



Virtual Portal for Interaction and ICT Training for People with Disabilities

Integrated ViPi Platform – ViPi Semantic Content Management (VSCM) Module: Requirements Analysis, Design and Technology Selection

Outcome No.		13 (Annex I)	
Workpackage No.	4	Workpackage Title	Deployment of ViPi interactive community portal and e-learning platform
Authors		George Milis, Demetrios Eliades, Demetris Stavrou (G.M EuroCy Innovations Ltd)	
Status (F: final; D: draft; RD: revised draft):		F	
File Name:		ViPi-D13-Integrated ViPi Platform-v2.0-Annex I-VSCM Module.docx	

The ViPi KA3 LLL project (511792-LLP-1-2010-1-GR-KA3-KA3NW) has been partially funded under the Lifelong Learning program, subprogramme KA3 ICT. This publication reflects the views only of the author(s), and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Version History table

Version no.	Date	Dates and comments
0.1	22/03/2011	Initial draft presenting the proposed components of the semantic module and discussing their implementation
0.9	25/05/2011	Submitted to the consortium for comments/feedback
0.92	21/10/2011	Feedback accommodated
1.0		Final version
1.1	Med Feb 2012	Provided comments and feedback
1.2	23/02/2012	Addressed comments and incorporated all required changes and clarifications
2.0	End of May 2012	Revision and finalization

Table of Contents

Version History table	2
Glossary	5
1 Introduction.....	6
1.1 The ViPi platform vision.....	6
1.2 The need for the Semantic Content Management module	7
1.3 Outline	9
2 General design of the VSCM module	10
2.1 The target users-groups	10
2.2 Architecture.....	10
2.3 Access Control and Access Rights Management	11
2.4 The back-end Administration component	12
2.5 The Semantic Repository component	14
2.6 The Content Annotation component	15
2.6.1 Annotation of content	15
2.6.2 Annotation of end-users' profile	15
2.6.3 Tools	16
2.6.4 Additional/Optional features:	16
2.7 The Semantic Navigation component	19
2.8 The Semantic Conceptual Search component	20
3 Use case scenarios.....	21
3.1 Installation and Configuration of VSCM plugin	21
3.2 The Administration component	22
3.3 The Semantic Repository component	26
3.4 The Content Annotation component	29
3.5 The Semantic Navigation component	32
3.6 The Semantic Conceptual Search component	34
4 Selection of tools.....	36
4.1 Recommended technologies and tools	36
4.1.1 View layer	36
4.1.2 Middleware layer – Business logic	36
4.1.3 Storage layer.....	37
4.1.4 Create ontology offline.....	37



4.1.5	Alternative tools for the ontology management:	37
4.1.6	Methodology for creating the ontology	38
4.1.7	Reasoning	38
4.1.8	Semantic navigation and annotation	39
4.2	VSCM Ontology.....	39
4.2.1	References for the VSCM Ontology:.....	40
5	Discussion.....	41



Glossary

OWL: Ontology Web Language. An xml-like knowledge representation language, used for authoring and storing of ontologies (http://en.wikipedia.org/wiki/Web_Ontology_Language)

RDF(S): Family of World Wide Web Consortium (W3C) specifications, to be used as a general method for conceptual description or modelling of information that is implemented in web resources, using a variety of syntax formats (http://en.wikipedia.org/wiki/Resource_Description_Framework)

1 Introduction

1.1 The ViPi platform vision

According to the 2002 EU Labour Force Survey (LFS) and the 2004 EU Statistics on Incomes and Living Conditions, an estimated 45 million people in Europe, being 16% of men and women aged 16-64, have a long-standing health problem or disability (LSHPD). A big proportion of these people are considerably restricted in the kind or amount of work they could do or their mobility to and from work, resulting to very low percentages of those actually being in employment.

Today the maturity level of the Information and Communication Technologies (ICT) provides alternative and creative solutions for the employment of people with several kinds of limitations and disabilities. This is evident also in recent studies conducted by various projects such as ACCESSIBLE (www.accessible-project.eu) and AEGIS (www.aegis-project.eu), which have highlighted that persons with disabilities can benefit enormously from digital competences which are core life and employability skills (thus contributing to the objectives of the Lisbon Objectives).

