learning scenarios 57

4.3 Graphical visualisation 59

5. Conclusions 65

References 67

Index of tables

Table 1. Scripts overview 52

Table 2. Results overview 65

Index of figures

Figure 1. A Concept Map of what a Concept Map is 21
Figure 2. The Domain Knowledge Acquisition Process 27
Figure 3. Structure of the eSit4SIP Project 35
Figure 4. Portion of the Learning Ontology 37
Figure 5. Result of the export as a text file 38
Figure 6. Result of the export as Life Map 38
Figure 7. Portion of "scenarios.json" file 41
Figure 8. Example of "scenario1.txt" 42
Figure 9. Output "scenario1_occurrences.txt" 43
Figure 10. Output "Map_Scenario1.txt" 44
Figure 11. Steps followed 44
Figure 12. Portion of the result of "Map_scenario1.txt" imported as LifeMap format 45
Figure 13. Example of hash added to the ontology 47
Figure 14. Part of the file Ontology_domain.json 48
Figure 15. Output "Mathematics.json" for the mathematics domain 49
Figure 16. Map of scenario1 created with d3.js 54
Figure 17. Project overview 55
Figure 18. Matches scenario0 57
Figure 19. Matches scenario1 57
Figure 20. Matches scenario48 57
Figure 21. Matches scenario56 57
Figure 22. CMap of scenario0 59
Figure 23. CMap of scenario1 60
Figure 24. CMap scenario0 without specifying nodes positions 60
Figure 25. Complete map with d3.js library 61
Figure 26. Map for domains with d3.js library 61
Figure 27. Map for devices with d3.js library 62
Figure 28. Map of "Mathematics" domain 62
Figure 29. Map of scenario1 and result of clicking 63

Introduction

In recent years, the ICTs[1] permeate every aspect of our life, including the field of education. Many researches demonstrated that the use of these technologies in the learning process can improve both learning and teaching.

The *eSIT4SIP Knowledge Base* [1] is a project built by the University of Education of Weingarten in cooperation with the Open University of Cyprus and the Linnäus University, as the result of the ERASMUS+ Strategic Partnership Project *Empowering the School IT infrastructures for the implementation of Sustainable Instructional Patterns*. The basic idea on which this project is built is that the ICTs improve learning in different ways: when they are used by students as a source of research, when they are used to provide examples during a lesson, when they are used as a form of support by teachers etc. The aim of the project is to support teachers, who work at the university, but also the ones who work at the high school or at the primary school, in organising the technical realisation of learning scenarios. A learning scenario is a recommended template useful to plan a certain lecture: it is referred to a specific learning domain (e. g. mathematics, economies, arts, etc.), it requires certain devices (e. g. table, projector, printer), it is related to a specific teaching approach (e. g. front learning, collaborative learning, etc.) and takes place in a given area (e.g. in class, in the school laboratories, in field trip, etc.).

The scenarios offered as examples on the web page of the project cover a wide area of applications, however most of them are targeted at the higher education. Most of them are written in English, then in German and a small part in Greek; they are classified by subjects, needed devices, patterns, teaching approaches, spatial settings, information functions and languages. A more detailed description of this project will be provided in the following chapters.

As the title points out, the project described in this paper falls within the scope of the Educational Data Mining. EDM is the Data Mining applied in the context of e-learning, where the term *e-learning* is used to denote the presence of ICT methods and resources that allows to improve the traditional teaching processes and resources.

In wider terms, the use of the ICTs in the field of education and the research linked to Educational Data Mining are intended to improve the learning process. On one side,

[1] Information and Communication Technologies.

their aim is to support students in understanding, studying, learning, accessing and organizing materials and they also allow distance learning (for example the MOOCs courses would not exist without them). On the other side, they have the purpose of supporting teachers in setting up lessons, materials, documentations.

The research goals are various: predicting students' performances and students' learning behaviour in order to prevent potential matters, discovering or improving domain models characterizing the content to be learned and best instructional sequences, investigating the effects resulting from different kinds of pedagogical (learning) support provided by e-learning software, building computational models including the previous three (of the students, the domain and the software's pedagogy) to push forward scientific knowledge about learning and learners. Particularly, as will be explained more in details in the next paragraphs, the aim of this specific project is to improve teachers' experience in planning their lectures, using the eSIT4SIP platform as a source of research and inspiration, and providing concept maps to give a graphical representation of the information. Concretely, in this project, the scenarios are scraped, analysed and compared to the ontology provided by the authors of the eSIT4SIP project. To do this, Ruby code language and CMap software have been used. Then, from the data collected, interactive concept maps have been created. In this case, d3.js library and JavaScript code language have been utilised.

Research Objectives

This project is part of the larger, academic, European project *eSIT4SIP* [1], a web-platform for improving teaching and focusing on teachers. At the present time, the eSIT4SIP web page provides a large number of learning scenarios, catalogued by subject, devices, language, patterns, teaching approach, information functions and spatial setting. The research can be filtered with some checking menus on the left side of the web page and the result of the research thus filtered is a list of the scenarios with their title and a little summary. However, additional information, at the same time synthetic and schematic, could be very useful to teachers and professors who search the best suitable learning scenario for their target topic, according to the requiring infrastructures available in their own school or, more often, university.

Furthermore, once a research is finished, it could be very useful for users having a graphical representation of the resulting learning scenarios to easily compare the candidates; the best way to do it, it is using concept maps. Finally, the current cataloguing has been done by hand by one co-worker to the project, this means that on one side it could be very precise and detailed, but on the other side it could contains mistakes due to human limits, besides the fact that it takes a very long time and it is unthinkable on large scale if the number of learning scenarios increases. It would be very useful doing this job automatically, with random checks by people involved in the project.

For these reasons, the final purpose of this project is providing concept maps for each scenario, showing the most important features of each learning scenarios according to the abovementioned criteria. In addition to these concept maps, this project aims also to provide other concept maps to show which scenarios are related to a certain topic (for this particular case, which scenarios belong to a specific domain, rather than which scenarios require a specific device). These maps should be interactive: the user is redirected to the web page of the scenario on the eSIT4SIP site by clicking on one node (that represents a scenario) of the concept map.

The purpose is to build them automatically, through a computer program written in the programming languages Ruby and JavaScript. The former programming language has been chosen because it is a fairly recent and user-friendly language, very useful for matters that concern web scraping and because it is the program language chosen by the developers of the eSIT4SIP project and it has already been used for some purpose. JavaScript has been chosen because it is required by the d3.js library, which allows to display concept map on web browsers. In order to display the concept maps created with 3d.js library on a web page, a third language, HTML, is needed.

The software chosen to display static concept maps is CMap. It allows users to import maps from text files properly written and it is the same software used by the authors of the *eSIT4SIP* project to display the maps they created from the ontologies about learning, infrastructures and learning domains.

Then, to display the final concept maps it has been chosen the library d3.js because it allows to create automatically concept maps without specifying the position of nodes, and it also allows to have an interaction (for examples using link to webpages).

Before displaying concepts maps, the scenarios have to be firstly downloaded from the place they are stored in (XWiki). Then, files containing learning scenarios and the text file containing the ontology must be edited in order to compare them. Once each scenario has been compared with the ontology, and once common points between one scenario and the ontology have been highlighted, new concept maps of the scenarios can be built. Starting from the text file of the ontology, every concept which has a match in the text file of the considered scenario, has to be kept, whatever has no match has to be deleted.

In this way, it is possible to generate text files, one for each scenario, containing matches with the starting ontology. These files can finally be imported in the CMap software, to be compared directly with the starting ontology but without the possibility of interaction and without the possibility to underline keywords matched with the ontology (for instance, writing them in a special character or colours). Otherwise, maps generated with the JavaScript library should be uploaded on the eSIT4SIP web page.

To sum up, this project can be divided in three phases: web scraping for downloading the scenarios, research of topics covered by each scenario, creation and display of concept maps.

Structure of thesis

This script consists of four chapters. The first chapter, *State of Art*, will contain preliminary information and previous researches belonging to the same field of interest of this project. In the second chapter, *Concept and Design*, the context of this research will be presented and a detailed description of used tools and the implementation steps will be outlined. In the third chapter, *Results*, the results of this work will be presented and discussed. Finally, the last chapter, *Summary & Conclusions*, will summarise the work and will suggest future implementations and improvements.

1. State of Art

In this chapter some preliminary information is provided about Data Mining and Educational Data Mining and about concept maps and ontologies. Furthermore, there will be an overview on some researches already pursued in this field of interest.

1.1 Data Mining and Educational Data Mining

As already noted in the *Introduction*, the aim of this project takes place in that area of the Educational Data Mining focused on improving teachers' condition, for this reason it is worth explaining what Educational Data Mining is.

The word *Data Mining* arose for the first time within the database community around the 1990s. Particularly, Christopher W. Clifton, a professor of Computer Science at Purdue University (Indiana), provided the following definition: «Data mining […] is the process of discovering interesting and useful patterns and relationships in large volumes of data» [2]. According to professor Clifton idea, the Data Mining involves sectors as statistic, artificial intelligence and database management in order to deal with the data sets. The field of application of Data Mining is very wide: it is used in business, in scientific research and even in government security. Since Data Mining tools allow enterprises to predict future trends, this area is very closed to learning analytics and is widely useful in education, hence we talk about Educational Data Mining (EDM).

This last concept extends the field of Data Mining, indeed the Educational Data Mining deal with the devising of methods which allow to study and analyse the abundance of data collected from educational environments and with the use of these methods to better known students and the contexts which they learn in [3].

Before proceeding, a brief digression about learning models. Actually, learning process began in the classroom and relied in behavioural, cognitive and constructivist models [4], [5]. In the behavioural models, observable changes of students in their behaviour are the basis to assess the learning outcome; in the cognitive models, instead, it is essential the active engagement of teachers in the learning process; constructivist models rely on the students' capability to learn on their own from the available knowledge. In the recent years, appeared the new notion of *Connectivism* which is the «amplification of learning, knowledge and understanding through the extension of personal network» [6]. As Siemens explained, learning is no more an internal, isolated activity [5], but it is like an action in a network of nodes that increases the learning experience of students and decreases the need for the direct involvement of a Professor. As a matter of fact, traditional learning environments progressively became community-based learning environments [7].

Anyway, the climax of the rise of the Educational Data Mining as a new and independent emerging discipline took place in 2008 when the *International Conference on Educational Data Mining* and the *Journal of Educational Data Mining* were created. A very important factor is the introduction of public repositories (e. g. the PSLC DataShop and the NCES) which allows a high feasibility of EDM. EDM algorithms are used for the extraction of knowledge from educational data in order to study the attributes which can help to maximize the performance.

Data Mining techniques and algorithms used for knowledge discovery from database can be classified into two categories: verification-oriented algorithms and discovery-oriented algorithms. The former check pre-defined models against the extracted data, the latter automatically discover new rules from the extracted data. Some of the techniques are: Clustering, Prediction, Classification, Statistics (which identifies outlier fields and records using mean and testing), Neural Networks, Web mining (a technique for mining web data), Decision Trees, Relationship Mining, etc.

According to researchers, these methods fall into the five categories briefly described below.

- *Clustering* is a method to group similar data into clusters. In this technique clusters (or groups) are not predefined, this means that the most shared features within the data set are unknown. Data points which naturally group together are found among the full data set in order to split it into smaller groups. The grain-sizes of the clusters can be different in accordance to the phenomenon that is intended to be investigated. Clustering algorithms might start with prior hypotheses resulting from previous researches with different data, but they might also start without prior hypotheses. Furthermore, the algorithm can assume that each data point must be part of just one cluster or that some points can be part of more than one cluster or part of no cluster.
- *Prediction* is a technique to predict a future state or a current state. In EDM the prediction technique is mostly used in two ways. The first way is using this

technique to detect which are the properties of a model that are important for prediction, by providing information about the underlying construct. Thank to this methodology it is possible to predict students' outcomes without the prediction of other intermediate elements. The second way which prediction technique can be used in, is the prediction of the output value in contexts where is better not to directly get a label for that construct. One kind of Prediction is the *Classification* technique. *Classification* is a two-way technique which consists of training and testing. Contrary to the *Clustering* method, it maps data into specified classes and the predicted variable is binary or categorical. In the *Classification* methods there are decision trees (graphics which contain decisions linked to their consequences, indeed, the only statements are conditional control), logistic regression and support vector machines. There are other two types of *Prediction*: *Regression* and *Density Estimation*. In the former, the predicted variable is continuous, in the latter the predicted variable is represented by a probability density function (e. g. Gaussian function). In *Regression* methods there are linear regression, neural networks (a method to improve the interpretability of the learned network which is based on the use of extracted rules for learning networks), and support vector machine regression. In any case, for each type of prediction the type of the input variables can be categorical or continuous.

- *Relationship Mining* is a method which identifies specific relationships among data. There are four kinds of Relationship Mining. The first (association rule mining) has the aim to discover "if-then" rules so that when a set of variable values is identified, from this set it is possible to define a specific value that another variable will typically have. The second type (correlation mining) tries to identify linear correlations between two variables. The third type (sequential pattern mining) aims to detect temporal associations among events. Lastly, the objective of the casual data mining is to discover whether one event was the cause of another. To do it, the covariance of the two events might be analysed, or information about what triggered one of the events might be used. The relationships found with the *Relationship Mining* method must be statistically significant and interesting.
- *Discovery with models* is a technique to create a model of a phenomenon through prediction, clustering and sometimes knowledge engineering (that means

preferring human reasoning instead of automated techniques). With the model thus created it is possible to carry out other analysis, for example the model can be used in the *Prediction* technique as a predictor variable in the prediction of new variables.

- *Distillation of data for human judgement.* Sometimes people could deduce something about data that exceeds the direct scope of totally automated data mining methods. We are talking about information visualization methods. Data must be extrapolated for human judgment for two reasons: to allow, or to improve, the identification and the classification. For instance, the data distillation for identification shows data in ways which let humans easily recognise well-known pattern difficult to formally declared. On the other side, data distillation for classification helps humans being in the development of future predictive models: portions of a data set are shown in visual or text format, then some human coders label them.

Critical issues that the Educational Data mining has to face in nowadays are:

- Educational data is incremental in nature, that is why maintaining the data warehouse is very difficult;
- Lack of Data Interoperability: because of the wide range of storage locations and data platform heterogeneity, a scalable data management has become critical to perform;
- Possibility of Uncertainty: because of the presence of uncertain errors, no model can predict hundred percent accurate results [8].

Some related works are listed in the following paragraphs.

Despite Data Mining is a useful tool into the e-learning domain, as well as into e-commerce and into business application (the predominant fields which is used in), there are some crucial differences in applying it to the former or to the latter. They are: the purpose (guiding students in their learning or guiding clients to purchase good) and the objective (improving the learning experience, which is subjective, or increasing profits, which are quantified) [9].

The current EDM research is mostly focused on mining data produced by the e-learning system [10]. Nonetheless, Educational Data Mining is expected to address problems linked to different phases of the learning process; these phases can be classified into four categories: formal or informal, intentional or unexpected [11]. The formal learning takes place in training and in educational institutes; it leads to the acquisition of official certificates or qualifications. The informal learning, instead, takes place in the daily life, out of training and educational institutes and does not lead to the acquisition of official certificates. An example of this may be learning through the ICTs or through the participation in the on-line life [4].

Even though it is fairly a new field of investigation, scientists have already applied data-mining methods in e-learning environments in different interesting ways. For example, as Arruabarrena et al. present [12], the most common practice is the application of knowledge extraction techniques for the evaluation of e-learning systems. This kind of work is achieved in order to constantly improve the learning experience.

Moreover, there have been also many investigations whose purpose was the analysis of students' performances to identify common issues and predict trends. In this field of application, it is noteworthy the study of Bhardwaj [13]: he predicted students' performances using the Naïve Bayes algorithm (already mentioned before). The results helped to build a predictive model which identifies students who are at risk of failure, so that it can be possible to act in time.

Besides, Varghese, Tommy and Jacob [14] conducted a research on students' performances too. They decide to use the K-means algorithm[2] to cluster about 8000 students' data. In this case, data were recorded in five consecutive years and they included attendances, marks and seminar assessments. Their results showed a strong relation between students' performance and their attendance.

The academics Gulati and Sharma [15] assert that the education system can be improved in the following branches: orientation, student performance; and organization management. Such improvements would be possible thanks to the knowledge derived from Data Mining analysis. Still in the field of students' performance prediction, Baradwaj and Pal [16] used the ID3 decision tree algorithm[3] on a group of 50 people enrolled in a specific course program. Their aim was to generate a decision tree to predict students' performance at the end of one semester; the prediction was based on data obtained from students of the same course program in earlier years. They chose to collect

[2] K-means clustering is a technique of vector quantization used to partition *n* observation into *k* clusters. Each observation is part of the cluster with the nearest meaninig.

[3] ID3 (Iterative Dichotomiser 3) is an algorithm that generates a decision tree from a dataset.

data concerning attendance, tests and marks. Even Abeer and Elraby [17] conducted a research focused on the creation of classification rules and on the prediction of students' performance; the prediction was based on previously recorded behaviour and activities of other students in the same course program.

Pandel and Pal [18], instead, used the Naïve Bayes classification to analyse, to classify and to predict students' performances. Data came from different colleges and have been collected from 600 people.

Bhardwaj and Pal [19], conducted in India another significant research using the Naïve Bayes algorithm. Their target was a group of BCA students (Bachelor of Computer Applications). They discovered that the grade in senior secondary school is the most influencing factor for students' performance in bachelor's study. In addition to this, they discovered that also factors like the living location, the medium of teaching, the students' other habits, the family annual income, the mother's qualification and the student family status affect students' performances.

To proceed in this overview, Ivancevic, Celikov & Lukovic [20] studied the possible relationship between students' seat selection in a classroom and their assessment. They applied a K-means algorithm and discovered that students with low levels of spatial deployment have lower assessment score (about 10% lower) than the others with high spatial choice.

A further interesting research about predictive models for early detection of student at-risk has been conducted by Wolff et al. [21]. In this case, the focus was on distance learning modules, where there is no direct contact between students and tutors. They collected current students' demographic data and previous students' data. Furthermore, they use a combination of three methods: Nearest Neighbours, Classification and Regression Tree and, lastly, Bayes network. The result of the research showed that most students who failed, actually faced difficulties very early, so this is the best moment to intervene.

1.2 The Semantic Web

The Semantic Web can be considered an extension of the World Wild Web. It is a mesh of data associated in a way so that they are easily processed by machines instead of human operators. In the current Web infrastructure, there is a distributed network of web pages and each of them can refer to another one through the URLs[4] global links. To ensure consistency within a page, sophisticated web site replace the structure mentioned above with databases or XML backend. The aim of the Semantic Web is to support a distributed Web in terms of data, rather than in terms of presentation: not one page pointing to another, but one data item pointing to another through the URIs[5] global references. The infrastructure of the Semantic Web enables the data to drive the presentation (web page), thus various web pages can present views into a consistent body of information [22].

As well as the WWW basic idea was to allow anyone to write a page and to say something, in the Semantic Web our data infrastructure shall allow anyone to express a piece of data about some entity so that can be combined with information of other sources. Hence, tools like RDF[6], RDFS, SPARQL, SKOS and OWL are fundamental to make the information resources from this medium sensible, usable and durable. There are already many public data sources available in RDF, for example *dbpedia,* an RDF form of Wikipedia.

Abstractions, also called models, are so widely used because they can assist people assembling their knowledge. Specifically, models help people in the communication, in the explanation and in making predictions, in mediation among different viewpoints [22]. In summary, models are used to organise human thoughts in form of explanation. The form of explanation is useful because it makes easier to reuse a model, totally or in part; it is the key to understand whether a model is suitable or not. Related to explanation, there is the idea of prediction: if a model is an adequate explanation of a situation, it can be used to make predictions of similar situations.