With respect to the above situation, the primary objective of the ViPi project is to fulfil that gap, by making available accessible and flexible training, designed to be adopted directly by people with disabilities, as well as through centres providing special education and vocational training. More specifically, the ViPi project aims at developing an interactive online platform where people with disabilities will be able to access a wide variety of ICT training courses and mobile and Internet/PC-based educational games for acquiring ICT skills, while on the other hand, trainers will be able to upload and download specific learning objects (LOs). Additionally, the ViPi platform will offer a mobile Android based social application, a customized and localized curriculum on ICT skills and training and a trainer handbook. Thus, the platform will comprise a virtual collaborative learning environment for people with disabilities and their trainers to interact.

It is obvious from the above objectives, that the ViPi platform adopts efficiently the concepts and technologies of Web 2.0, that is, it moves beyond the traditional one-way interaction of the Internet, providing collaborative and social networking facilities. The importance of this fact is twofold: i) The two-way interaction capabilities of the Web 2.0 are employed also for the benefit of people with disabilities, increasing their quality of life and communication means, ii) The social networking technologies are employed in the training/education domain, showing the path to efficient technology adoption in domains other than the pure leisure communication.

Moving one step further, the ViPi project will also explore the adoption of the Web 3.0 concepts. That is, the environment of the ViPi platform will be semantically enriched, thus improve the searchability and relevance of the results obtained. The presentation and the design of the modules that will undertake the semantic content management in the ViPi platform, is the objective of this document. The following sections and chapters provide further insight in the need for such a module within the ViPi platform and detail the design considerations that will allow the subsequent implementation of the module.

1.2 The need for the Semantic Content Management module

Before dealing specifically with the semantic module of the ViPi platform, we provide a short explanation of what Web 3.0 and Web 3.0 technologies are. These concepts and technologies deal with the semantics of the exchanged knowledge and are growing and becoming increasingly successful and visible in the Web world, following the "Web 2.0" revolution with the growth of social networking, blogging, wikis, etc. Examples of successful applications, range from open and shared databases, to semantic search, to tools for collective intelligence of groups and teams, to control of personal information on the Web, to online television and many more. The shared keyword of these advancements is the "semantic". The Semantic Web is the concept aiming at combining the Web data resources, standard languages, tools and ontologies into new applications that exploit the power of semantic exchange of content. Figure 1 shows how Web 2.0 is combined with the semantic Web to leverage the accepted definition of the Web 3.0 generation, that is, Semantic Web technologies integrated into, or powering, large-scale Web applications.

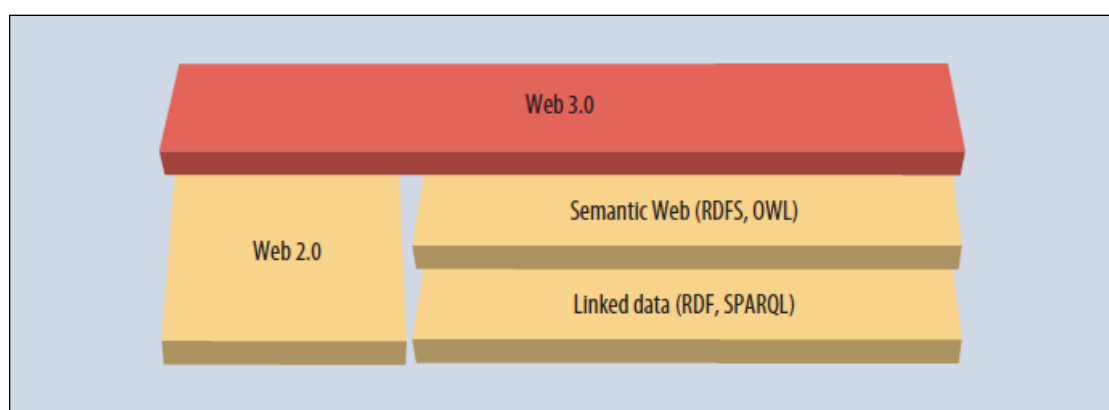


Figure 1: Relation of Web 3.0 to Web 2.0 and Semantic Web

The base of Web 3.0 applications is the Resource Description Framework (RDF) that provides the means to share and link the data from multiple sources in a structured format. Once the data is in RDF form, the use of uniform resource identifiers (URIs) facilitates development of multisite mashups of the data. For more on the use of RDF in Web-application development, see J. Hendler and O. Lassila, "Embracing Web 3.0," IEEE Internet Computing, May/June 2007, pp. 90-93.

In addition to the structure, the data are also semantically described in the RDF Schema (RDFS), using ontologies (adopting the Web Ontology Language (OWL) or the Web Service Modelling Ontology (WSMO) frameworks). The ontologies allow for the assertion of relationships between data elements, thus enabling the exchangeability of data among Internet applications and users, providing the means of generating useful information and subsequently knowledge. Furthermore, the ontologies provide the ability to infer relationships between data in different sites; reasoning algorithms are employed to perform these tasks. A reasoning algorithm comprises a set of rules that are used to discover relationships among concepts in the ontology, which are not directly given in the ontology. For example, an ontology may define that a specific content of a web-page is suitable for a specific age group. At the same time, the ontology may define that an individual is classified under a specific age group. Whether the content is suitable for the individual is not directly encoded

in the ontology but can be easily inferred combining the already given relations. Building reasoning rules, one can extract “complex” knowledge using the ontology.

In a perspective, Web 3.0 can be seen as integrating content in a meaningful world. At the same time, the success of the ViPi platform is highly related with the quantity and the quality of the provided content. As introduced in the previous section, the ViPi platform will be enhanced with semantic annotation and discovery of content, thus moving beyond the basic keyword search functionality and adopting at the extent possible the value of the Web 3.0. The types of content to be managed within the ViPi platform and the related specific objectives of the project are presented in the following paragraphs.

A first type of content will be the static content, to be provided through the ViPi portal, providing and disseminating the necessary information to the target users. This type of content is usually of limited quantity and its presentation is undertaken by the main navigation areas of each portal, which makes it easy for the users to locate it, decode its meaning and obtain the exact information they need. This type of content can be further described with semantic metadata (descriptive information stored together with the content itself, to help mainly machines, i.e. other websites, to exchange part of it). Though this is part of the concept of Web 3.0, it will not be the main focus of the ViPi platform.

A second type of content will be the content to be created through the collaborative interaction of the users of the social facilities. People with disabilities, trainers and policy makers will exchange emails, chat, discuss in forums, add comments, rate the quality of the provided material etc. Semantic languages (ontologies) to describe the meaning of such type of content can be also defined to allow for more advance exchange and localization. However, since this content will not be restricted in one single domain and research results revealed that it is highly ambitious to try to pre-define ontological barriers to such communication, the ViPi project decided not to deal with the semantic annotation of this second type of content.

A third type of content will be the learning objects to be created, stored and managed within the ViPi platform. This type of content can be placed somewhere in the middle, that is, it is not controlled in the sense that it increases in quantity through time, and at the same time, it is of controlled types and within controlled domain (ICT skills education of people with disabilities to acquire or sustain employment positions). Furthermore, it is very important for the ViPi platform users to be able to locate such content easily and effectively. A task that looks easy for people without any limitations might be very annoying for people with some kinds of limitations and disabilities. To leverage the effective storing, search and thus usage of the learning objects, the ViPi project decided to embed semantic annotation and navigation capabilities for the learning objects.

Finally, a fourth type of content will be the profiles of the users accessing the ViPi platform. In order to enable automatic matching of the preferences of users to the learning objects, the ViPi project will embed semantic annotation on the user profiles, as well.

The semantically enhanced content management within the ViPi platform will be undertaken by a dedicated module, to be called the VSCM.

1.3 Outline

In fulfilling the set objectives, the document is organised as follows:

Chapter 1 introduces the vision of the ViPi project and more specifically presents the driving need for the VSCM. Chapter 2 then analyses the components that will comprise the module and discusses the main functional and technical specifications. Chapter 3 follows with more details on the implementation guidelines of the components of the semantic module, providing also a set of indicative use cases to support the understanding and subsequent analysis by the developers. Chapter 4 continues by presenting and proposing individual third-party tools to be used for the implementation of the components of the VSCM, as well as, providing the ontology to be incorporated within the ViPi platform. Finally, Chapter 5 concludes the document and highlights the important aspects to be considered by the developing team.