The Semantic Web borrowed from the Object-Oriented Programming[7] the notion of hierarchy of classes and subclasses, and it uses this notion to represent commonality

[4] URL is the acronym of Uniform Resource Locator.
[5] URI is the acronym of Uniform Resource Identifier.
[6] RDF stands for Resource Description Framework.
[7] An object-oriented programming language model (OOP) is based on the concept of *objects*: an object can contain data in the form of fields (also called *attributes*) and code in the form of procedures (called *methods*). The most popular ones are *class-based*: objects are instances of classes and a class can contains different sub-classes, thus the structure is hierarchical.

and variability: this means that, while high-level classes constitute commonality among a large range of entities, on the other side, lower-level classes constitute commonality among a small and specific set of objects.

Modelling languages provided by the Semantic Web (e. g. RDF, RDFS, SPARQL etc.) have different levels of expressivity in order to better describe every kind of model, because each one is different from the others. The Semantic Web standards organization makes each language-level sustained by the one before, in this way languages themselves appear to be layered. RDF serves as the foundation for the others representation languages and it manages distributed data. A *triple* is the basic building block for RDF language; it consists of one subject, one predicate and one object[8]. There are several ways to express RDF in textual form, they are:

- N-Triple: it refers to resources through their fully URIs;
- Turtle: it is similar to N-Triple but more compact. It consists of a sequence of directives, triple-generating statements or blank lines. Prefixes can be defined at the beginning of the file and multiple triples with the same subject are grouped;
- RDF/XML: it is an XML serialization of RDF. Like Turtle, prefixes can be defined at the beginning of the files, thus avoiding unnecessary repetition of URIs;
- Blank Nodes: they are used to represent resources with no identity on the web. They are nodes in the RDF graph.

In the following paragraphs, the modelling languages of the Semantic Web are shortly described.

The standard way to access RDF files is the SPARQL[9] language. As well as for SQL[10] language, the main structure of a SPARQL query consists of the syntax SELECT – WHERE. Then, variables (e. g. DINSTINCT), some filters, any optional matches, or even negations, can be added to better articulate the request.

RDFS is the schema language for RDF; thanks to RDFS, it is possible to define the constructs for types of objects (called *Classes*), the constructs for properties describing objects (*Properties*), the relationships between properties (*subProperty*) and the

[8] The sequence subject-predicate-object (SOV) comes from Linguistics [61].
[9] SPARQL stands for Protocol And RFD Query Language.
[10] SQL (Structured Query Language) is the standard query language for databases.

relationships between types of objects (*subClasses*). RFDS is written in RDF itself; therefore all the information is expressed in triples.

OWL[11] is a set of constructs which provides a considerable flexibility for modelling in the Semantic Web. RDFS-Plus is a subset of OWL, it includes some features like *equalitClass*, *inverseOf*, *TransitiveProperty*, *DatatypeProperty* etc. A key functionality of OWL is represented by the possibility to define restriction classes.

Finally, SKOS[12] is a W3C[13] Recommendation. It provides the resources for representing knowledge organization system, in such a way that they are distributed and linkable. The vocabulary of SKOS is fundamental for linking information on the web.

In the following chapters, some examples of ontologies will be presented.

1.3 Web Scraping

The field of Web Scraping is related to the Data Mining and the Semantic Web. Furthermore, it will be used in the first part of this project, when the scenarios are downloaded from the web environment they are stored in. For these reasons it is worth saying something on this subject.

Scraping is the conversion from human readable to machine readable data. Since the data is frequently incorporated into layers for formatting meta data, formatting and data must be split and sometimes it can be useful using the formatting for data interpretation. The four phases of the scraping are: *Document Load*, that means loading the fully web page (or XML page or PDF document etc.), usually as a string of characters; *Parsing*, which consist of interpretation and conversion of HTML (or XML or PDF etc.) meta data into a language readable by the script (usually HTML pages are parsed as Document Object Model); *Extraction*, that is the extraction of desired data from the result of the Parsing phase; *Transformation*, which is the conversion of data unto useful formats.

Web Scraping methods evolved with the development of the WWW. For instance, the methods that are extensively used nowadays, were not available at the beginning. They are the DOM parsing technique (born after the 2000, when the Document Object

[11] OWL stands for Wen Ontology Language.
[12] SKOS is the acronym for Simple Knowledge Organization System.
[13] W3C is the abbreviation for World Wide Web Consortium, the main international standards organization for www.

Model established itself in the DHTML) and the APIs technique (born after the 2005). Anyway, Web Scraping methods can be classified in:

1. *Manual Scraping*. It can be performed if the among of data is minimal or when a repetitive task is not usable, or even when the setting of an automated scraping would take longer than manually scraping the data set. Finally, it is necessary if security measures or particular features of the web site does not enable automated methods.
2. *HTML Scraping*. It is the scraping accessing elements through their HTML tag (e. g. head, body, h1, etc.). It is the method used in this project.
3. *DOM Parsing*. It is the evolution of the HTML Parsing based on DOM, used for CSS and JavaScript. It provides an easier navigation through the web page.
4. *XPath*. It stands for XML Path Language, despite the name it is usable both for XML documents and for HTML format. It necessitates a more detailed structured webpage than DOM and provide the same navigation through the web page content.
5. *APIs*. Contrary to the methods above, Application Programming Interface does not work to reach human-readable outputs, but it requires an application as a communication partner: for this reason, it can be defined a machine-readable interface. The APIs world is very varied and fragmented; indeed, each API has its own specification and options. By sending a standard HTTP Request to an API Endpoint, from server returns an answer whose format can be specified as an option in the request. Anyway, the most common format is JSON.

Tools available for web scraping can be divided into the two following categories:

- *Visual Scraping tools*. They do not require a programming knowledge and they are mainly web-based. The access to data is quick but since they are not customised, it might be necessary scrubbing data after scraping. *ScraperWiki, Import.io* and *Scrapinghub* belong to this category.
- *Scraping with a scripting language*. In contrast to the former, this kind of scraping requires a strong knowledge of the programming language used and a more detailed setting, but the result has a better quality. Libraries to perform scraping exist in most languages (R, Python, PHP, JavaScript, Ruby etc.). This is the type of scraping used in this project.

A second classification of the available scraping tools is based on the type of software:

- *Cloud Software*. In this kind of web scraping, the application backend is in the cloud server and a User Interface is given through a web browser. It is possible to scrape large amounts of data without the need of own computer on. Attention must be given to the cloud location: it might happen that some web pages are not accessible from a country in their complete spectrum. Some examples are *Octoparse*, *Scrapinghub* etc.
- *Desktop Software*. Since the data are downloaded from the Website, parsed and saved locally, this typology requires a RAM of large size. Some examples are *ParseHub*, *FMiner* ect.
- *Programming libraries*. These libraries allow users who need a custom Web Scraping solution not to develop the full program. They can use frameworks to re-use generic modules, thus saving time to develop their own solution. *Scrapy* and *Goutte* are two of the available libraries.
- *Browser Extensions*. These have been the first type of web scraping. They work well only if the amount of data comes from few websites and only for a quick research. They exist as add-ons for the browser (e. g. *Outwit Hub*, *Web Scraper*).

1.4 Concept and Ontological Maps

A concept map can be described as a graphical tool for the representation of some knowledge. Its structure consists of nodes and labelled lines. The former, nodes, correspond to important terms that identify the concepts of a knowledge-domain; the latter, lines, correspond to relationships between two concepts (nodes); furthermore, the label associated to each line denotes how two concepts are related. Finally, a preposition is the association between two nodes through a labelled line.

Concepts in concept maps are represented in a hierarchical order: the most inclusive and general concepts are at the top of the map, whereas the more specific and less general concepts are below. By the way, the hierarchical structure for a specific domain of knowledge relies also on the context in which this knowledge is applied or considered.

Another relevant characteristic of concept maps is the incorporation of *cross-links*: they are relationships (and so, propositions) between concepts that belong to different domains of the given concept map.

The following concept map [23] shows what a concept map is:

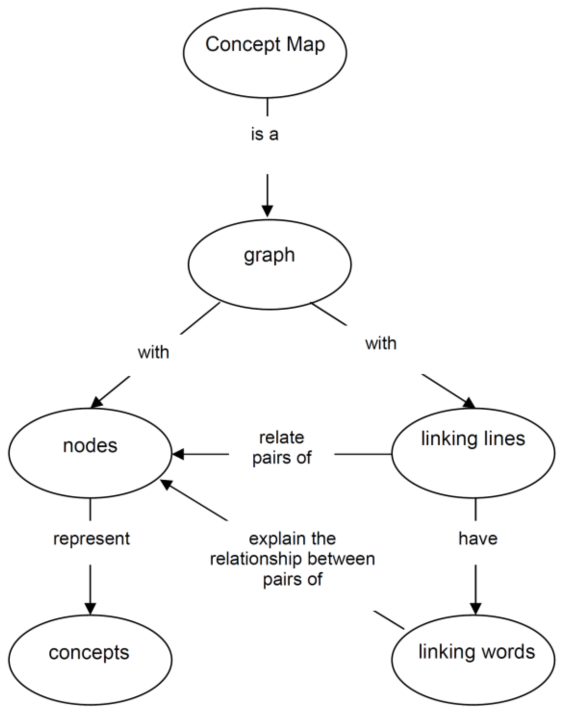


Figure 1. A Concept Map of what a Concept Map is

Maps are important because they allow to intermediate the outer and physical world with the inner and mental world. Concept maps are not the only kind of map used to describe intellectual landscapes. *Knowledge cartography* is the study of the design and the engineering of the different kinds of maps; particularly, its aim is studying how knowledge maps can be created. Moreover, *Knowledge cartography* is the study of *cartographic practices*, which means the study of the approaches used both by beginners and experts in generating and utilizing maps. In this case, it investigates how effective knowledge maps are in relation to different kinds of users.

It is shown below the classification of the different kinds of knowledge maps [24]:

- *Concept Mapping* (already explained above);
- *Mind Mapping*: the user maps keywords, sentences and pictures from a central idea. It is useful for visual notetaking and brainstorming because of the low constrains on how labelling elements or making links;

- *Thinking Maps*: they consist of a set of abstract visual conventions planned to support cognitive skills;
- *Issue Mapping*: it is designed to face socio-technical problems. Issues, Positions and Arguments are linked in consistent ways and can be visualised as textual outlines or graphical maps;
- *Argument and Evidence Mapping*: they are useful to expose the structure of an argument to clarify the status of a debate;
- *Web Mapping*: it as a way for users to capture, iconify, link, position and annotate hyperlinks in a visual space while they are navigating. Thus, users create a personal and rich trail.

The field of application of concept maps is very wide and it includes many different areas: from education to knowledge management, from business to intelligence. In this paper, we will focus our attention on the field of education and on the benefits which concepts maps can provide both in learning and teaching. In 1990, in an article on the Journal of Research in Science Teaching, the American educator JD Novak, was among the first to outline the potentiality of concept maps to improve learning and teaching in science classrooms. He defined concept maps as «tools for organising and representing knowledge» [25].

Actually, Novak has studied concept maps since the 1960s. He based his work on the assimilation theory formulated by the psychologist Ausubel, who underlined that a new concept or a new proposition is connected with the existing relevant knowledge of learners. These concepts can be represented visually by concept maps. Furthermore, thanks to the advantage of visualisation, concept maps are fit to represent domain ontology. In fact, concept maps and ontologies are quite similar: the triple subject-predicate-object can be the formalization of an ontology, while the triple concept-relation-concept is the concept map formalization. The most meaningful difference between these two representations is that ontologies are more formal and more expressive by reason of their attributes, values and restrictions.

In an ontological map, nodes (equivalent of concepts) are linked to other nodes with specific relationships (e. g. is a/an, has a/an, belongs to, represent etc.). Concept mapping may be the first step in ontology building and it is also suitable for representing knowledge structure for meaningful learning.

Previously, most concept maps were built manually by experts; although several computer-based tools are nowadays available, the actual data input still depends solely on human experts' contribution.

The automatic or semi-automatic generation of concept maps from texts or other kind of documents (e. g. images, videos, audio recordings, etc) is called Concept Map Mining (CMM) [26]. In the semi-automatic generation of concept maps, the system suggests elements of a map and then a person manually fill in the missing information; in the automatic generation, the user's assistance is not required.

The Semantic Web[14] view counts upon domain ontologies to outline web content and to make it understandable by software agents. In this sense, the semantic web can be compared to an enormous expert system knowledge base. In the Intelligence Tutoring System (ITS) the importance of domain ontologies has become bigger and bigger, in fact they are useful for modelling learner and expert modules [27]. The creation of ontology-based metadata understandable both by machines and humans, like RFD or OWL, allows to express semantic annotation in a standard way. Protégé [28] and other new generation of ontology engineering environments have generally encouraged the creation of ontologies.

Zubrinic, Kalpic and Milicevic [29] provided an overview of CMM for the automatic creation of concept maps from documents written with a lexically rich language. The core of their idea is that in personal learning a concept map can be utilised as a tool to represent a learning plan. Most learners in fact have difficulties to begin building a map from a blank sheet, but if experts provide a skeleton then it would be easier to start the procedure. From the CMM perspective, a document can be formalised as a set of concepts and a set of relationships extracted from the document. The process for CMM generation, as it will be explained more in detail, consists of: concept extraction, relationships extraction and summarization. For the creation of maps from unstructured texts, there are three possible techniques:

- Techniques to create a concept map from a single document [30], [31], [32];
- Techniques to create a concept map from multiple documents [33], [34], [35];
- Techniques to create a concept map from one long document [36].

[14] W3C claimed «The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries» [58].

Usually, in a process of CMM the authors use statistical or machine learning methods[15], which are computationally efficient but not always precise. For this reason, the authors use to combine them with linguistic tools or database of keywords in the same domain of the map.

More specifically, statistical methods are useful to analyse the frequency of terms in a document, but they are imprecise since the meaning of words are not considered. On the other side, machine learning methods include classification, clustering and association rules. For example, TEXCOMON software tool [37] uses classification method for extracting keywords from a document. It uses a KEA[16] algorithm based upon the Naïve Bayes classification [17].

Most of methods are in effect hybrid. They can use dictionaries, ontologies and lists of pre-established terms as a seed to define concepts and relationships more precisely. However, dictionaries are also useful to minimise the number of terms needed during the phase of extraction of concepts, thus words are grouped by similarity into clusters.

Other tools and techniques to improve CMM methods are Lemmatization and Stemming. Lemmatization is the procedure which allows to determine the morphological root of a word (called *lemma*) thanks to a vocabulary and a morphological analysis. On the other hand, Stemming is a computational procedure which removes the beginnings or the endings of terms. Since it doesn't require a morphological analysis of the all text and since the created root is often different from the morphological root of the term, Stemming is used for text in less inflected languages (e. g. English).

When documents consist not only of texts, but they also include images or videos, other tools are needed to transform data into unstructured text and then to create the concept map.

A relevant problem is the need of an objective evaluation method for comparing CMM results. Humans are good evaluators because they are able to capture important terms, but the problem is that each person look at the source content in his own way, so the process for choosing keywords is a matter of personal judgment. To avoid this problem, an inter-human agreement among evaluators may be introduced.

Some related works are listed in the following paragraphs.

[15] A classification of these methods has been provided in Data Mining chapter.
[16] Implemented in Java and platform-independent, KEA is an algorithm which extract key-phrases from text document.
[17] Naïve Bayes classification is a simple probability classification technique. It is based on the assumption that all given attributes in a dataset are independent from each other. Thus, the name Naïve [60].

Reference has already been made to the importance and usefulness of concept maps. A lot of studies and researches have been done into this area, especially in the automatic generation of concept maps, that is called Concept Map Mining (CMM) [26]. Researches have pointed out how supports for appropriate navigation can be helpful to partially solve the problem of information overload which we live in. Among other things, maps can support people in general (so learners too) to refer and simplify information research belonging to a certain domain [38].

Chen et al. [33] carried out a research with the aim to generate concept map for the e-learning field using academic and conference articles as sources from which gather data. To do it, they decided to use keywords appearing in journal articles as *nodes* which represent concepts, and relation-strengths between two keywords in the articles (measured by the distance between these two words in the text) as *links* which represent the relationships between the concepts. Generally, they followed these steps:

1. Concept Item Extraction: in this phase, keywords were classified into appropriate groups. Firstly, synonymous were identified and then keywords were obtained from articles. Then, since authors had to declare an acronym when it appeared for the first time to avoid acronyms with dual meanings, an acronym mapping table was defined to store them. Finally, suffixes (e. g. -ed, -ing, -ions…) were removed using Porter's algorithm for suffix stripping [39]. In this way, similar terms were not used as different keywords;
2. Research Keyword Indexing: this phase was performed using the Principal Component Analysis method (PCA), a comprehensive index for indexing and selecting emblematic research keywords. The variables used to evaluate the importance of topics were: appeared times (number of appearances of a keyword in the article), related counts (number of other keywords in the same sentence) and sustained periods (sentences in which a keyword was recurring from the first to the last time);
3. Calculation of Relation Strength: the higher the strength of relation between two words was, the stronger their relationship resulted.

Maps thus generated were evaluated by a group of experts. The results were positive, in fact experts verified that those concept maps extremely accorded with their idea on e-learning domain knowledge.

Berlanga et al. [40], instead, highlighted the advantages that a tool for automatically generation of concept maps could provide. Specifically, they focused on concept maps from texts written by students (e. g. blogs and essay). That is because they believed that the comparison of one student's concept map over time or the comparison of this map with other students' concept maps or even with concept maps generated from tutor's material, can ascertain learners' progresses and identify remedial actions. In their paper they also provide an overview of some existing tools and then describe their experience with CONCSPECT, the first prototype of an automatic tool for concept maps.

Another way of using concept maps to improve learning is showed by Huang, Tsai, Hsu and Pan [41]. In their study they used text mining technology to visualize students' points of view and to compute the Average Path Length (APL)[18] of each students' map, applying the APL of small world concept. In this study students of different classes (e.g. Science, Engineering, Liberal Art) were asked to watch the same film and then to write a short essay about it. Through text mining technologies was created a concept map for each essay and then all maps were compared. From this experiment came out that students from different education backgrounds perceive facts differently. It must be specified that differences can also be caused by the individual capability in putting sensations down to words.

Moreover, Zouaq and Nkambou [37] presented the *Knowledge Puzzle*'s approach, referring to the e-learning and to the importance of ontologies in the Semantic Web vision. The *Knowledge Puzzle* is a semi-automatic method for generating ontologies from concept maps, that were in their turn created with a semi-automatic process. A Domain Ontology is defined as a specific vocabulary which describes a certain reality. It relies on the identification of key-concepts and relationships in a domain of interest. Human evaluation is crucial at each step of the process, which is not linear but recursive, since it needs many revisions before developing the final solution.

[18]APL is defined as the mean number of steps along the shortest paths computed on all possible pairs of network nodes.