2 General design of the VSCM module

2.1 The target users-groups

The target user-roles of the VSCM are the following (see Figure 2).

- i. Administrator: The users with the “Administrator” role will be dedicated to managing the back-end part of the module in order to retain it operational and at expected performance levels. For example, these users will be responsible for the uploading and management of ontologies and reasoning (see previous section) tools.
- ii. Content provider: The user with the “Content Provider” role will have rights to create and upload training content (learning objects) within the ViPi platform. In turn, these users will be responsible for the semantic annotation of the content using the dedicated functionality of the VSCM.
- iii. End-Users (general public, including people with disabilities, Content Providers, intermediaries, and policy makers): Users with the “End-user” role will have the right to access, rate and make use of the content provided within the ViPi platform. More specifically, the users with this role will be able to use the semantic search capabilities of the ViPi platform to locate the exact content they are looking for, easily and efficiently.

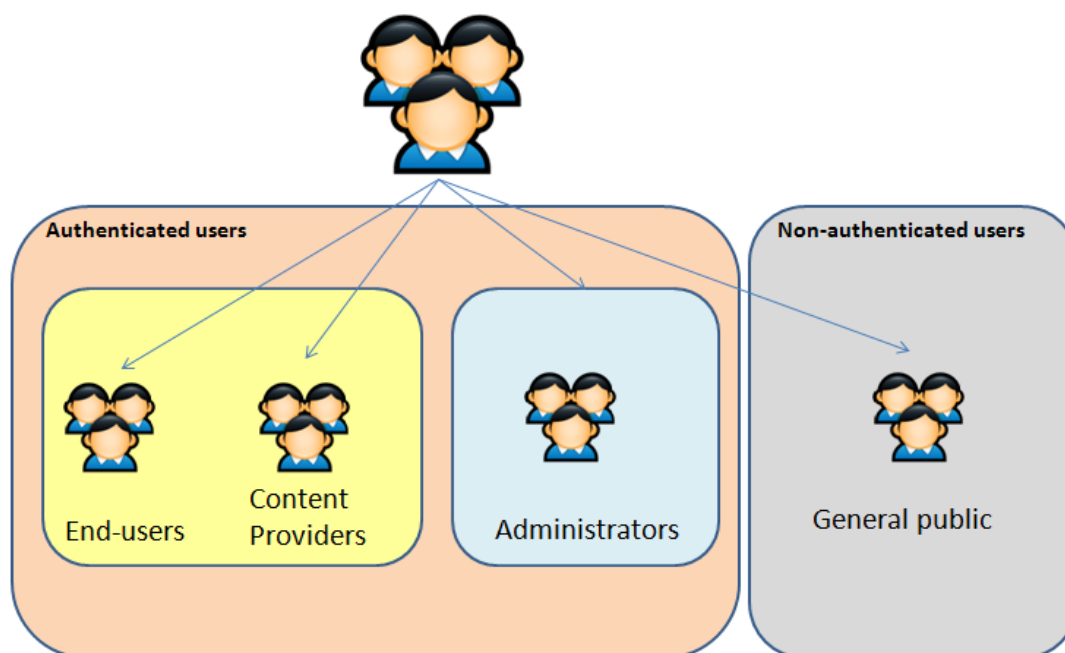


Figure 2: Target users of the VSCM module

2.2 Architecture

This section provides a comprehensive architectural overview of the VSCM modules, using a high level architectural view. It captures and conveys the significant architectural decisions which are to be made on the system.

The Design Model of the VSCM module adopts a 3-layer architecture, depicted in Figure 3:

- **View (presentation) layer:** This layer comprises the view options provided by the module to the end users. The view layer will be implemented in PHP (Web view) and native Android (mobile view) technologies. The mobile view will be specific to the type of mobile devices used (e.g. tablet or smartphone) and will be contained in the respective mobile applications.
- **Middleware (Business) layer:** The business logic, as well as, the interface implementation with the other two layers will be developed in PHP/MySQL and SOAP (web services) technologies. Mobile devices will interact with the main system, for gathering information regarding the implementation logic and pure data, through Web services. All functionalities that will be provided through the mobile applications, will need to be exposed as Web services with clear SOAP interfaces.
- **Storage (database) layer:** The Database Server will be the main storage repository. The MySQL database management system will be utilized. Where necessary, mobile applications will use local databases on mobile devices to store temporary data or mobile-specific data.