The following figure shows the process:

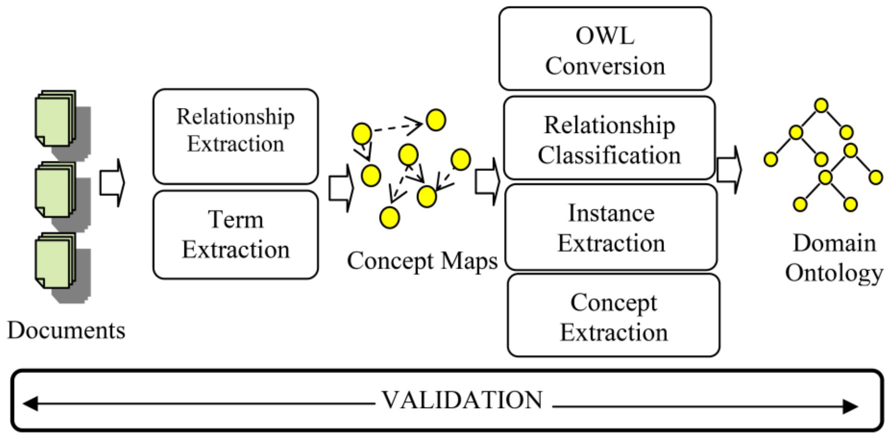


Figure 2. The Domain Knowledge Acquisition Process

Domain Knowledge is usually split into declarative knowledge, procedural knowledge and strategic knowledge[19], but for the moment the *Knowledge Puzzle* focuses on mining the first of them (declarative knowledge). The approach is based on natural language processing (NLP) and machine learning. The first step is determining the content of each document through annotators developed with the IBM's Unstructured Information Management Architecture (UIMA) [42]. The second step is extracting representative n-grams[20] from documents through the machine learning algorithm Kea. Thus, a natural language processing parser (Stanford Parser [43]) finds dependency representations, that can be defined as grammatical relationships linking two different words. In the *Knowledge Puzzle* a word can be considered as a concept if it respects simultaneously two conditions:

1. It is the main subject of several sentences, and so it is a frequent subject;
2. It is linked to other domain words with semantic relationships.

Then, a linguistic editor is used to define rules about semantic-relationship categories, so that it is possible populating progressively a linguistic knowledge base. Until the time the authors wrote the article [37], these were the generic categories:

- Taxonomical relationships: e. g. is-a (-kind/form-of);

[19] Declarative knowledge is what Gilbert Ryle called *know-that* (knowing that something is so), procedural knowledge is the *know-how* (knowing how to do something) [59], strategic knowledge is knowing which, when and why applying a specific knowledge in a particular context.
[20]A n-gram is a sub-sequence of *n* elements from a given sequence.

- Composition relationships: e. g. contain, compose;
- Description relationships: e. g. describes;
- Definition relationships: e. g. defines;
- Explanation relationships: e. g. explains;
- Causal relationships: e. g. causes, is-caused-by;
- Reference relationships: e. g. refers-to, is-associated-with.

At the end of their research, the authors concluded that the *Knowledge Puzzle* allows to define formal relationships which represent the context of the concept. This is a very important feature in the constructivism environment, because in that kind of learning the prior knowledge is used as a framework for the new knowledge to understand and learn.

Another framework for fuzzy generating ontologies is *FOGA* [44]. It is composed by fuzzy formal concept analysis, by fuzzy conceptual clustering and ontology generation, and, finally, by semantic-representation conversion. *FOGA* method expands the formal concept of analysis approach, which is a systematic method that allows to derivate implicit relationships among instances described by a set of attributes. Considering that it provides intentional description for abstract concepts, it can be considered as a conceptual clustering technique[21].

Next, a few examples of some ontologies. *Good Relations* (GR) is an ontology developed by a member of the E-Business and Web Science Research Group at the Universität der Bundeswehr (München). It is used to annotate a considerable number of web pages; thanks to GR, web content providers are able to mark up their web pages to describe offerings. Besides giving page a higher rank, Google uses GR data to improve the information it gives about a product when it is searched.

Quantities, Units, and Dimensions (QUDT) is the ontology developed by NASA to deal with units of measure. Through QUDT, NASA captures and manages information of the *International System* (SI). It provides a global reference for units, a conversation service between systems and a check of the dimensions of the components in a formula, thus checking for errors.

Finally*, Chemical Entities of Biological Interest* (CHEBI) has been developed by the European Bioinformatics Institute. Biological ontologies have been created to provide

[21] The notion of "context", which means the relevant features or attributes shared by a set of objects of the same class, is central for this technique.

unambiguous references to biological concepts because it is necessary to organize information when they are generated in different laboratories. Specifically, CHEBI contains information about 20,000 chemical compounds. It is published as part of the OBO Foundry[22].

2. Concept and Design

In this chapter are presented the context and the existing project which this work is related to, the way followed to implement the project and the tools utilised. In this context the word "tools" means the programming language chosen to write the code, the editor, the software used to display the concept maps and any libraries.

2.1 eSIT4SIP Project

Mention has already been made of the *eSIT4SIP Knowledge Base* [1], a web page that contains learning scenarios. It was born thanks to the cooperation of three European university: the University of Education of Weingarten (Germany), or more specifically the Media Education and Visualization Group (it is the coordinator of the project); the Linnäus University (Sweden), and more specifically the Department of Computer Science and Media Technology; and the Open University of Cyprus (Greece), particularly the Information Systems Department.

The University of Education of Weingarten, as its name suggests, is a school of education focused on pedagogy; the Media Education and Visualization Group is a body of the university which applies computer, media and technology sciences to support the pedagogical aim. The Linnäus University is mainly a school of education too and, as well as for the German university, its Department of Computer Science and Media Technology has the purpose to improve and support the pedagogical studies. Finally, the Greek university offers a large variety of courses, available as distance learning; among them there are computer science and education sciences.

[22] OBO Foundry is the Open Biological and Biomedical Ontologies Foundry. It is a sort of wiki space gathering science-based ontologies.

As can be seen, the three university are involved both in educational and computer technology. They want to encourage practices which could favour schools in the achievement of a "e-mature level", that means a good knowledge of the information and communication technologies, which improve, among the others, the learning process. More in detail, the project aims to guarantee that the stakeholders involved in learning (teachers, students and others) are capable of better enhancing the advantages of investing in ICT infrastructure, encouraging the introduction of innovative models and ideas for the education which connect digital items and learning, collecting best ideas about new kinds of usage of ICT infrastructures at schools in order to promote the exchange of such practices.

The methodology followed by the eSIT4SIP project to reach the targets explained above is:

- Creating ontologies to represent concepts about learning, infrastructures, devices and instructional affordances;
- Gathering of best practices concerning the inclusion of ICT in classroom, in the form of learning scenarios, selected from eSIT4SIP partner schools' practices and from open educational resources (*MnSTEP* and *LehrerOnline*).

The eSIT4SIP web page offers teachers the following features: looking for scenarios; checking whether they have the technical equipment needed to implement a scenario or whether there are options for meeting the technical requirements; eventually, using default school infrastructure specifications as filters while they are searching for scenarios in the Knowledge Base. The issues concern mathematical sciences, computer science, chemical sciences, physical sciences, biological sciences, earth environmental sciences, electrical and electronic information engineering, mechanical engineering, materials engineering, health sciences, economies and business, sociology, political sciences, media and communications, history and archaeology, languages and literature, arts and interdisciplinary topics.

In order to realise the learning scenarios, it is necessary to know whether they are feasible or not. It depends on the knowledge of the ICTs facilities of the schools, and so the infrastructure required to achieve the functions delineated in each scenario. Teachers could use the Knowledge Base to look for scenarios that match a targeted learning context and some objectives, just like their infrastructure and technology at hand, to finally detect the candidates for the implementation. A useful tool to have a complete overview on each scenario is the concept map, or, more specifically, the ontological map; that is because the eSIT4SIP approach is founded on a learning and infrastructure ontology. «The ontology attends as a standardized vocabulary to tag learning scenarios». «Furthermore, — the authors explained on the web site — it serves as the semantic backbone to express relations between scenarios and to link them to learning and infrastructure concepts» [1].

Pay attention to the word *ontology*: even though it comes from philosophy, in computer science it has a completely different meaning: in this area it can be defined as «a formal explicit specification of a shared conceptualization» [45].

Now it is necessary to explain how the *eSIT4SIP Knowledge Base* web site is structured.

Firstly, the web page is a static page. On the top there is a "head" HTML element with the title of the project and four buttons: each of them corresponds to one section of the page. Then, on the left side, there are a search bar and a popup menu which are used for navigating and looking for scenarios. Eventually, scrolling down in the web page, there are an explanation of the project, contacts and publications.

Since the web page is a static page, the content of each scenario needs to be scraped from a *.json* file ("articles.json") which contains every scenario. Actually, the original scenarios are scraped from *LehrerOnline*, a web portal for teachers to download educational material, or *MnSTEP*[23], the homonymous web page of the project. The MnSTEP provided science workshops: «The goal was to provide inclusive, convenient, reliable, high-quality science professional development for teachers» [46].

The scenarios are thus uploaded to *XWiki*, a free wiki software platform written in Java, and some authoring is made there. Afterwards they are scraped again and the resulting file is "articles.json".

The ontology created for the topics contained on the web page are RDF and OWL files: the former is used to define the structure of the data, the latter describes semantic relationships. To display the content is used JENA API, and then SPARQL allows queries.

[23] MnSTEP is the acronym of Minnesota Science Teachers Education Project.

2.2 Programming languages

In this project, the first of two programming languages used to build the program is Ruby [47]. Ruby is fairly recent because it has been developed by the Japanese Yukihiro Matsumoto in the middle of 1990s. It is object-oriented[24] but since it supports multiple programming paradigms, it is also functional and imperative[25]. In addition to this, it has a dynamic type system, which means that types (of the variables) are checked during execution and not before run time; it has an automatic memory management too, indeed it automatically manages the allocation/deallocation of memory. This is a considerable advantage to programmers, who don't need to write code to perform memory management. It is completely user friendly: for example, in contrast to Java programming language, variables in Ruby don't need to be declared. According to its creator [47], Ruby can be defined dynamic, open source, focused on simplicity and productivity. Its syntax is easy and intuitive.

On the official web site [47], a complete documentation about installation, standard library and APIs are provided. Most libraries are released as *gem* and *RubyGems* is the tool that allows to install the packaged library or application (*gem*). It also facilitates the creation, the installation and sharing of libraries [47]. The other libraries are distributed as archived directories of source code.

The second programming language is JavaScript. It became necessary when the first concept maps created for the scenarios and imported into CMap could not be interactive. This is the reason why it was decided to use the d3.js library, which is written in JavaScript. The use of this library allowed to display the maps as HTML files whose source is a JavaScript code. With JavaScript and HTML, it was possible to add links to the nodes.

[24] As already written in note 5, an object-oriented programming language model (OOP) is based on the concept of *objects.* An important feature is that an object's procedures can access and also modify attributes of the object they are associated with.

[25] Functional programming deals computation as the evaluation of mathematical functions avoiding changing-state and mutable data. Programming is done with expressions or declarations, not with statements. The output value of a function is determined only by the arguments which are passed to the function, this means that calling several times the same function with the same input values produces the same result. On the other side, imperative programming uses statements that change a program's state. It focuses on describing how a program works: it is made up of commands for the computer performances.

2.3 Editors

For coding in Ruby, it has been chosen the *RubyMine* editor, downloadable from WebStorm platform [48]. It is released by the JetBrains Company. Besides Ruby, it supports JavaScript, CSS and many other languages.

Finally, it allows VCS and so the possibility to manage the project on a repository (for example GitHub); in this way more people can work on the same project and at the same time.

For the second part of this project it has been chosen *RubyMine* to write some JavaScript and HTML code, in order to use the d3.js library.

2.4 XWiki

XWiki [49] is an application wiki[26]. It is a software platform written in Java, which allows storing structured data. It is also possible to execute script within the wiki interface. Users can query data and instances using either Hibernate query language or XWiki's own query language.

In this project XWiki is used as a database which stores all learning scenarios[27]. A Ruby script is used to download this data and to write them into a Json file.

2.5 CMap

CMap can be downloaded from the Cmap web page [50]. It is a tool developed by HICM[28], a research institute of the State University System of Florida. It allows users to create, navigate and share concept maps. CMapServers are the dedicated servers on which CMaps can be shared and edited by more people at the same time. The tool is easy to understand, it facilitates the visualization of the information giving the user the

[26] Wiki is a collaborative software that allows users to cooperate in editing pages or entries through a web browser.
[27] Learning scenarios are stored on https://wiki.esit4sip.eu/bin/view/Scenarios and on https://wiki.esit4sip.eu/bin/view/Main/Spaces.
[28]HICM stands for Institute for Human & Machine Cognition.

opportunity to choose text formatting or boxes colours. In addition to this, it supports different import/export formats. For example, it is possible to export into pdf, image (jpeg), postscript, scalable vector graphic, html, text, xtm (a specialization of xml language), life map files (a txt file which keeps the positions but not the colours of the boxes) and naturally CMap files. The options for the import are text, life map, xtm and CMap.

In connection with this project, CMap has been used to display a complete concept map as reference. This map merges the ontology about learning, the ontology about learning domains and the ontology about infrastructures. Moreover, it has been used at the end, to display the concept maps resulting from the comparison between the map of each scenario and the map of reference.

2.6 D3.js

D3.js[29] is a JavaScript library which allows to create dynamic and interactive graphical representations of data. This data can be visualised with HTML, SVG, or CSS.

In this project, the library has been used to create concept maps from data collected in Ruby environment and written in JSON format. Specifically, data contained: a global concept map of all the ontology with the names of the scenarios on the corresponding node; a global concept map of all the devices and the scenarios related to them; a global concept map of all the domains (or subjects) and the scenarios related to them; a concept map for each device connected to the scenarios which use that specific device; finally, a concept map for each device connected to the scenarios which are related to the specific device. All the nodes corresponding to one scenario, when clicked, redirect to the page of the scenarios on the eSIT4SIP site.

[29] D3 stand for Data-Driven Documents.

2.7 Additional information

All the documentation and the implementation about the eSIT4SIP project is available on GitHub [51].

The following scheme has the purpose of clarify the structure of the eSIT4SIP project:

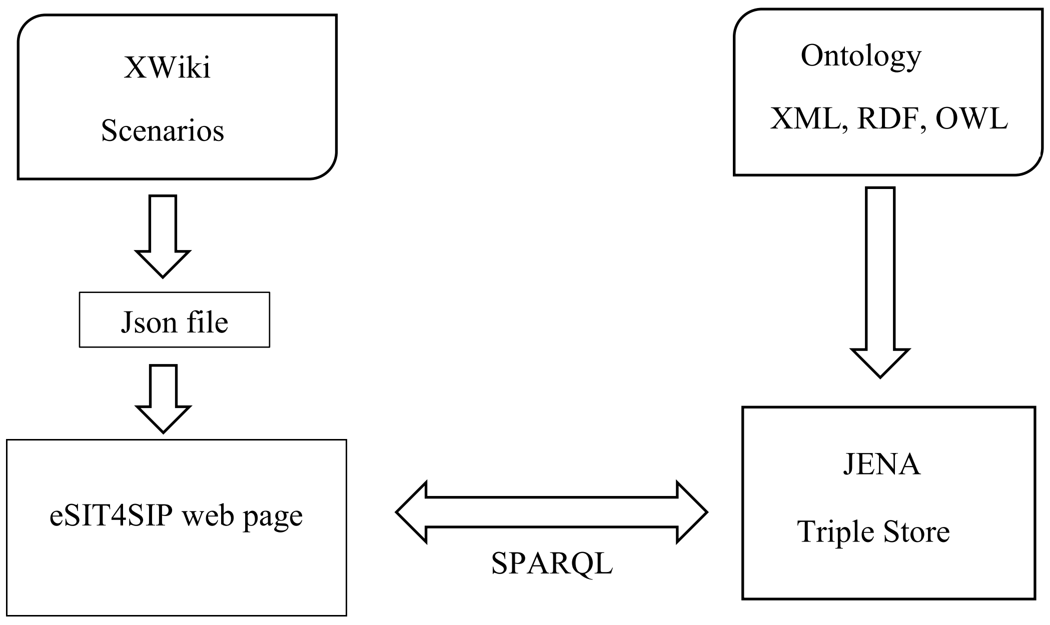


Figure 3. Structure of the eSit4SIP Project

As already written before, RDF is the file extension for storing metadata data-model. Data from RDF files are usually stored in a triplestore: a kind of graph-database storing semantic facts.

In the eSIT4SIP project, the ontologies (learning ontology, domains ontology and infrastructures ontology) are written in rdf/xml and stored on Apache Jena Fuseki [52] (triple store). Jena is a Java framework for building Semantic Web applications; it offers an environment for RDF, RDFS and OWL, SPARQL and provides a rule-based inference engine.

At the same time, learning scenarios are downloaded from *LehrerOnline* and *MnSTEP* platforms or they are written manually, and then they are uploaded on XWiki environment where some manual editing can be performed by the authors of the project.

At this stage, data are downloaded from XWiki and then stored in the json[30] file "articles.json". JSON [53] is a data interchange format based on two structures:

- A cluster of name/value pairs;
- An ordered list of values

The scenarios from the json file are thus uploaded on the eSIT4SIP web page, which is a static page, as already mentioned in the previous chapters.

Finally, SPARQL is the query language used to load dynamically one scenario data on www.eSIT4SIP.eu, while Apache Fuseki server provides a SPARQL endpoint. Http-requests are used to query the server and they return scenario data.

SPARQL, downloadable from the W3.org web site [54], is a query language for RDF files, developed by W3C. It uses the Turtle syntax, an N -Triples extension. There are two possible ways to use SPARQL:

- Locally, for example bound to a programming environment;
- Remotely, over the network or into a database (this is the case of the eSIT4SIP project).

3. Implementation

In this section there will be a detailed description of all the implementation steps. Firstly, since the ontology provided was written partly in English and partly in German, a proper translation of German words has been needed. It has been done manually, under the supervision of people who created the ontologies. Furthermore, it has been necessary to incorporate the ontologies in a single map. It has been done manually too, adding the missing concepts directly in the CMap visualization.

[30] Json is the acronym of JavaScript Object Notation.

The following figure shows a portion of the learning ontology map provided:

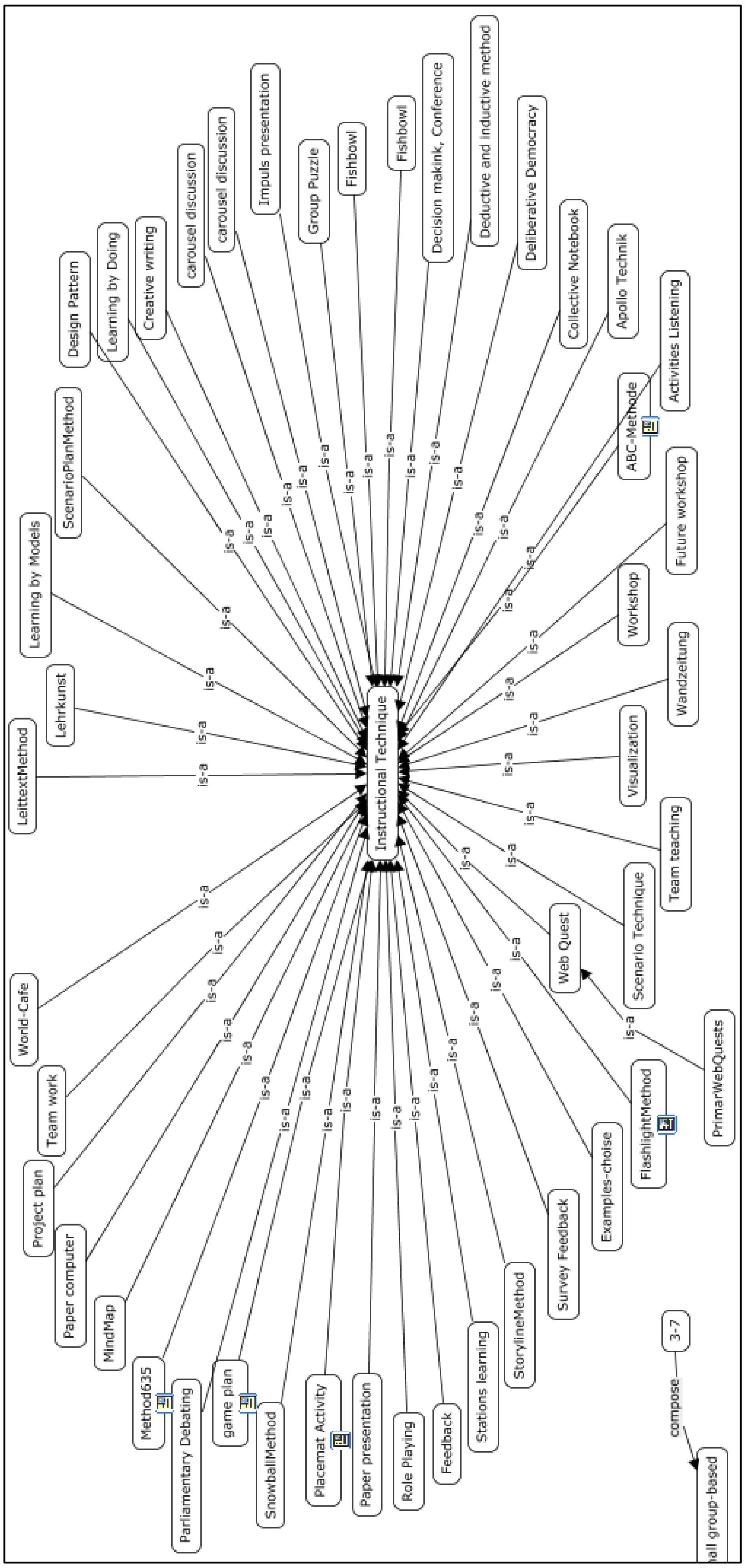
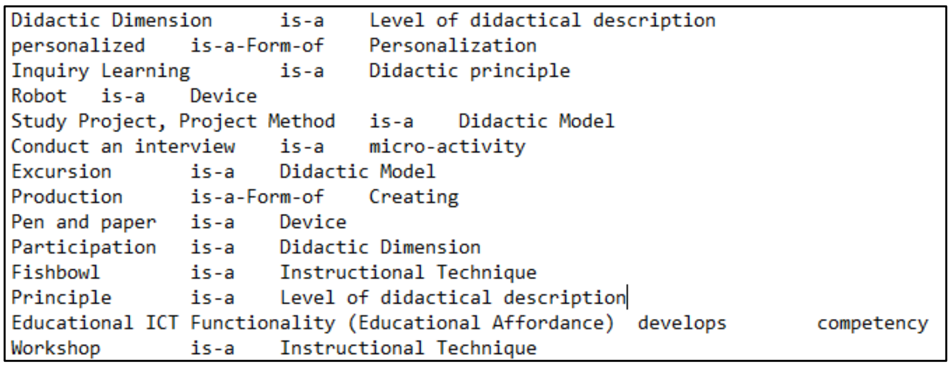


Figure 4. Portion of the Learning Ontology

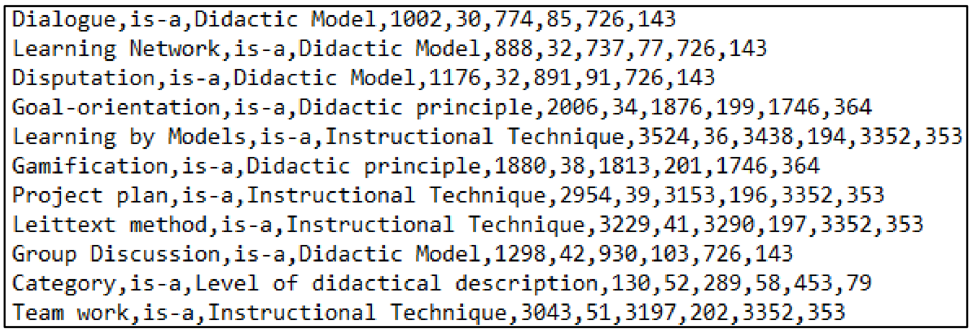
Once the complete map was ready, it has been exported as a text file called "LearningOntology.txt". The same map has also been exported as the LifeMap file "Learning OntologyPos.txt": in this way all the positions of nodes were recorded.

The figures below show a part of the two text files containing the exported ontology:



```
Didactic Dimension      is-a    Level of didactical description
personalized    is-a-Form-of    Personalization
Inquiry Learning        is-a    Didactic principle
Robot   is-a    Device
Study Project, Project Method   is-a    Didactic Model
Conduct an interview    is-a    micro-activity
Excursion       is-a    Didactic Model
Production      is-a-Form-of    Creating
Pen and paper   is-a    Device
Participation   is-a    Didactic Dimension
Fishbowl        is-a    Instructional Technique
Principle       is-a    Level of didactical description
Educational ICT Functionality (Educational Affordance)  develops        competency
Workshop        is-a    Instructional Technique
```

Figure 5. Result of the export as a text file



```
Dialogue,is-a,Didactic Model,1002,30,774,85,726,143
Learning Network,is-a,Didactic Model,888,32,737,77,726,143
Disputation,is-a,Didactic Model,1176,32,891,91,726,143
Goal-orientation,is-a,Didactic principle,2006,34,1876,199,1746,364
Learning by Models,is-a,Instructional Technique,3524,36,3438,194,3352,353
Gamification,is-a,Didactic principle,1880,38,1813,201,1746,364
Project plan,is-a,Instructional Technique,2954,39,3153,196,3352,353
Leittext method,is-a,Instructional Technique,3229,41,3290,197,3352,353
Group Discussion,is-a,Didactic Model,1298,42,930,103,726,143
Category,is-a,Level of didactical description,130,52,289,58,453,79
Team work,is-a,Instructional Technique,3043,51,3197,202,3352,353
```

Figure 6. Result of the export as Life Map

In figure 5 there are three columns: in the left column there are the concepts (or subjects), in the central one there are the relationships (or predicates), in the right column there are the objects (that can be in their turn subjects).

In figure 6, there are nine columns separated by commas. The first three columns are equals to the ones in figure 5, while the others indicate the position: they must be read in pairs, each couple contains the coordinates of one subject (or one predicate or one object) starting from the top left corner.

Secondly, it has been written some code to download articles from the *XWiki* environment and to write them into a Json file. Once being able to read both files (the one with scenarios and the one with ontological map), a script has been written to find which topics were covered or which infrastructures were needed by each scenario. To this end, three scripts have been written:

1. *screeping.rb;*
2. *jsonToTextConverter.rb;*
3. *matchesResearch.rb.*

The last step was trying to provide a graphical overview of all results. To do this, *mapsWriter.rb* has been written. At this stage, every script will be described more in detail.

3.1 *screeping.rb*

The *screeping.rb* allows to download the learning scenarios from the XWiki environment in which they are stored in (in the case of the eSIT4SIP project is https://wiki.esit4sip.eu/bin/view/Scenarios). Firstly, some gems are required: *json* gem enables to read and write json files, *date* gem enables to manage dates, *logger* gem enables to output messages, *typhoesus* gem runs HTTP request, *nokogiri* gem is a reader parser and which allows to search objects through CSS selectors, *mechanize* gem allows automatic interaction with websites, for example it stores and sends cookies automatically and follows redirects. In order to download scenarios from the XWiki environment it is necessary to provide a login, and so the username and the password of the user registered on XWiki. Besides the variables to login in XWiki environment, at the beginning of the script other variables have been defined: the name of the XWiki Space from which downloading the scenarios, the name of the file in which saving the downloaded scenarios (in this project "scenarios.json"), and the name of the directory in which saving the output files (in this case "screnarios_screeped").

At this stage, some methods called later have been defined. In this paper, only the most significant ones are reported.

Scraper_get_page is a method that returns a GUI page as a mechanize-node set and it gets as input a string (the url pointing to a XWiki space), a mechanize object, an integer (the number of previous trials, used for recursion: in fact, if parsing of page fails, it is retried after a certain time. This is possible thanks to the method *timer* which gets as input an integer, that represents the time to wait for, and calls the function *sleep*) and another string used as a flag.

The *get_page_by_url* method gets as input several strings: two of them are the credentials to login in XWiki, one is the url of the web page, another one is the type of encoding of the objects to download (e. g. json, xml) and the last one is a string to make login dynamically. The method makes a HTTP request through *typhoesus* gem and it returns the body of the page as a json format.

The *parse_json* method allows to convert a json string into a hash table. In this project it is used to save different information from scenarios into a hash, e. g. using *id, title*, *content,* etc as keys of the hash.

Furthermore, *write_file* is a method which allows to write some content into a file in a specific directory. It gets as input the string of the file name, the string with the content to save, the string of the directory in which the file must be saved and the type of encoding (in this project UTF-8).

The method *read_file* gets as input a file and its encoding and returns the file read. It is useful when the file to read is not encoded as UTF-8.

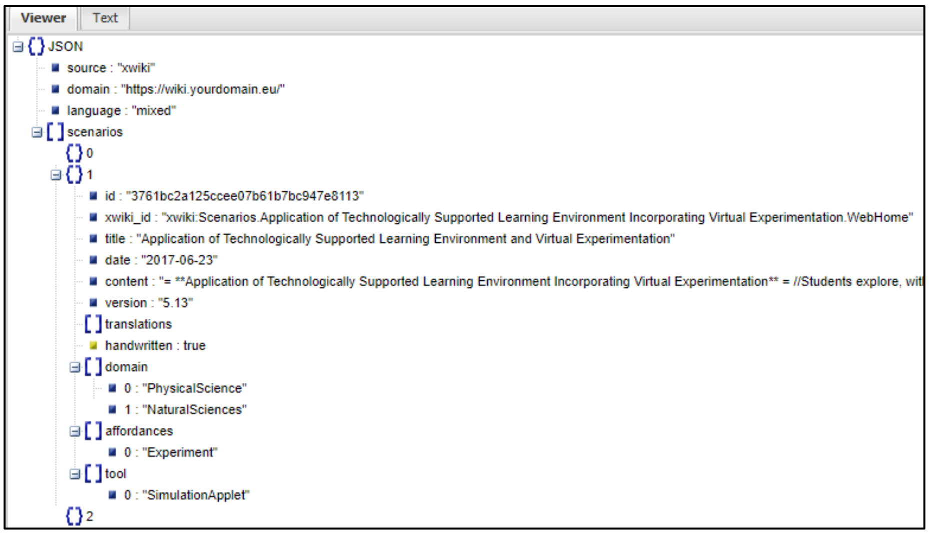
The *scenario_to_file* method saves the GUI page of one scenario to file. Firstly, it calls the method *scraper_get_page* and from the result of this method gets the file content. Specifically, it requires the CSS object called "#xwikicontent". Secondly, it calls the *write_file* method to write the content with HTML file extension.

To collect all relevant information of a scenario page, the *collect_article* method is used. This method saves all information into a hash, it saves *id, xwiki_id, title*, *date* and *content*.

Only at this point the main program of the script *screeping.rb* begins. A *mechanize* object that bypasses SLL verification is defined. Then, the username and the password are checked. At this point, the scenarios are firstly saved into an array: for each scenario *collect_article* method is called, and the result is appended to the array. Later, the array is parsed to *json* and the method *write_file* is called to write the file "scenarios.json" in the defined directory.

The file "scenarios.json" contains a hash with *source, domain, language* and *scenarios* as keys. The value corresponding to the key *scenarios* is the array that contains *id*, *title*, *date*, *content* and other information for each scenario.

Figure 7 shows “scenarios.json” displayed through JSON Viewer [55] with the second scenario opened:



```
{} JSON
    source : "xwiki"
    domain : "https://wiki.yourdomain.eu/"
    language : "mixed"
  [] scenarios
      {} 0
      {} 1
          id : "3761bc2a125ccee07b61b7bc947e8113"
          xwiki_id : "xwiki:Scenarios.Application of Technologically Supported Learning Environment Incorporating Virtual Experimentation.WebHome"
          title : "Application of Technologically Supported Learning Environment and Virtual Experimentation"
          date : "2017-06-23"
          content : "= **Application of Technologically Supported Learning Environment Incorporating Virtual Experimentation** = //Students explore, wit
          version : "5.13"
          [] translations
          handwritten : true
        [] domain
            0 : "PhysicalScience"
            1 : "NaturalSciences"
        [] affordances
            0 : "Experiment"
        [] tool
            0 : "SimulationApplet"
      {} 2
```

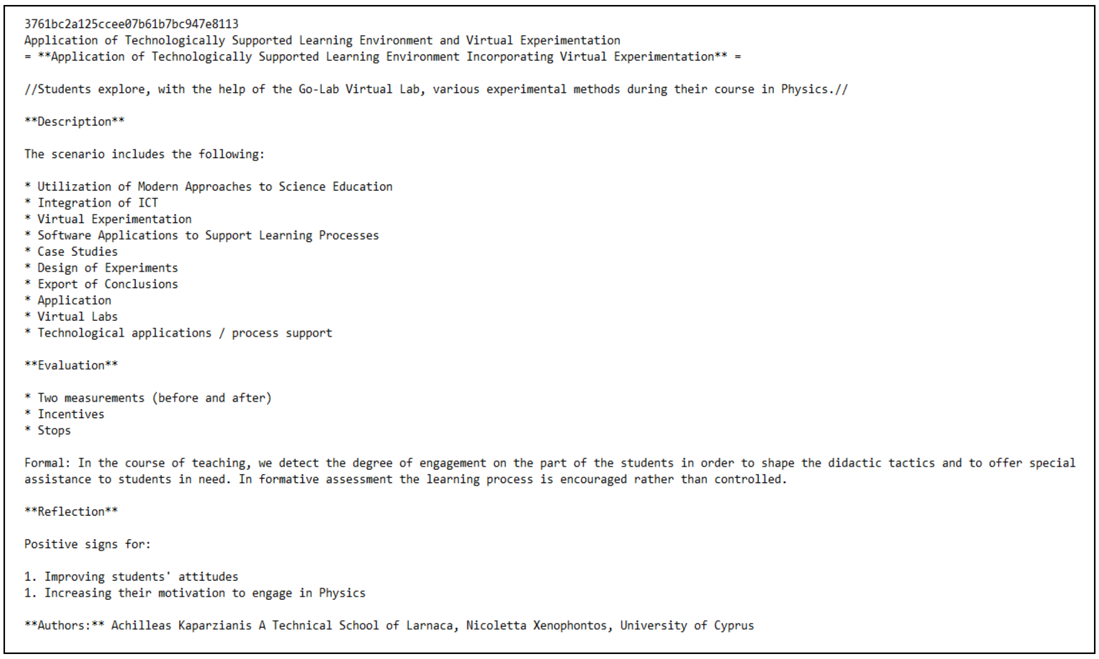
Figure 7. Portion of "scenarios.json" file

3.2 *jsonToTextConverter.rb*

The second script, *jsonToTextConverter.rb,* is a very short script that requires only the *json* gem. It reads the scenarios from "scenarios.json" file and writes each scenario into a separate text file. During this stage, some unnecessary information for this project is lost, more specifically the *xwiki_id*, the *date*, the *version*, the *translations* and the *handwritten* variable are not copied.

At the beginning of this script the directory in which text files must be saved is defined. Furthermore, the only method required is defined: as well as in the previous script, this method is *write_file.* Then, "scenarios.json" file is read and parsed into a hash. Only the value that corresponds to the key *scenarios* is extracted and it is saved as an array. At this point, it is called a cycle which defines the name and the content of the file to write for each element of the array: the former is “scenario” followed by a number (it starts from 0 and it is increased in each iteration), the latter is the union of the *id*, the *title*, the *content*, the *domain*, the *affordances* and the *tool*. Therefore, at the end of each iteration, the cycle calls the method *write_file.*

In figure 8, an example of one the scenarios:



```
3761bc2a125ccee07b61b7bc947e8113
Application of Technologically Supported Learning Environment and Virtual Experimentation
= **Application of Technologically Supported Learning Environment Incorporating Virtual Experimentation** =

//Students explore, with the help of the Go-Lab Virtual Lab, various experimental methods during their course in Physics.//

**Description**

The scenario includes the following:

* Utilization of Modern Approaches to Science Education
* Integration of ICT
* Virtual Experimentation
* Software Applications to Support Learning Processes
* Case Studies
* Design of Experiments
* Export of Conclusions
* Application
* Virtual Labs
* Technological applications / process support

**Evaluation**

* Two measurements (before and after)
* Incentives
* Stops

Formal: In the course of teaching, we detect the degree of engagement on the part of the students in order to shape the didactic tactics and to offer special
assistance to students in need. In formative assessment the learning process is encouraged rather than controlled.

**Reflection**

Positive signs for:

1. Improving students' attitudes
1. Increasing their motivation to engage in Physics

**Authors:** Achilleas Kaparzianis A Technical School of Larnaca, Nicoletta Xenophontos, University of Cyprus
```

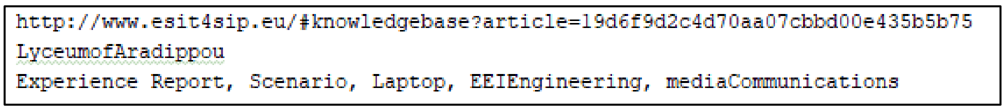
Figure 8. Example of "scenario1.txt"

3.3 *matchesResearch.rb*

Once all the scenarios have been written in a separated text file, the *matchesResearch.rb* script is launched. This script gets as input the text file of the scenarios and the text file of the ontology. From the ontology file, all unnecessary information is deleted, particularly mathematical symbols, empty spaces, strings describing relationships (is-a, is-a-Form-of, etc.). Then the file is split into an array of words which contains the concepts. Thus, for each scenario text file, a search for string matches is performed. Every time a match is found, it is saved in an array ("occurrences") and at the end of the cycle doubles are removed. Finally, the function *write_file* is called, it gets as input the file name (the string "scenario" followed by an incremental number starting from 0, and by the string "occurrences"), the content of the file (the first line is the *id* of the scenario, the second line is the title and the third line is the array of occurrences converted to string), the directory in which saving files ("Matches") and the encoding (UTF-8).

The directory "Matches" where outputs are saved were declared at the beginning of the script as well as the part of the path in common among all the scenarios text files.

In figure 9, is displayed the output for the “scenario1.txt”:



```
http://www.esit4sip.eu/#knowledgebase?article=19d6f9d2c4d70aa07cbbd00e435b5b75
LyceumofAradippou
Experience Report, Scenario, Laptop, EEIEngineering, mediaCommunications
```

Figure 9. Output “scenario1_occurrences.txt”

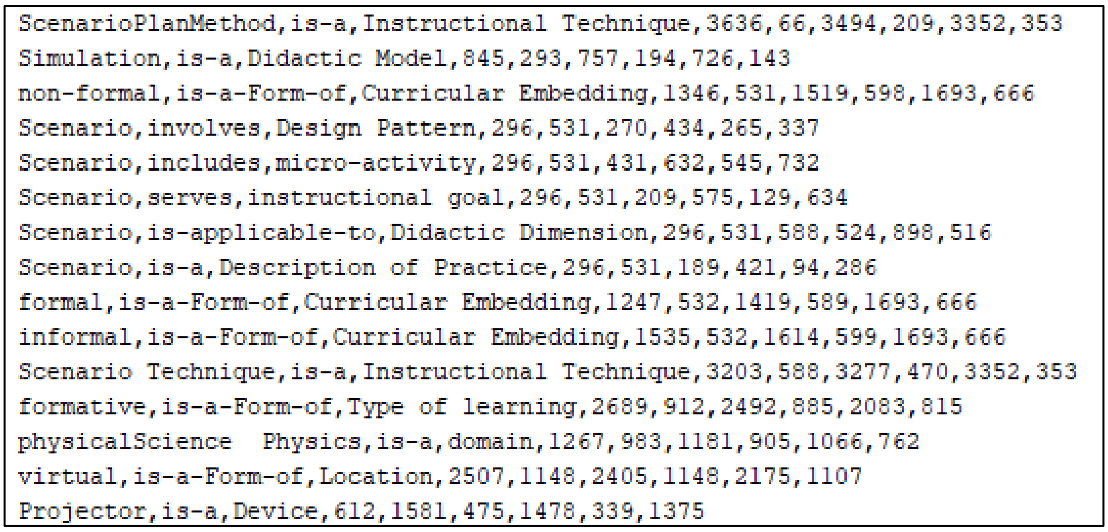
Initially, in the output was saved only the array with the occurrences, but then, when it became necessary also to store information about title and *id* (the latter is necessary to build links in the final concept maps), these fields have been added in the first two lines of each file.