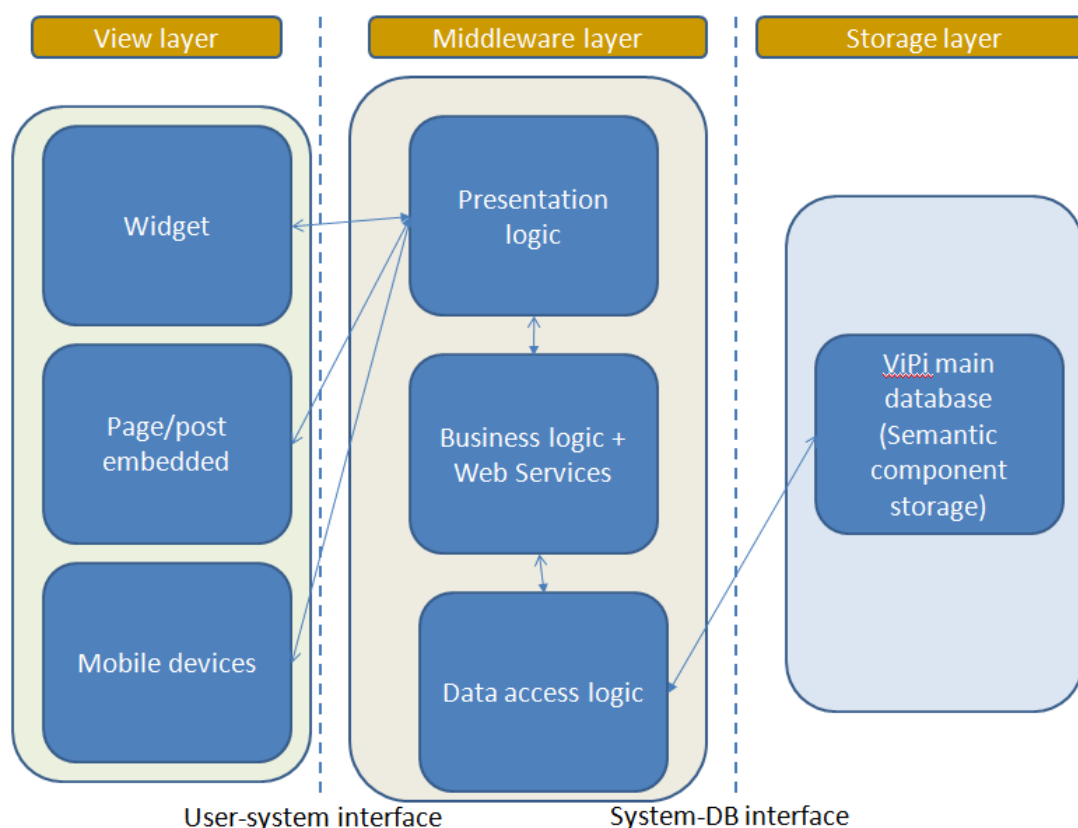


Figure 3: High-level architecture of the VSCM module

2.3 Access Control and Access Rights Management

The implementation of the ViPi core platform will adopt the WordPress platform (<http://WordPress.org>) as the back-end software. Therefore, the Access Control mechanism of WordPress will be also utilised to cover the needs of the VSCM module. There is no need for separate access control functionality to be provided specifically for the needs of this module.

More specifically, the implementation of the module will consider incorporating the access control of WordPress for the several resources of the VSCM module (e.g. ontology, back-end administration

component, annotation component, search component). Therefore, the roles that will be created for the ViPi platform will be assigned the proper rights so as to access and use the components of the VSCM module.

2.4 The back-end Administration component

Unless otherwise decided by the consortium before or during the implementation phase, the VSCM module will be implemented as a plugin of the WordPress platform. This not only benefits the module by allowing portability and integration with several side services and functionalities, but also serves as input to the WordPress community and increases the exploitability.

A starting point where one should look when interested to implement a plugin for WordPress is the relevant page of the WordPress organisation itself, at <http://WordPress.org/extend/plugins/about/>.

Figure 4 presents a snapshot of the administration panel of WordPress. For example, one may see the “My Calendar” section indicated by the red rectangle. This is an individual sub-panel that provides the administrator with a set of functionalities. Like all other sub-panels, it can be collapsed and expanded depending on whether the administrator wants the set of functionalities hidden or visible at a specific moment. It is noted that this sub-panel is not included in the WordPress core functionality, but was added later by installing the “My Calendar” plugin from the “Plugins” sub-panel.

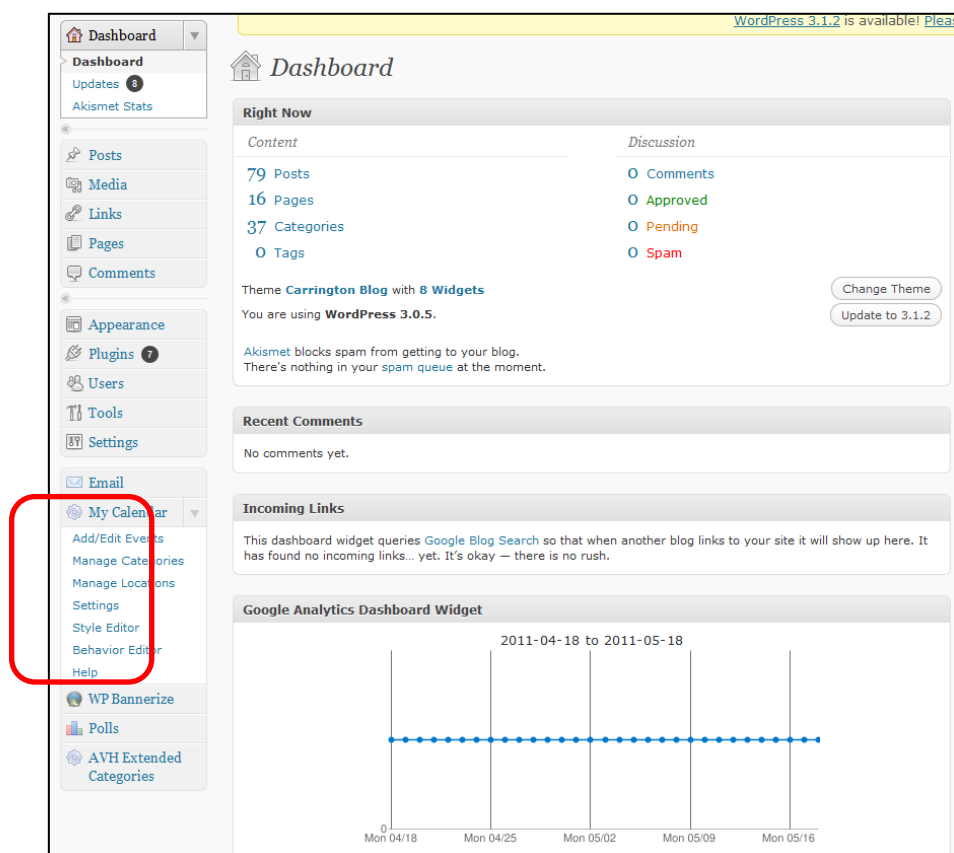


Figure 4: Snapshot of WordPress general administration panel

The Administration component of the VSCM module will also provide such a sub-panel, which will appear in the main ViPi platform administration panel upon the installation of the VSCM plugin.