3.4 *mapsWriter.rb*

At this point, the *mapsWriter.rb* script is launched. As well as the script before, at the beginning of this script are defined the directory in which saving the output (“Maps”) and the path in common among the files containing the matches of the different scenarios (“../Matches/scenario”). In this case, instead of the ontology exported as a text file, it has been utilised the one exported as LifeMap; it is always a .txt format, but it allows to keep the position of the nodes in the map. Then, two functions are defined, the former is *write_file* function (the same used in the scripts before), the latter is called *print_maps* and it is explained as follows.

The *print_maps* function gets as input the path where is saved the text file containing a certain scenario matches and an index. It is called by the main function for every file contained in the “Matches” directory and at every iteration the index (which starts from 0) is increased. In the *print_maps* function, an empty variable “file_data” is declared, the file with matches of one scenario is opened in the readout mode, and then the line containing matches (the third line of each file) is converted to an array type and its letters are converted to downcase. At this stage, the text file with the ontology is opened and for each line is performed a search for matches among the line and each word of the occurrences: when a match is found, the all line is kept and added to the “file_data”. At the end of the cycle, the *write_file* function is called to write, for each scenario, a text file containing information to build the concept map.

In figure 10 is shown the output from the *writeMaps.rb* script applied to “scenario1_occurrences.txt”:



```
ScenarioPlanMethod,is-a,Instructional Technique,3636,66,3494,209,3352,353
Simulation,is-a,Didactic Model,845,293,757,194,726,143
non-formal,is-a-Form-of,Curricular Embedding,1346,531,1519,598,1693,666
Scenario,involves,Design Pattern,296,531,270,434,265,337
Scenario,includes,micro-activity,296,531,431,632,545,732
Scenario,serves,instructional goal,296,531,209,575,129,634
Scenario,is-applicable-to,Didactic Dimension,296,531,588,524,898,516
Scenario,is-a,Description of Practice,296,531,189,421,94,286
formal,is-a-Form-of,Curricular Embedding,1247,532,1419,589,1693,666
informal,is-a-Form-of,Curricular Embedding,1535,532,1614,599,1693,666
Scenario Technique,is-a,Instructional Technique,3203,588,3277,470,3352,353
formative,is-a-Form-of,Type of learning,2689,912,2492,885,2083,815
physicalScience  Physics,is-a,domain,1267,983,1181,905,1066,762
virtual,is-a-Form-of,Location,2507,1148,2405,1148,2175,1107
Projector,is-a,Device,612,1581,475,1478,339,1375
```

Figure 10. Output “Map_Scenario1.txt”

As can be seen from figure 10, in the first column there are the concepts, in the second column there are the relationships, in the third column there are the nodes the concepts are linked to and in the last six column there are the information about the position of nodes in the original concept map of the learning ontology.

To summarise these steps, the following scheme is provided:

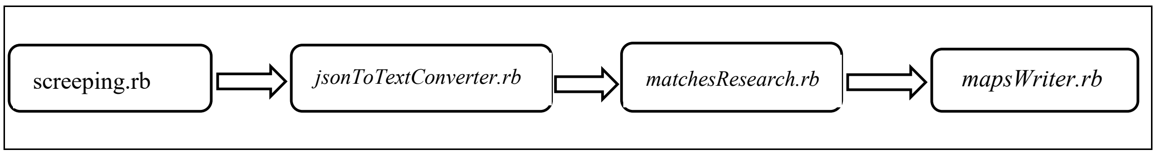


Figure 11. Steps followed

The files created in this way have then been imported in CMap as LifeMap files, and the figure 12 shows a part of the map for the scenario1.

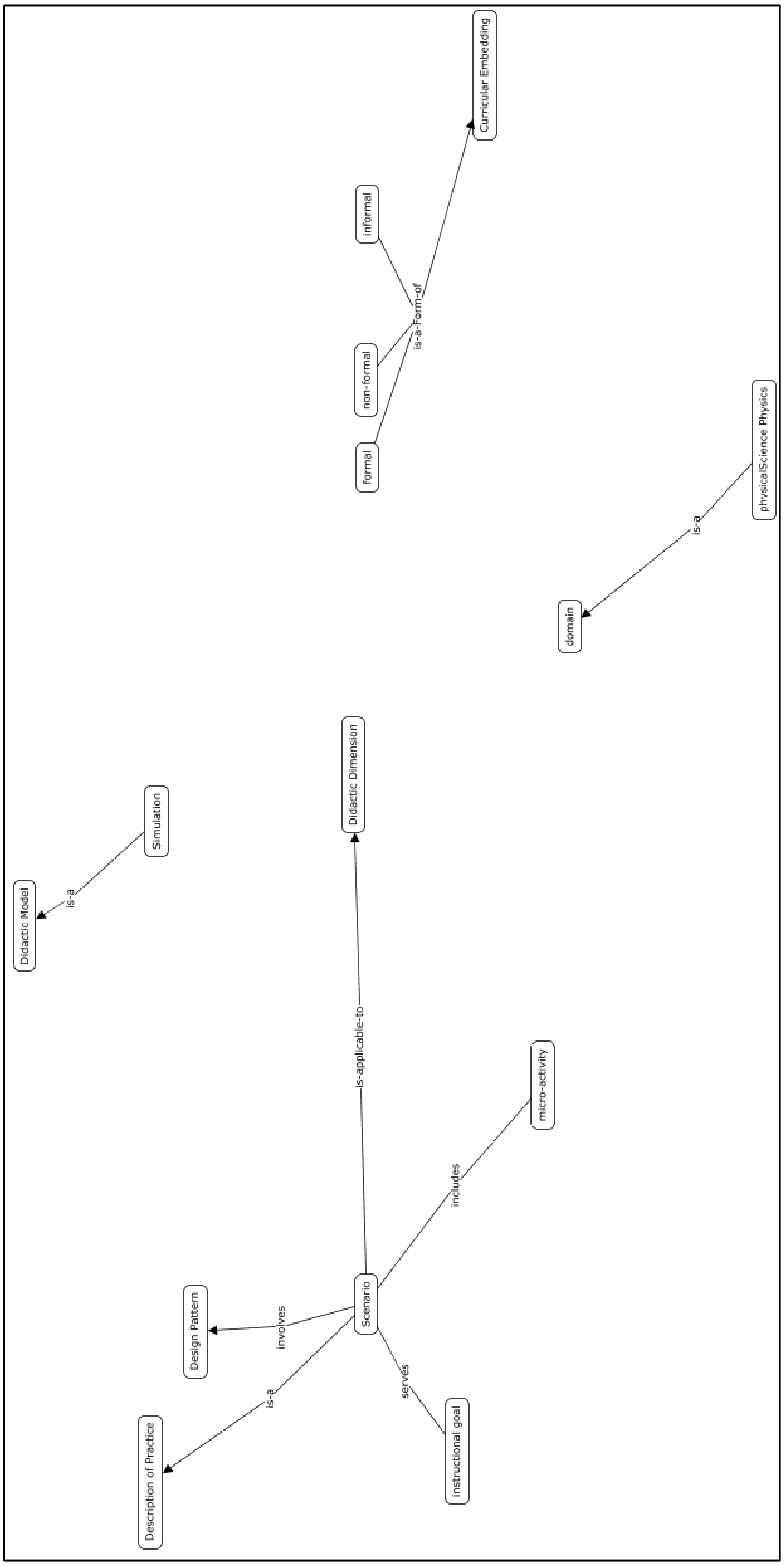


Figure 12. Portion of the result of “Map_scenario1.txt” imported as LifeMap format

What has come out is that maps created in this way provided a partial overview of the scenarios. Specifically, the maps contain the nodes of the words resulting from the matching between the scenario and the ontology and contain also the nodes linked to them, but without any specification of which were the words found in the scenario and which the ones only linked. Since CMap software does not allow to specify the layout of nodes or save this kind of information in the text file exported, every kind of editing on the maps can be performed only in the CMap environment, except for the position of the nodes. On the other side, the maps such created present a structure where not all nodes are linked: there are more different groups of nodes linked to each other but then there is no connection between these groups. This is an obstacle in reading and navigating the maps. In addition to this, there is the fact that CMap does not provide an automatic distribution of nodes in the space, so it might happen that some nodes overlap if the user does not specify the position of each node, and this is a limitation, for example, when creating new nodes. Finally, CMap does not allow to attach links to the nodes, so the navigation of the maps can be only a static visualisation.

Even though the concept map keeps the position of nodes from the starting map (learning ontology map) and it is easy to compare the new maps with the original one for example by overlapping them, this is not yet the kind of experience that it was intended to provide. For these reasons, it has been decided to improve this work using the JavaScript library d3.js.

The aim was to create a full learning ontology map and to add nodes representing learning scenarios linked to the concepts they covered, this kind of information is stored in the text files of the “Matches” directory. Then, the following step was to create a d3 map for each device and a d3 map for each domain, always with the proper scenarios connected.

To do this, it was first necessary to create files of data to be read by the library and the format chosen to write data was JSON. Secondly, it was necessary to have a skeleton of the complete map, with the most important concepts presented and already linked: this file “Ontology_complete.txt” has been written manually, starting from the text file of the learning ontology map and deleting the less important concepts according to the information available in the scenarios. In fact, since the scenarios are examples of lectures it is quite difficult to find some words referring to specific kind of didactic model or cognitive processes involved, as well as for the type of learning tool or instructional techniques and class organization. In most cases all this information can be extracted by the context and not from a string matching.

3.5 *mapAppending.rb*

The script to create data and to add them to the complete map is *mapAppending.rb.* In the first lines of the script it is required the *json* gem, it is defined the path from which getting information about scenarios matches, it is opened and parse to Json the text file “Ontology_complete.txt” and saved into the variable “ontology”. Then, a new function is defined: it is called *addLine,* it gets in input the string with the link to the web page of the scenario required, the string of the title of the scenario, the string of the word matching and, finally, the file of the ontology where a new node had to be added. When it is called, it creates a hash composed by three keys and three values (like the one in the example below) and it adds the hash to the array “ontology”.

```
{"source" => title of scenario, "target"=> word matching, "value" => link of the scenario}
```

Figure 13. Example of hash added to the ontology

The main method of *mapAppending.rb* script is a cycle which saves for each file contained in the “Matches” directory the first line of the file in the variable “link”, the second line in the variable “title” and the third line in the variable “lineFile”. This last variable, that is a string, is then converted into an array by splitting the content every “,”. Then, for each element of the array such created it is made a check into the ontology. Every time a match is found, it is called the *addLine* function. At the end of this script, the ontology with the new lines added is parsed to JSON and it is called the ruby function *write* to create a json file for the new complete ontology.

Before seeing the part concerning the graphical representation of these data with HTML and JavaScript, it is important to explain how the other data are obtained.

For the extraction of data to create a map with all domains, the script *domainMap.rb* has been created. Whereas for the extraction of data to create a map with all devices, the script *deviceMap.rb* has been create.

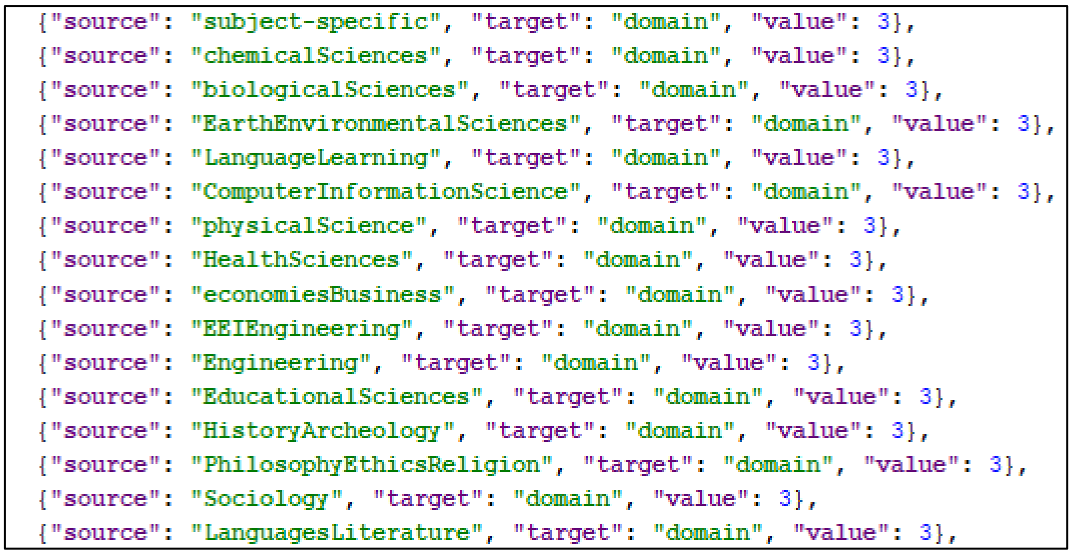
These two scripts are very similar to each other and they are similar to the previous script. The essential difference is the referential ontology: while in the *mapAppending.rb* script the complete ontology was manually written to keep only the key concepts, in the *domainMap.rb* script was written only the part of the ontology containing the different types of domains; in the *deviceMap.rb* script, instead it was only the part of the ontology containing the different types of devices to be written. Apart from these differences, the scripts are the same.

Actually, as will be seen below, these scripts do not provide ideal data for the representation because, since the ontologies (the full ontology, the one with only domains and the one with only devices) are themselves very big, when a new node is added for every match, the result is a very disordered and chaotic map, difficult to understand, read and navigate.

3.6 *mapForEachDomain.rb* and *mapForEachDevice.rb*

To collect data for the concept maps of each domain and of each device, have been written the scripts *mapForEachDomain.rb* and *mapForEachDevice.rb.* Since they are the same, except for the starting ontology (ontology of domains in the first case, ontology of devices in the other), only the *mapForEachDomain.rb* will be explained in detail.

Firstly, a part of the ontology with the only domain is shown in figure 14:



```
{"source": "subject-specific", "target": "domain", "value": 3},
{"source": "chemicalSciences", "target": "domain", "value": 3},
{"source": "biologicalSciences", "target": "domain", "value": 3},
{"source": "EarthEnvironmentalSciences", "target": "domain", "value": 3},
{"source": "LanguageLearning", "target": "domain", "value": 3},
{"source": "ComputerInformationScience", "target": "domain", "value": 3},
{"source": "physicalScience", "target": "domain", "value": 3},
{"source": "HealthSciences", "target": "domain", "value": 3},
{"source": "economiesBusiness", "target": "domain", "value": 3},
{"source": "EEIEngineering", "target": "domain", "value": 3},
{"source": "Engineering", "target": "domain", "value": 3},
{"source": "EducationalSciences", "target": "domain", "value": 3},
{"source": "HistoryArcheology", "target": "domain", "value": 3},
{"source": "PhilosophyEthicsReligion", "target": "domain", "value": 3},
{"source": "Sociology", "target": "domain", "value": 3},
{"source": "LanguagesLiterature", "target": "domain", "value": 3},
```

Figure 14. Part of the file Ontology_domain.json

The *mapForEachDomain.rb* script begins with the *json* gem requirement. Then, it is defined the path of the directory from which loading data to elaborate (../Matches) and the JSON file containing the ontology of the domains is loaded, parsed and stored in the variable “ontology”.

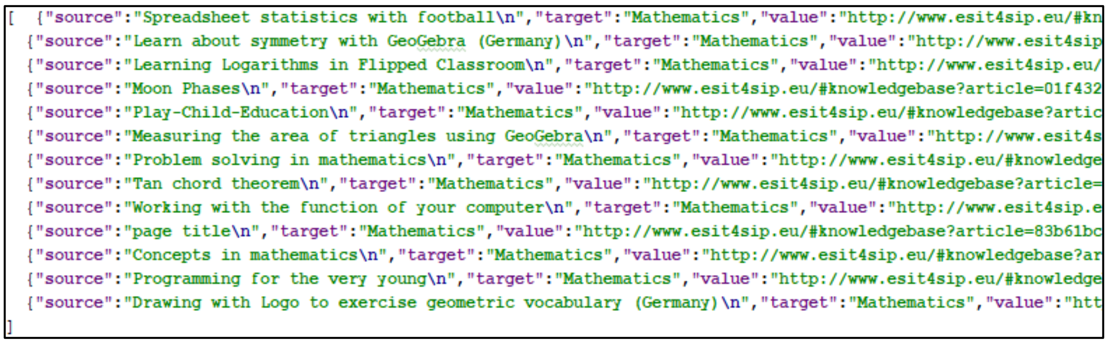
At this point, a cycle allows to create a new empty json file for each element (and so for each domain) of the variable “ontology”. These files will be filled with the scenarios that cover the specific domain, stored as a hash whose keys are the title of the scenario, the match (and so the domain covered) and the link to the web page of the scenario on the eSIT4SIP web site.

As well as for the scripts described above, an *addLine* function is defined. In this script, the function firstly gets in input the link, the title, the domain matched and the ontology, and then it creates a hash, converts it into JSON format and appends it to the proper file.

The main method of the script is a cycle that for each file contained in the directory specified at the beginning (that means for each domain) checks whether there is a match within all files contained in the directory “Matches”: every time there is a match, it calls the function *addLine.* At the end of the cycle it is called the function *writeFile* to write for each domain the output in JSON format file.

Since a JSON file begins with “[“, ends with “]” and each statement is followed by “,” (except for the last one before “]”), some editing has to be performed on the resulting files. It is achieved by the *writeFile* function. It gets the ontology file and writes a new file in this way: it adds “[“ at the beginning of the file, reads the last line and removes the “,” after the “]” and finally adds “]” at the end of the file.

In figure 15 it is shown a portion of the output for the domain “Mathematics”:



```
[  {"source":"Spreadsheet statistics with football\n","target":"Mathematics","value":"http://www.esit4sip.eu/#kn
  {"source":"Learn about symmetry with GeoGebra (Germany)\n","target":"Mathematics","value":"http://www.esit4sip
  {"source":"Learning Logarithms in Flipped Classroom\n","target":"Mathematics","value":"http://www.esit4sip.eu/
  {"source":"Moon Phases\n","target":"Mathematics","value":"http://www.esit4sip.eu/#knowledgebase?article=01f432
  {"source":"Play-Child-Education\n","target":"Mathematics","value":"http://www.esit4sip.eu/#knowledgebase?artic
  {"source":"Measuring the area of triangles using GeoGebra\n","target":"Mathematics","value":"http://www.esit4s
  {"source":"Problem solving in mathematics\n","target":"Mathematics","value":"http://www.esit4sip.eu/#knowledge
  {"source":"Tan chord theorem\n","target":"Mathematics","value":"http://www.esit4sip.eu/#knowledgebase?article=
  {"source":"Working with the function of your computer\n","target":"Mathematics","value":"http://www.esit4sip.e
  {"source":"page title\n","target":"Mathematics","value":"http://www.esit4sip.eu/#knowledgebase?article=83b61bc
  {"source":"Concepts in mathematics\n","target":"Mathematics","value":"http://www.esit4sip.eu/#knowledgebase?ar
  {"source":"Programming for the very young\n","target":"Mathematics","value":"http://www.esit4sip.eu/#knowledge
  {"source":"Drawing with Logo to exercise geometric vocabulary (Germany)\n","target":"Mathematics","value":"htt
]
```

Figure 15. Output "Mathematics.json" for the mathematics domain

3.7 *mapForEachScenario.rb*

A very similar proceeding has been followed to extract data to build a dynamic and interactive map for each scenario. The script which implements this extraction of data is *mapsForEachScenario.rb.* This script contains a main function and four other functions: *addScenarioLink, addLine, fileLength* and *writeFile.* At the beginning of the script, there is the definition of the path from which getting information about scenarios matches, the definition of the path in which writing the output files ("../Maps") and the reading of the ontology (in this case, it is the complete ontology manually edited in order to keep only the main nodes; in this way information like the all kinds of device and domains are deleted).