In order to provide the necessary functionality to the VSCM module's administrator, the plugin is suggested to offer the following set of administration functionalities (each functionality will open a dedicated page, as shown in Figure 5, which will allow performing the available tasks):

- **General settings:** The respective page will offer functionality to configure the plugin within the ViPi (or other) platform. E.g. to configure where and how to store the ontology, to select default ontology, to decide on several annotation capabilities, etc. In order to maintain a user-friendly access to the plugin, it is recommended that the settings are grouped based on what they help to achieve and provided in tabs within the main page.
- **Upload/Update/Delete Ontology (e.g. RDFS, OWL format):** The respective page will offer simple functionality to either upload a zip file from the local file-system of the user, or give the URI pointing to an ontology file. The format of the ontology (e.g. RDFS, OWL) will be pre-defined and will be decided during the development phase in order to avoid the inconvenience of potential wrong automatic detection by the platform. The ontology will be stored as indicated in the repository component, later in this document. For convenience, the page will offer the option to upload new ontology, replace an existing one, or delete an existing one. That is, the administrators will be able to maintain several ontologies within the ViPi platform, always defining one of them as the default (currently used). Of course, the latter will have to be performed with high caution and by persons with sufficient skills, since there might be confusion generated for content previously annotated using an ontology and searching using another. To avoid any loss of information with evolution of the ontology, after changing the default ontology or performing structural changes, the administrator will have to provide a mapping between old and new ontology concepts. The mapping will be mandatory and facilitated by a dedicated tool. The mapping tool can be implemented as part of the VSCM plugin or even provided by third party (external) solution. In the latter case, the VSCM plugin must provide an interface through which the Administrator will upload the mapping and run the process to update the stored annotations accordingly.

Add Event

Add an Event

Enter your Event Information

Event Title(required)

Event Description (HTML allowed)

Event Short Description (HTML allowed)

Event Host admin

Event Category General

Event Link (Optional)
☐ This link will expire when the event passes.

Event Date and Time

Enter the beginning and ending information for the first occurrence of this event.

Start Date (YYYY-MM-DD) (required) 2011-05-18 Time (hh:mm) 15:49

End Date (YYYY-MM-DD) End Time (hh:mm)

Current time difference from GMT is 2 hour(s)

Recurring Events

Repeats for 0 Units Does not recur

Enter "0" if the event should recur indefinitely. Your entry is the number of events after the first occurrence of the event: a recurrence of 2 means the event will happen three times.

Event Registration Status

My Calendar does not manage event registrations. Use this for information only.

☒ Open ☐ Closed ☐ Does not apply

☐ If this event recurs, it can only be registered for as a complete series.

Event Location

Figure 5: An example of a page of an administration functionality

2.5 The Semantic Repository component

The ontology, the semantic annotation information of the content, as well as any other data collected by the VSCM plugin, will need to be stored in a dedicated media. In order to promote portability of the solution, it is recommended that separate relational database tables are created during the installation of the plugin, to undertake the storing of all VSCM data. The use of any local file-system for storing must be avoided.

The following tables are at least envisaged:

- **Ontology Table:** This table will host the ontologies in an XML-like format. In case versioning of ontologies is supported, the information will be stored in the same table with additional metadata. The ontology will include the hierarchy of classes, as well as, any properties of the objects and relations among them. The individual entities will not be stored as part of the ontology. In other words, the knowledge base will be separated and comprised of the ontology and the semantic annotations that follow.
- **Semantic Annotations table:** Each content type in WordPress is stored with a specific "ID". The annotations will be performed on entities uploaded within the platform (mainly of type "Post" in WordPress) and will be associated with them. Therefore, the annotations can comprise an XML-like text that will be stored in the SemanticAnnotations table, next to the ID of the specific content type. The structure of the XML-like file will be decided during the

development time and based on the feedback to be received up to that moment. The semantic annotation of the learning objects can be implemented as an extension of the default tagging feature of WordPress. The new set of tags that can be associated with content types, can be called “vstaggs” and can adopt a similar storage mechanism as the default tags.

Ontology mappings table: The table will store one row for each mapping performed between ontologies. The ID of the mapping, or the IDs of the two mapped ontologies can serve as the primary key, and then another field can host the mappings in a convenient XML-like format.

2.6 The Content Annotation component

There are two types of content that need to be annotated within the ViPi platform: i) the users and ii) the learning objects. This semantic annotation can be considered as implementing a semantic profile of users and learning objects so as to facilitate intelligent matching when searching.

2.6.1 Annotation of content

Every type of content to be created as a learning object within the ViPi platform, will need to be annotated upon creation. Therefore, the annotation functionality is recommended to be mandatory in order to avoid inconsistency of stored learning objects.