Then, the main program launches a first cycle which creates for each element in the directory "Matches" a new file and writes the ontology into.

The other cycle, actually a nested cycle, for each file in the "Matches" saves into separated variables the first line (the url link of the scenario), the second line (the title of the scenario) and the third line (matches founded with the *matchesResearch.rb* script). Thereafter, another check with the complete ontology (not the one with the only skeleton) is performed and, every time a match is found, it is called the function *addLine*. This function gets in input the hash corresponding to the line of the ontology in which the match has been detected, and the path of the file in which adding the line (this file is the one created at the beginning and it contains already the ontology skeleton). At the end of each iteration, two functions are called: *addScenarioLink* (it gets in input the link of the scenario, the title, and the path of the file in which adding the line, and adds to the proper file a hash converted to JSON) and *writeFile* (it gets in input only the path of the file to edit and writes a new file from the previous one in order to respect the JSON format; it adds "[" at the beginning and "]" at the end and removes some mismatches but, to do this, it needs to know how long the file is and for this reason it calls the function *fileLength*).

Table 1 summarises the different scripts:

Script	Task	Most important methods	Gems Required
screeping.rb	Downloading learning scenarios from XWiki Environment	1. screeper_get_page 2. get_page_by_url 3. parse_json 4. read_file 5. write_file 6. scenario_to_file 7. collect_articles	1. json 2. *date* 3. *logger* 4. *typhoesus* 5. *nokogiri* 6. *mechanize*
jsonToTextConverter.rb	Writing each scenario into a separate text file	1. write_file	1. json
matchesResearch.rb	Comparing each scenario with the ontology	1. write_file 2. print_words_occurrences	/
mapsWriter.rb	Writing text files of the maps to be imported into CMap	1. write_file 2. print_map	/
mapAppending.rb	Creating data to be added to the complete ontology map	1. add_line	1. json
mapForEachDomain	Collecting data to create concept maps for each domain in the ontology	1. add_line 2. fileLength 3. writeFile	1. json
mapForEachDevice	Collecting data to create concept maps for each device in the ontology	1. add_line 2. fileLength 3. writeFile	1. json
mapForEachScenario	Collecting data to create concept maps for each scenario downloaded from XWiki	1. add_line 2. fileLength 3. writeFile	1. json

Table 1. Scripts overview

3.8 Interactive maps building

Once all data files are ready, it is necessary to switch to HTML programming to be able to create the interactive maps. Since all the HTML files, as well as all the JavaScript files, are the same (except for the input JSON file they get), in the next paragraphs, only the implementation of the ontology map will be described.

The HTML file created presents the following structure:

```
<!DOCTYPE html>
<meta charset="utf-8">
<style>

  .link {
    fill: none;
    stroke: #676767;
    stroke-width: 1px;
  }
  .scenario circle {
    fill: #0eff67;
    stroke: #fff;
    stroke-width: 2px;
  }

  .other circle {
    fill: #0062ff;
    stroke: #fff;
    stroke-width: 2px;
  }

  text {
    font: 10px sans-serif;
    pointer-events: none;
  }

</style>
<body>
<script src="//d3js.org/d3.v3.min.js"></script>
<script type="text/javascript" type="module"
src="fullMap.js"></script>
</body>
```

In the section of the tag "style" there are instructions about the colour of the line linking two nodes, the colour and the stroke of the shape (in this case circles) representing different nodes (the concepts, belonging to the class "other", and the scenarios, belonging to the class "scenario") and the font of the text associated to each node.

The body contains only instructions about the referential scripts: the former ("//d3js.org/d3.v3.min.js") is the d3 library, the latter ("fullMap.js") is the JavaScript file for the implementation and the creation of the graph.

The file JavaScript consists of a main function which is called at the beginning and gets in input the JSON file of data:

```
d3.json("Map.json", function(data) {
// coding
}
```

Inside this function, data obtained from the JSON file are saved into the variable "links", which is an array. Afterwards, a new empty variable "nodes" is declared. Then, there is a cycle to compute the distinct nodes from the elements of the array "links". Specifically, for each element of the hashes contained in the array variable "links" whose key is "source" or "target" a new node is created and two properties are associated to this node: the name and the value (corresponding to the hash key "value" which contains the url link if the node corresponds to one scenario; otherwise it contains a number).

Following, the size of the area on which the maps is displayed (width and height) is defined and some variables to manage the graph are declared. The variable "force" is one of them, it is used to compute the starting position of nodes and to set the distance among them to avoid any overlapping.

Another variable is "svg", it is a scalable vector graphics object, it is given the size defined before and then it is appended to the body.

The variable "link" selects all the "link" object from the "svg" and appends a line. On the other side, the variable "node" selects all the "node" object from the "svg" and associates to them the call to other functions when some events occur. These events are: mouse over, mouse out, mouse click and drag. Furthermore, if each node contains a link to a web page, it is declared to belong to the "scenario" class; otherwise it is declared to belong to the "other" class.

When the mouse is over a node, the function *mouseover* selects the node over which the mouse is and increases its shape with a transition.

When the mouse, after being over, is out, the function *mouseover* selects the node over which the mouse was and brings the size of the shape to its original value.

Finally, when the mouse clicks on one node, the function *click* gets the variable "value" of the node, checks whether it is a URL string (checking whether it includes the “http” string) and, if it is an URL, it calls the function *window.open* which redirects to the web page of the scenario on the eSIT4SIP web site.

On each node a circular shape is appended, as well as a text to display the name of the node (the title of the scenario or the concept from the ontology).

Thank to this library, when the user clicks on one node and drags it within the area of the “svg” object, the positions of all nodes are recalculated in order to avoid overlapping and to respect the distance among nodes.

In figure 16 there is an example of the screenshot displayed:

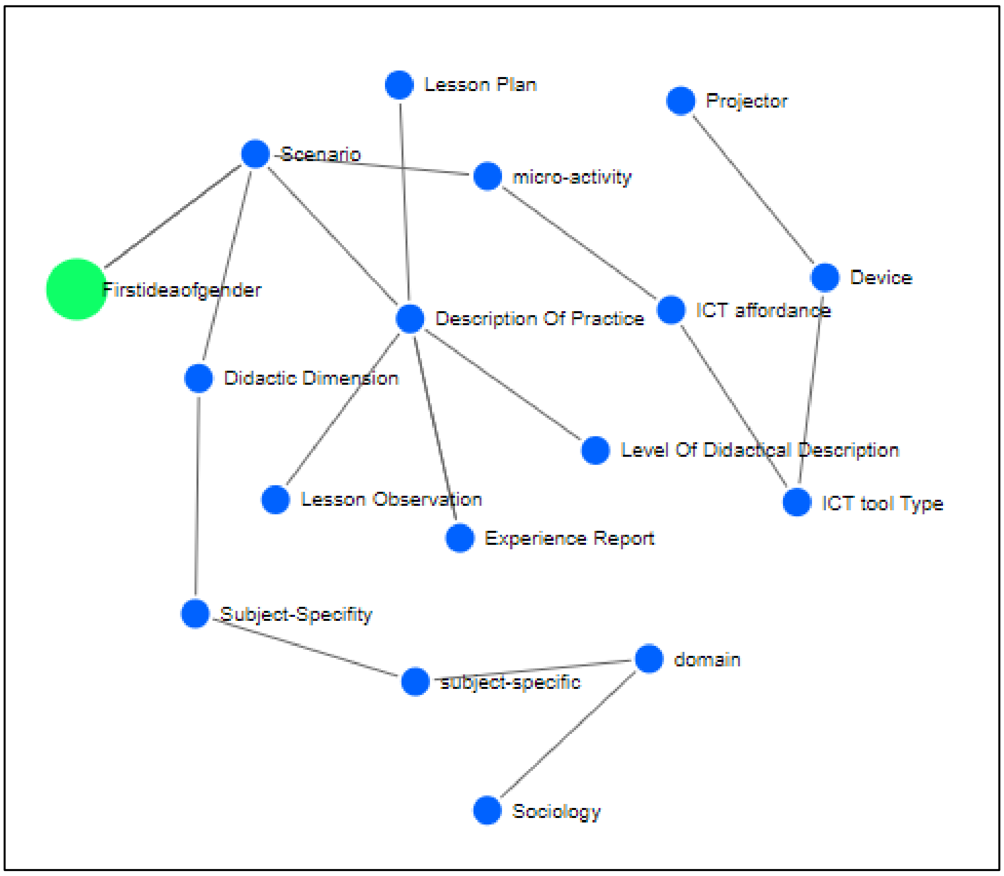


Figure 16. Map of scenario1 created with d3.js

The following scheme aims to clarify the workflow followed:

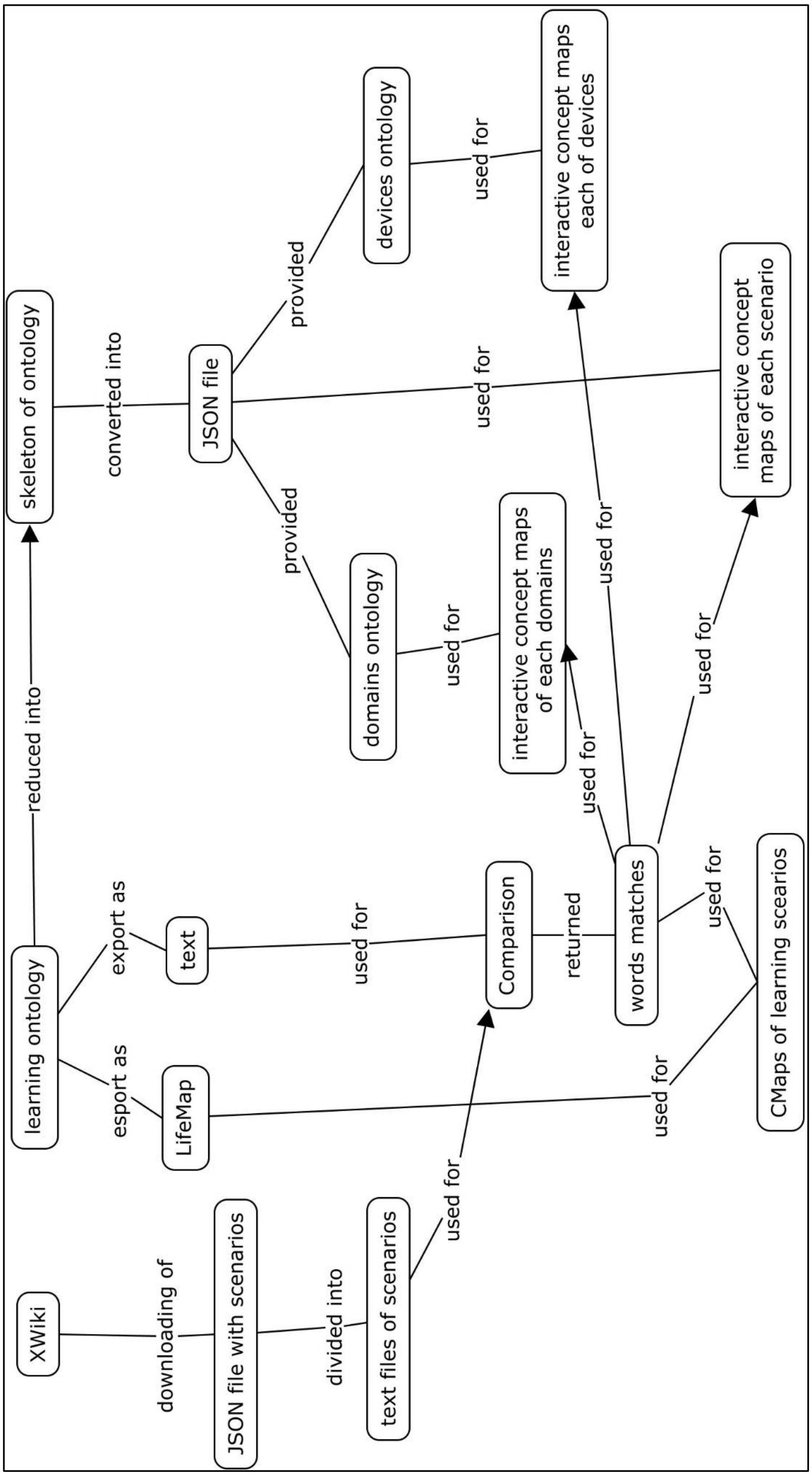


Figure 17. Project overview

4. Results

In this chapter the results are presented. Some of them have already been shown in the previous figures; here a more detailed presentation of them is provided.

Overall, the all work can be subdivided into three parts: downloading of scenarios, matches research and maps building.

4.1 Scraping XWiki

The first part consisted in scraping the scenarios from the XWiki web environment. The Ruby script which performed this task (*screeping.rb*) has achieved satisfactory results. Fifty-seven scenarios have been downloaded from the XWiki space “Scenarios”, including images, tags and other attachments. They have been saved in the file “scenarios.json”, then HTML files of the scenarios have been saved in the “scenario_screeped” directory, as well as the attachments of each scenario (most of all them were images in png or jpeg encoding) have been saved in a separated directory into the “scenario_screeped” directory. The number of the scenarios on the XWiki environment is fifty-seven, the same of the scraped scenarios, furthermore, the content of the HTML files of some random scenarios has been manually compared by the authors of the eSIT4SIP project with the content of the same scenario on the web page of eSIT4SIP to check if the content was the same.

It can be concluded that all the scenarios have been correctly downloaded. Since the scenarios are written into different languages (English, German and Greek) and since the next parts of the work are developed to deal with the English language, it could be useful to develop one more script to translate the different languages of the scenarios into English. Indeed, due to the fact that the contents of the scenarios are compared with the words in the ontology and because the ontology is written in English, for the German or in Greek scenarios (the minority) no match was found and so no map could be provided.

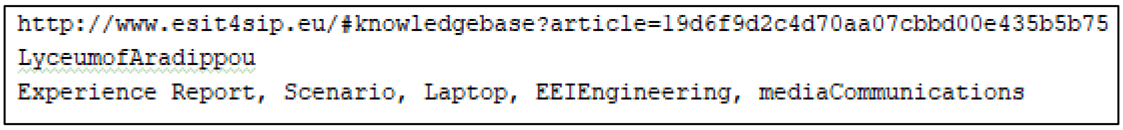
Another issue is the fact that on the eSIT4SIP web page there are 295 scenarios, while on the XWiki space “Scenarios” they are only fifty-seven, this means that not all the scenarios on the web page can have a map associated, but only the ones existing in

XWiki. To improve the project, all the scenarios available on the eSIT4SIP web page should be also present on XWiki.

4.2 Comparison between the ontology and the learning scenarios

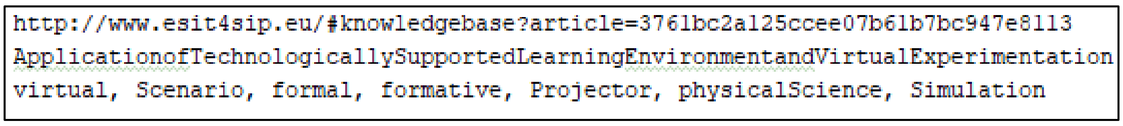
The second part of the work consisted in looking for matches between the ontology and the scenarios. The scripts involved in this task were two: *JsonToTextConverter.rb,* which wrote each scenario into a distinct text file, and *matchesResearch.rb,* which compared the ontology with the content of each scenario and wrote the matches in a distinct text file for each scenario.

The following figures show the output of some scenarios (the first line is the url link to the scenario on eSIT4SIP web page, the second line is the title of the scenario and the third line, if exists, is the array of matches):



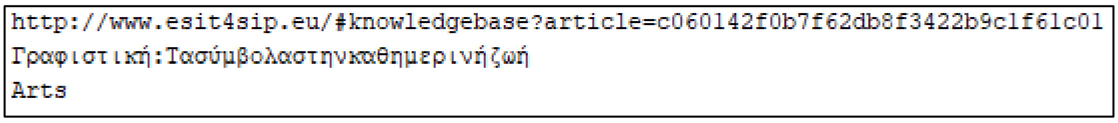
```
http://www.esit4sip.eu/#knowledgebase?article=19d6f9d2c4d70aa07cbbd00e435b5b75
LyceumofAradippou
Experience Report, Scenario, Laptop, EEIEngineering, mediaCommunications
```

Figure 18. Matches scenario0



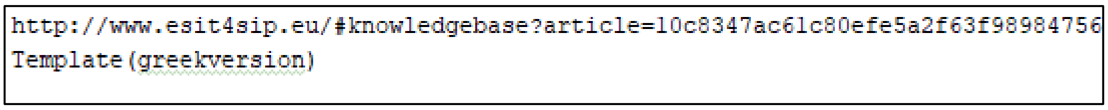
```
http://www.esit4sip.eu/#knowledgebase?article=3761bc2a125ccee07b61b7bc947e8113
ApplicationofTechnologicallySupportedLearningEnvironmentandVirtualExperimentation
virtual, Scenario, formal, formative, Projector, physicalScience, Simulation
```

Figure 19. Matches scenario1



```
http://www.esit4sip.eu/#knowledgebase?article=c060142f0b7f62db8f3422b9c1f61c01
Γραφιστική:Τασύμβολαστηνκαθημερινήζωή
Arts
```

Figure 20. Matches scenario48



```
http://www.esit4sip.eu/#knowledgebase?article=10c8347ac61c80efe5a2f63f98984756
Template(greekversion)
```

Figure 21. Matches scenario56

Firstly, it is worth noting that in the last scenario (figure 21) no matches have been found and this is because the scenario was written in Greek, as well as the one in figure

20, but in contrast to the former, the latter, despite its language, contains the word “Arts” among its tags.

Secondly, it must be said that most of matches found concern a certain type of domain (e. g. Arts, EEIEngineering, mediaCommunications, physicalSciences are the ones found in the examples above) or a certain type of required device (e. g. Projector, Laptop). It is very difficult to find terms which express concepts about learning like the one in the ontology; it happens because while the ontology provides a complete overview of learning including instructional techniques, cognitive processes involved etc., the scenarios are examples of lectures which implicitly practise these concepts without declaring them.

Another problem in the matches research is the fact that it is a non-case-sensitive string matching and so only words written in the same way are found. This is a problem when there are synonyms and paraphrases because they are not detected. For example, if one scenario contains the phrase “students must work alone” and in the ontology there is the concept “individual work”, this is not recognized. The same thing happens if one scenario contains “the student has to learn alone by experiencing himself and by integrating the new knowledge with the one already learned” while in the ontology there is the word “constructivism”. Extending the research to paraphrases is a task which requires a long time and more sophisticated tools (e. g. neural network), on the other side, extending the research to synonyms is something easier and it can be developed with *WordNet* tool [56]. WordNet can be used as a dictionary to find synonyms of the words in the ontology; in fact, the tool groups English words into sets of synonyms and gives definitions and examples of use.

However, the matches found with the *matchesResearch.rb* script have been random manually compared with the study of the scenarios manually carried out by two co-workers of the eSIT4SIP project. Specifically, they read each scenario and saved important information in a .doc file. Most information saved concerns the devices and the domains, then there is some information about the type of activity (individual or group). This comparison showed that in most cases the matches corresponded. Sometimes the .doc file had some more information than the matches (for example, this happened when in the scenario was mentioned a specific device while in the ontology there is only the general category). Other times the .doc file had less information than the matches; it happened when some words were found in the comparison with the ontology but in the manually study they have not been considered because of mistakes or because of their meaning in relation to the specific concept.

4.3 Graphical visualisation

The third part of this work consisted in providing a graphical visualisation of the results, possibly in the form of concept maps. Initially, maps were created following the ontology: all parts of the ontology detected in the scenarios have been kept, the others have been deleted, then the resulting files were imported into CMap.

This kind of approach led to the following results (the figures below are the output corresponding to the maps created from figure 18 and 19):

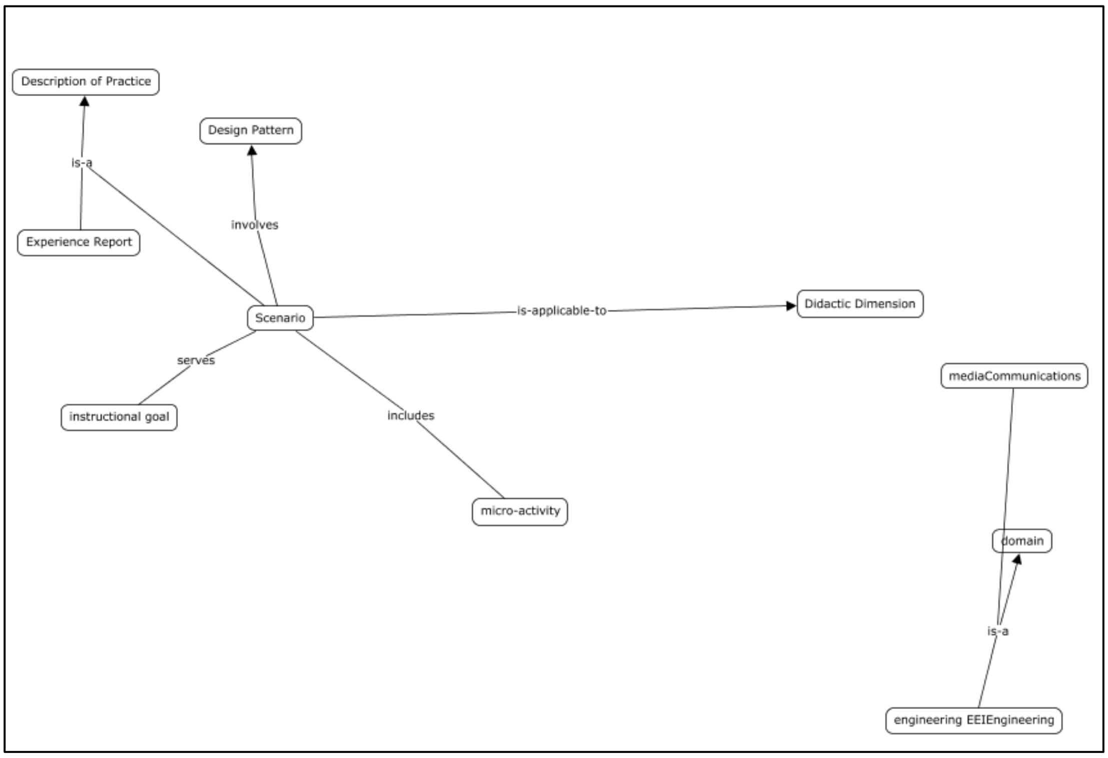


Figure 22. CMap of scenario0

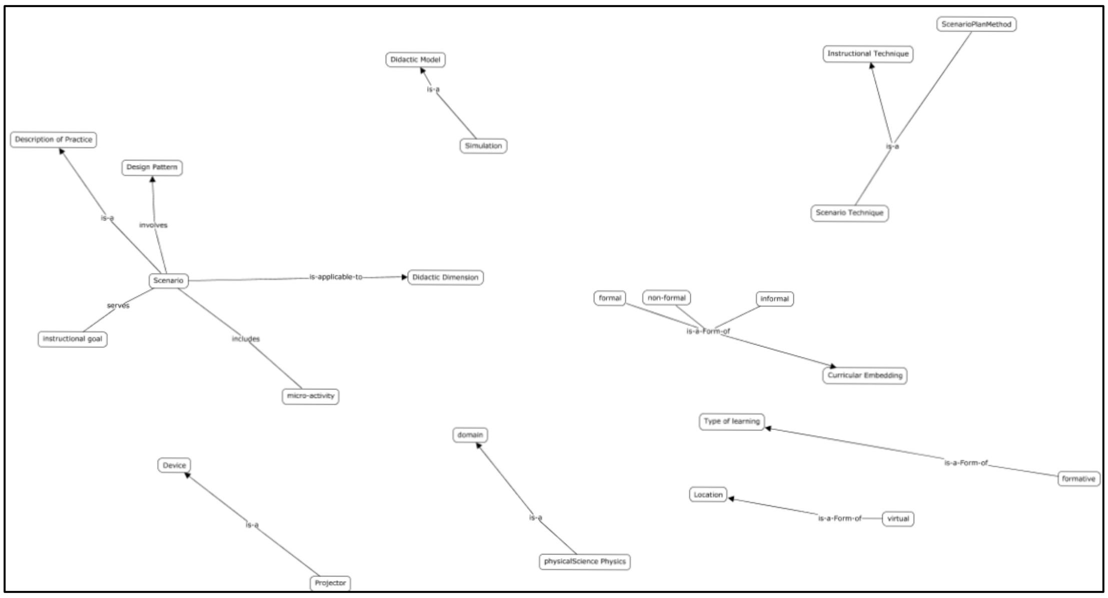


Figure 23. CMap of scenario1

As already said in the previous chapter, the main problem of this kind of visualisation is that it is a static and non-interactive visualisation. Furthermore, groups of nodes are not joined. Another negative aspect is that the positions of the nodes are fixed and they are the same of the ones they had in the learning ontology map. Otherwise, if positions had not been specified in this way, the results would have been maps with all nodes overlapping, like the example below:

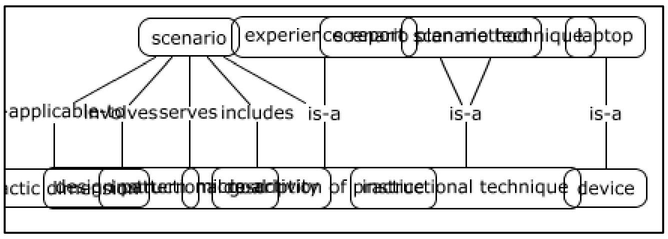


Figure 24. CMap scenario0 without specifying nodes positions

Because of the reasons explained above, it has been decided to switch to another kind of visualisation, displayed in HTML files supported by JavaScript and the d3.js library. Particularly, it has been created the full learning ontology map with the scenario covering concepts linked to the proper nodes (figure 25), then a map with all the domains in the ontology and the proper scenarios attached (figure 26) and finally a map with all the devices in the ontology and the proper scenarios attached (figure 27).

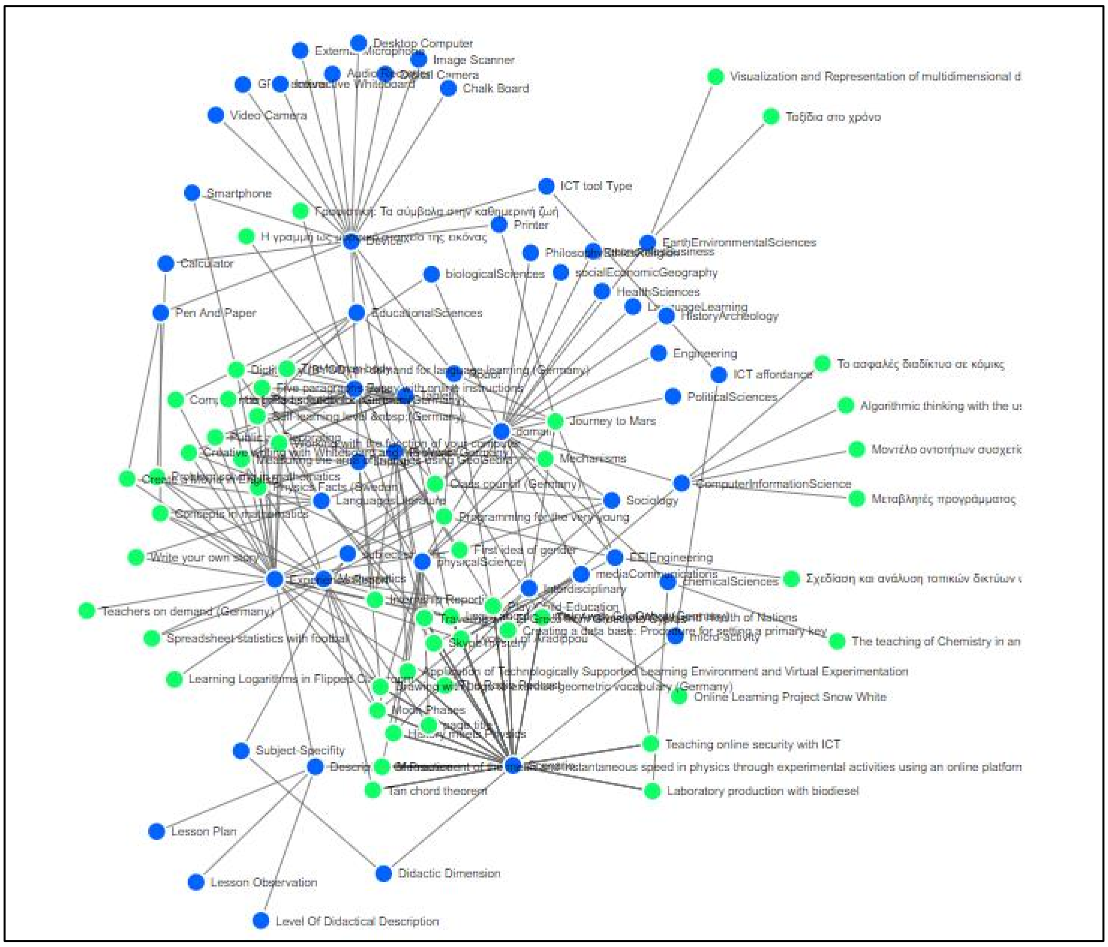
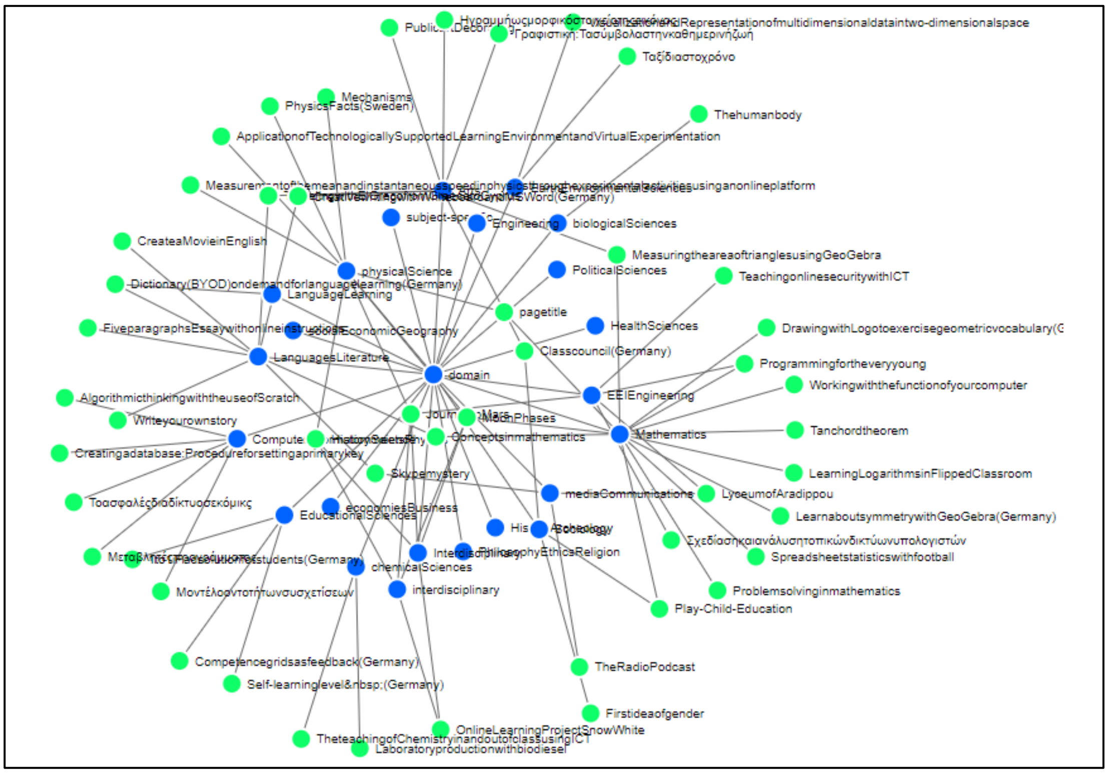


Figure 25. Complete map with d3.js library



VisualizationandRepresentationofmultidimensionaldataintwo-dimensionalspace
Γραφιστική:Τασύμβολαστηνκαθημερινήζωή
Ταξίδιαστοχρόνο
Mechanisms
PhysicsFacts(Sweden)
Thehumanbody
ApplicationofTechnologicallySupportedLearningEnvironmentandVirtualExperimentation
Measurementofthemeanandinstantaneousspeedinphysicsthroughexperimentalactivitiesusinganonlineplatform
subject-specifity
Engineering
biologicalSciences
CreateaMovieinEnglish
MeasuringtheareaoftrianglesusingGeoGebra
physicalScience
PoliticalSciences
Dictionary(BYOD)ondemandforlanguagelearning(Germany)
LanguageLearning
TeachingonlinesecuritywithICT
pagetitle
FiveparagraphsEssaywithonlineinstructions
socialEconomicGeography
HealthSciences
DrawingwithLogotoexercisegeometricvocabulary(G
Classcouncil(Germany)
LanguagesLiterature
Programmingfortheveryyoung
domain
Workingwiththefunctionofyourcomputer
AlgorithmicthinkingwiththeuseofScratch
EEIEngineering
Writeyourownstory
MoonPhases
Tanchordtheorem
Conceptsinmathematics
Mathematics
Creatingadatabase:Procedureforsettingaprimarykey
LearningLogarithmsinFlippedClassroom
Skypemystery
mediaCommunications
LyceumofAradippou
Τοασφαλέςδιαδίκτυοσεκόμικς
EducationalSciences
economiesBusiness
LearnaboutsymmetrywithGeoGebra(Germany)
Sociology
Σχεδίασηκαιανάλυσητοπικώνδικτύωνυπολογιστών
PhilosophyEthicsReligion
Spreadsheetstatisticswithfootball
chemicalSciences
interdisciplinary
Μοντέλοοντοτήτωνσυσχετίσεων
ProblemsolvinginMathematics
Play-Child-Education
Competencegridsasfeedback(Germany)
TheRadioPodcast
Self-learninglevel (Germany)
Firstideaofgender
OnlineLearningProjectSnowWhite
TheteachingofChemistryinandoutofclassusingICT
Laboratoryproductionwithbiodiesel

Figure 26. Map for domains with d3.js library

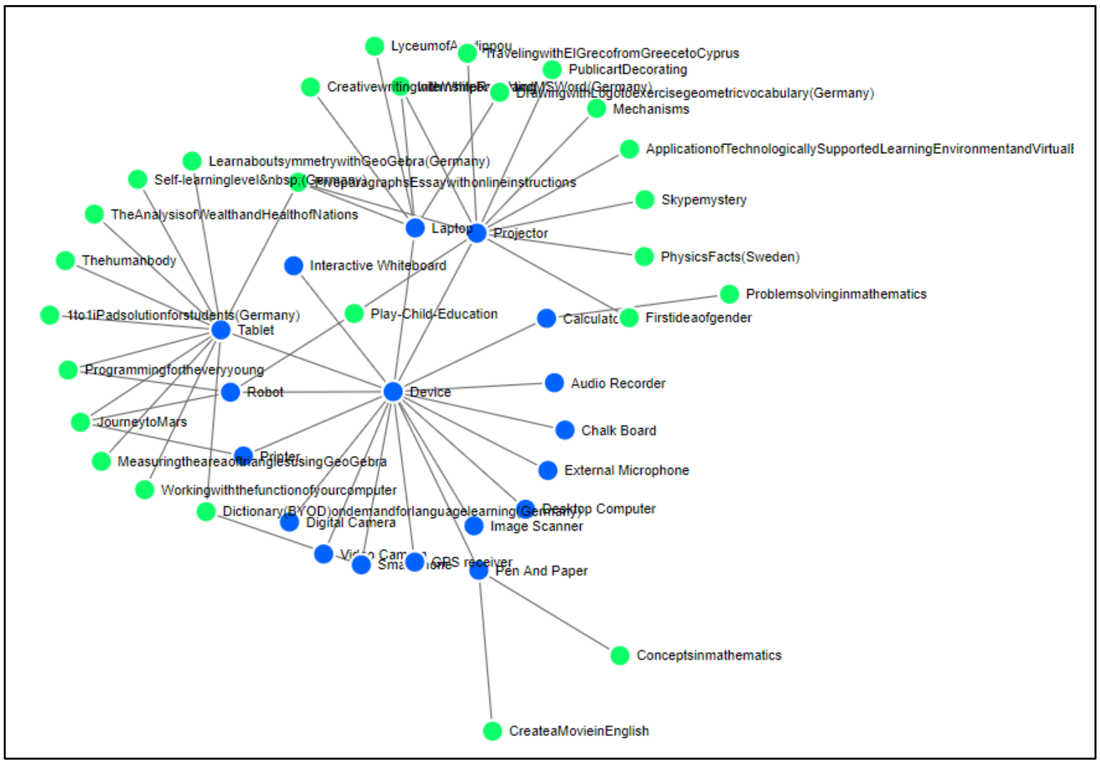


Figure 27. Map for devices with d3.js library

As can be noticed, also this kind of visualisation in not optimal: due to the large number of nodes, the final result is very disordered and confusing. Therefore, it has been created a map for each device, a map for each domain and a map for each scenario. The first of them have been built on the basis of a skeleton of the learning ontology map which keeps only the most important relations, thus avoiding the occurrence of separated groups of nodes. The following figures show as examples the map of the domain “Mathematics” (figure 28) and the map for the “scenario1” with the result of the interaction by clicking on the node representing the scenario (figure 29).

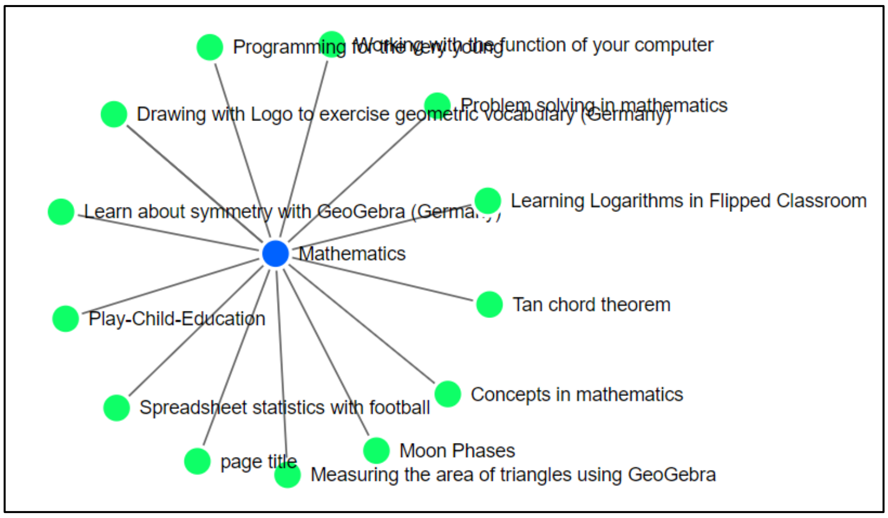
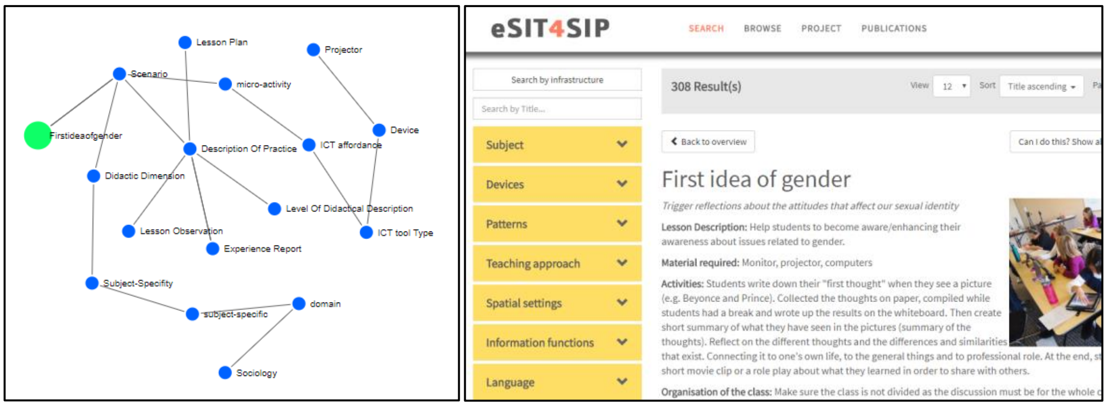

Mathematics
Programming for the very young
Working with the function of your computer
Drawing with Logo to exercise geometric vocabulary (Germany)
Problem solving in mathematics
Learn about symmetry with GeoGebra (Germany)
Learning Logarithms in Flipped Classroom
Play-Child-Education
Tan chord theorem
Spreadsheet statistics with football
Concepts in mathematics
page title
Moon Phases
Measuring the area of triangles using GeoGebra

Figure 28. Map of "Mathematics" domain



308 Result(s)

First idea of gender

Trigger reflections about the attitudes that affect our sexual identity

Lesson Description: Help students to become aware/enhancing their awareness about issues related to gender.

Material required: Monitor, projector, computers

Activities: Students write down their "first thought" when they see a picture (e.g. Beyonce and Prince). Collected the thoughts on paper, compiled while students had a break and wrote up the results on the whiteboard. Then create short summary of what they have seen in the pictures (summary of the thoughts). Reflect on the different thoughts and the differences and similarities that exist. Connecting it to one's own life, to the general things and to professional role. At the end, st short movie clip or a role play about what they learned in order to share with others.

Organisation of the class: Make sure the class is not divided as the discussion must be for the whole c

Figure 29. Map of scenario1 and result of clicking

This kind of visualisation and interaction is exactly what was requested. In order to provide a better user experience and a more efficient and appropriate navigation, all nodes which represent the title of learning scenarios are displayed as green circles, in contrast with the other types of node which are displayed as blue circles.

Nevertheless, some improvements should be suggested. Firstly, in this kind of visualisation, the label of the links does not exist, however it would be useful appending the label to the lines to clarify and distinguish the different types of relationships. One possible way to do it could be extracting the relations from the ontology while looking for matches and saving them into a new key of the hash created during the cycle when matches are found; then, these values could be appended to lines.

Another improvement of this visualisation might be differentiating the nodes giving them more different colours according to the concepts they represent (for examples, all domains with a certain colour, all devices with another colour). This improvement would be helpful in the map understanding.

Furthermore, it could be useful to move from one map to another one by clicking on one node. For example, moving to the web page of the domain “Mathematics” map by clicking on the node “Mathematics” in the complete ontology map, or also moving from the map of the “Tablet” device to the map of one scenario linked to it by clicking on the node which corresponds to the title of the scenario.

Another last but difficult improvement might be the creation of maps for each scenario without the support of the skeleton map, as it has been performed in this work. It means that for each scenario, whenever some matches with the learning ontology are detected and the corresponding nodes are created, these nodes must be linked automatically with a program that checks for the shortest path of the complete ontology which allows to join these nodes.

After having pointed out these considerations, it can be claim that the work has achieved the pre-set objective and the maps created for the scenarios existing on the XWiki space will be included on the web site eSIT4SIP.

The table below summarise the results:

Objective	Results	Improvements
Scraping XWiki environment to download the learning scenarios	57 learning scenarios downloaded with attachments too	• Translation of scenarios which are written in German and Greek • Downloading all the 295 scenarios which are on the eSIT4SIP web site
Comparison between each scenario and the learning ontology	Most of matches concern type of domain or devices required	• Adding a research for synonyms (e. g. through WordNet) • Extension of the research to paraphrases to detect concepts
Graphical Visualisation	Interactive maps for each domain, for each device and for each scenario	• Adding labels to line to define relationships • Adding links among the different maps • Better differentiate the type of nodes through the use of colours

Table 2. Results overview

5. Conclusions

This paper has shown the work conducted as an improvement of the existing project eSIT4SIP. This project aimed at improving the learning process providing teachers with learning scenarios on a variety of topics and subjects. The learning scenarios already existed on the web page of the eSIT4SIP project, thus the aim of this work was to create a graphical and interactive representation of the information contained in the scenarios and a graphical and interactive representation of the classification of the different learning scenarios by domains and by required devices to realise them.

The work was carried on step-by-step and for each step some limits, both intrinsic and not, have been detected and, if possible, a potential solution has been suggested.

The first step was the scraping of the scenarios from the platform they were stored in, a XWiki environment. In this stage, the following limits have been pointed out: not all the scenarios that are on the eSIT4SIP web site are also on the XWiki environment (only fifty-seven in two hundred-ninety-five), so it was not possible to work on the scenarios missing in XWiki.

Some of the scenarios were not written in English but in German and Greek, so the following matches detection could not bring results; the *eSIT4SIP Knowledge Base* project is not completed yet. First of all, in spite of the great variety of topics offered, that covers scientific and humanistic studies, the amount of learning scenarios in each subject is still too small. The scenarios are downloaded from two platforms (*LehrerOnline* and *MnSTEP*) or handwritten by professors who took part in the project's realisation. It would be desirable to find more platforms providing scenarios and also a functionality to allow qualified people to upload scenarios that will be firstly automatically vetted and then published.

Another limit, closely linked to the previous one, is the sequence of the process: the scenarios are firstly downloaded from the two platforms mentioned before, then they are uploaded on *XWiki,* later some manually authoring is made, finally they are scraped and written into a *.json* file, ready to be shown on the *eSIT4SIP* web page. It would be better to remove the most manual steps and to automate most of the work. It would be also useful to download automatically learning scenarios from the web.

The second step was the comparison between each scenario and the learning ontology map. In this stage the limits concern the research of matches, that up to now is a string matching. It would be useful to improve the research including the research of synonyms, for example with the WordNet dictionary, as well as including the research of meaning of phrases which put some concepts into words. Despite everything, the research brought satisfying and relevant results.

The third step, the building of concept maps, has shown the insufficiency of the CMap tool to display an interactive concept map and to dynamically compute and assign the position to the nodes according to their features and their amount. On the other side, thanks to the use of the d3.js library, it has been possible to create concept maps displayed with HTML. The positions of the nodes are dynamically computed in a way that the overlapping of two or more of them cannot happen. It has also been possible to add to the nodes which represent the scenarios a link to the scenario itself on the eSIT4SIP web site; thus, when the user clicks on the node, he is immediately redirected to the web page of the specific scenario.

It must be added that even this visualisation has some limits or better possibilities for being improved. Specifically, two important missing features which can help and guide the navigation of the map are: more differentiation of the nodes colouring (for example giving the same colour to the nodes belonging to the same concepts) and the addition of labels to the lines to specify the kind of relationship between two nodes (in fact, one important feature of the ontologies, which distinguishes them from typical concept maps, is the kind of relationships between two concepts). The relations are semantic-relationships (for example taxonomical relationships, composition relationships, description relationships, definition relationships and so on).

Another possibility to improve the work concerns the way through maps for the scenarios are built. In this project the maps are created starting from a skeleton of ontology which derived from the learning ontology. A more sophisticated way to build the maps could be the automatic detection of the relationships to link two nodes along the shortest path.

Overall, despite the limits explained above and despite the possibility of future improvements, the work can be considered successfully concluded.

References

[1] «eSIT4SIP,» [Online]. Available: http://www.esit4sip.eu/.

[2] «Enciclopaedia Britannica,» [Online]. Available: https://www.britannica.com/technology/data-mining.

[3] «Educational Data Mining,» [Online]. Available: http://educationaldatamining.org.

[4] G. Chiosso, Studiare Pedagogia. Introduzione ai significati dell'educazione, Firenze: Mondadori, 2018.

[5] G. Siemens, «Connectivism: A Learning Theory for the Digital Age,» *International Journal of Instructional Technology & Distance Learning,* vol. 1, pp. 1-8, 2014.

[6] F. Bell, «Connectivism: Its place in Theory-informed research and innovation in technology-enabled learning,» *International Review of Research in Open and Distance Learning,* vol. 12, n. 3, pp. 98-118, 2011.

[7] T. J. Ertmer e P. A. Newby, «Behaviorism, cognitivism, and constructivism: Connecting yesterday's theories to today's contexts,» *Perform. Improv. Q.,* vol. 26, n. 2, pp. 43-71, 2013.

[8] M. D. Borah e R. Jindal, «A Survey on Educational Data Mining and Research Trends,» *International Journal of Database Management Systems,* vol. 5, n. 3, 2013.

[9] C. Romero, S. Ventura e P. D. Bra, «Knowledge discovery with genetic programming for providing feedback to courseware author. User Modeling and User Adapted Interaction,» *The Journal of personalization Research,* vol. 14, n. 5, pp. 425-464, 2004.

[10] M. Grobelnik, D. Mladenic e M. Jermol, «Exploiting text mining in publishing and education,» in *Proceedings of the ICML workshop on data mining lesson learned*, Sydney, Australia, 2002, pp. 34-39.

[11] M. Bienkowski, M. Feng e B. Means, Enhancing Teaching and Learning Through Educational Data Mining and Learning Analytics: An Issue Brief, Office of Educational Technology, US Department of Education, 2012.

[12] R. Arruabarrena, T. Perez, J. Lopez-Cuadradoo e J. Vadillo, «On evaluating adaptive system for education,» in *Adaptive Hypermedia*, 2002, pp. 363-367.

[13] V. Kumar, «An Empirical Study of the Applications of Data Mining Techniques in Higher Education,» *International Journal of Advanced Computer Science and Applications,* vol. 2, n. 3, pp. 80-84, 2011.

[14] J. Varghese, M. Bindiy, J. Tomy, A. Unnikrishnan e Poulose, «Clustering Student Data to Characterize Performance Pattern,» *International Journal of Computer Science and Applications,* pp. 138-140, 2010.

[15] P. Gulati and A. Sharma, «Educational data mining for improving educational,» *International Journal of Computer Science, Information Technology, and Security,* vol. 2, n. 3, pp. 648-650, 2012.

[16] B. Baradwaj e S. Pal, «Mining Educational Data to Analyze Students' Performance,» *International Journal of Advanced Computer Science and Applications,* vol. 2, n. 6, 2011.

[17] A. B. E. D. Elaraby e I. S. Ahmed, «Data Mining: A prediction for Student's Performance Using Classification Method,» *Journal of Computer Application and Technology,* vol. 2, n. 2, pp. 43-47, 2014.

[18] U. Pandey e S. Pal, «Data Mining: A prediction of performer or underperformer using classification,» *International Journal of Computer Science and Information Technologies,* vol. 2, n. 2, pp. 686-690, 2011.

[19] B. Bhardwaj e S. Pal, «Data Mining: A prediction for performance improvement using classification,» *International Journal of Computer Science and Information Security,* vol. 9, n. 4, 2011.

[20] V. Ivancevic, M. Celikovic e I. Lukovic, «The individual stability student spatial deployment and its implications,» in *International Symposium on Computers in Education*, 2012.

[21] A. Wolff, Z. Zdrahal, D. Herrmannova, J. Kuzilek e M. Hlosta, «Developing predictive models for early detection of at-risk students on distance learning modules,» in *Machine Learning and Learning Analytics Workshop at The 4th International Conference on Learning Analytics and Knowledge*, Indianapolis, Indiana, USA, 2014.

[22] D. Allemang e J. Hendler, Semantic Web for the Working Ontologist, Waltham, Massachusetts, USA: ELSEVIER, 2011.

[23] M. Ruíz-Primo, « On the use of concept maps as an assessment tool in science: What we have learned so far,» *Revista Electrónica de Investigación,* vol. 2, n. 1, pp. 29-52, 2000.

[24] A. Okada, S. Buckingham Shum e T. Sherborne, Knowledge Cartography: Software tools and mapping techniques, London: Springer, 2008.

[25] J. Novak, «How do we learn our lesson?: Taking students through the Process,» *The Journal of Research in Science Teaching,* vol. 60, pp. pp. 50-55, 1993.

[26] J. J. Villalon e R. A. Calvo, «Concept Map Mining: A Definition and a Framework for Its Evaluation,» *Proceedings of the International Conference on Web Intelligence and Intelligent Agent Technology,* vol. 3, pp. 357-360, 2008.

[27] P. Suraweera, A. Mitrovic e B. Martin, «A Knowledge Acquisition System for Constraint-based Intelligent Tutoring Systems,» in *Proceedings of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*, IOS Press, Amsterdam, 2005, pp. 638-645.

[28] «The Protégé Ontology Editor,» [Online]. Available: http://protege.stanford.edu/. [Consultato il giorno 29 01 2019].

[29] K. Zebrinic, D. Kalpic e M. Milicevic, «The automatic creation of concept maps from documents written using morphologically rich languages,» *Expert Systems with Applications,* vol. 39, n. 16, pp. 12709-12718, 2012.

[30] R. B. Clariana e R. Koul, «A computer-based approach for translating text into comcept map-like rapresentation,» in *Proceedings of the First International Conference on Concept Mapping*, Pamplona, Spain, 2004.

[31] B. R. Gaines e M. L. G. Shaw, «Using Knowledge Acquisition and Representation Tools to Support Scientific Communities,» in *Proceedings of the 12th National Conference on Artificial Intelligence*, Calgary, Alberta, Canada, 1994.

[32] J. H. Kowata, D. Cury e M. C. S. Boeres, «Concept Maps Core Elements Candidates Recognition from Text,» in *Proceedings in the Fourth International Workshop on Managing Requirements Knowledge*, Viña del Mar, Chile, 2010.

[33] N. S. Chen, Kinshukb, C. Wei e H. Chen, «Mining e-Learning domain concept map from academic articles,» *Computers & Education,* vol. 50, pp. 1009-1021, 2008.

[34] J. W. Cooper, «Visualization or relational text information for biomedical knowledge discovery,» in *Information Visualization Interfaces for Retrieval and Analysis workshop*, 2003.

[35] M. Hagiwara, «Self-organizing concept maps,» in *Proceedings of the IEEE International Conference on System Man and Cybernetics Intelligent System for the 21st Century*, 1995.

[36] A. M. Olney, W. L. Cade e C. Williams, «Generating Concept Map Exercises from Textbooks,» in *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, Stroudsburg, PA, USA, 2011.

[37] A. Zouaq e R. Nkambou, «Building Domain Ontologies from Text for Educational Purposes,» *IEEE Transactions on Learning Technologies,* vol. 1, n. 1, pp. 49-62, 2008.

[38] Waugh e Beasely, «Cognitive mapping architectures and hypermedia disorientation: an Empirical study,» *Journal of Educational Multimedia and Hypermedia,* n. 4, pp. 239-255, 1995.

[39] M. Porter, «An algorithm for suffix stripping,» *Program,* vol. 14, n. 3, pp. 130-137, 1980.

[40] A. J. Berlanga, M. Kalz, S. Stoyanov, P. van Rosmalen, A. Smithies e I. Braidman, «Language Technologies to Support Formative Feedback,» *Edicational Technology & Society,* vol. 14, n. 4, pp. 11-20, 2011.

[41] C.-J. Huang, P.-H. Tsai, C.-L. Hsu e R.-C. Pan, «Exploring Cognitive Difference in instructional outcomes using Text mining technology,» in *2006 IEEE International Conference on Systems, Man and Cybernetics*, Taipei, Taiwan, 2006.

[42] «UIMA,» 13th April 2007. [Online]. Available: http://uima-framework.sourceforge.net/. [Consultato il giorno 29 01 2019].

[43] D. Klein e C. Manning, «Accurate unlexicalized parsing,» in *Proc. of the 41st Meeting of the Association for Computational Linguistics*, Sapporo, Japan, 2003.

[44] R. Y. K. Lau, A. Y. K. Chung, D. Song e Q. Huang, «Towards fuzzy domain ontology based concept map generation for E-Learning,» *ICVL: Proceedings of the 6th international conference on Advances in web based learning,* pp. 90-101, 2007.

[45] Studer, Benjamins e Fensel, «Knowledge Engineering: Principles and Methods,» *Data and Knowledge Engineering,* vol. 25, pp. pp. 161- 197., 1998.

[46] «Hamline University,» [Online]. Available: https://www.hamline.edu/education/environmental/cgee/mnstep.html.

[47] «Ruby. A programmer's best friend,» [Online]. Available: https://www.ruby-lang.org/en.

[48] «WebStorm,» [Online]. Available: https://www.jetbrains.com/webstorm/.

[49] «XWiki,» [Online]. Available: https://www.xwiki.com/en.

[50] «Cmap,» [Online]. Available: https://cmap.ihmc.us/cmaptools/.

[51] [Online]. Available: https://github.com/MeVis/esit4sip.

[52] «Apache Jena,» [Online]. Available: https://jena.apache.org.

[53] «Introducing Json,» [Online]. Available: https://www.json.org.

[54] «w3.org,» [Online]. Available: https://www.w3.org/TR/rdf-sparql-query .

[55] «Online JSON Viewer,» [Online]. Available: http://jsonviewer.stack.hu/.

[56] P. University, «WordNet,» 2010. [Online]. Available: https://wordnet.princeton.edu/.

[57] A. Zouaq e R. Nkambou, «Building Domain Ontologies from Text for Educational Purposes,» *IEEE Transactions on Learning Technologies,* vol. 1, pp. 49-62, 2008.

[58] W. W. W. Consortium, «W3C Semantic Web Activity,» [Online]. Available: https://www.w3.org/2001/sw/. [Consultato il giorno 29 01 2019].

[59] G. Ryle, The Concept of Mind, University of Chicago Press, 1949.

[60] A. A. Saa, «Educational Data Mining & Students' Performance Prediction,» *International Journal of Advanced Computer Science and Applications,* vol. 7, n. 5, pp. 212-220, 2016.

[61] G. Berruto e M. Cerruti, La linguistica. Un corso introduttivo, Torino: UTET, 2011.