The majority of the content types can be uploaded in WordPress using the main type of “Page” or “Post”. That is, images, videos, links to external sources, documents, etc can be conveniently associated with a post. However, there are also other types of content that can be maintained separately, and these are, for example, the Media (direct storage of files of any type) and Links. As shown in Figure 6 - Figure 8, each of these content types is created through dedicated pages. In addition to the content itself, other information can be provided at the time of creation, through separate sub-boxes as indicated by the red highlighting boxes on the figures.

The VSCM plugin will thus create a similar separate section under each content type entity (e.g. post, page, media, link, etc), through which the annotation of the content will be performed. Updates of the annotations will be also possible at subsequent visits of the content’s page.

From within this section (after expanding), the user with a “Content provider” role, will be able to navigate through the ontology and annotate the content entity with specific ontology classes. It is recommended that the annotation mechanism is implemented in a way similar to the classification of posts into categories, in WordPress. That is, the user will be able to browse through a hierarchical view of the classes and select the ones with which to tag the content. If any properties need to be filled in, these can appear when selecting a specific class. As mentioned also in previous section, these semantic annotations can be visible as special types of tags and associated with the content type. This way, the user will be able to delete and add such tags as deemed necessary.

2.6.2 Annotation of end-users’ profile

In addition to the content (Learning Objects) the annotation of the profile of end-users/individuals is also considered useful in VSCM plugin, in order to allow association of users to specific content types

and skills and facilitate the relevant reasoning while searching. The annotation will be performed within the user creation page in the Administration panel.

This feature can be considered as implementing a semantic profiling of the users, to facilitate the matching with the learning objects' semantic profiling. It is recommended that the functionality be available for both the administrator and the user herself. The benefits of the semantic profiling of users will also comprise a motivation for the users to register within the ViPi platform rather than using it as public users.

2.6.3 Tools

As an alternative to the “manual” implementation of the ontology browsing functionality, the development team may consider using third-party tools that provide browsing capabilities within ontology graphs. Examples of such tools are given in Chapter 4. Furthermore, it will be decided at implementation time whether the third-party tool or the implemented functionality will be embedded in a WordPress page or a new window will open to allow for more space.

It is also important that the navigation through the ontology is made accessible. That is, the navigation panel must be compliant to the WCAG 2.0¹, at the extent possible. The semantic annotation component will be also made usable and accessible through mobile devices. In this case, the communication between the mobile device and the core ViPi platform will adopt the SOAP protocol. The necessary services will be implemented as Web services allowing external devices to call them and obtain access to the content.


Finally, the annotation information will be stored as described in section 2.4.

2.6.4 Additional/Optional features:

In order to provide scalability features, the users that perform the annotation of the content can be provided with the functionality to suggest new classes and attributes/properties, as extensions to the ontology. This can be implemented as a simple contact form with no structure or a semi-structured form guiding the user on how to submit this suggestion. It is noted, that the suggestions will be always checked by the administrator and incorporated in the master ontology, if deemed appropriate.

One more optional feature is the automatic annotation of media of textual type (e.g. posts, files in doc, odt, ppt, pdf format, etc). In case that is desirable, although not recommended by the authors of this document, the VSCM module needs to integrate calls to external service that will generate an “annotation-vector” based on the currently active ontology. The implementation of such annotation capabilities requires considerable text- and language-processing effort, turning it to highly unreliable. Basic free language processing features may be required for the conceptual search component described in section 2.7.

¹ <http://www.w3.org/TR/WCAG/>

 Add New Post

Enter title here

Upload/Insert

Visual HTML

Paragraph

Path: p

Word count: 0

Excerpt

Excerpts are optional hand-crafted summaries of your content that can be used in your theme. [Learn more about manual excerpts.](#)

Send Trackbacks

Custom Fields

Discussion

Author

Event Editor

Start

End

All Day

All in One SEO Pack

Figure 6: Example of content post with additional features associated with it




Upload New Media

Choose files to upload [Select Files](#) [Cancel Upload](#)

Maximum upload file size: 10MB

You are using the Flash uploader. Problems? Try the [Browser uploader](#) instead.

After a file has been uploaded, you can add titles and descriptions.



File name: ViPi_Platform-Semantic_Content_Management_module.pdf

File type: application/pdf

Upload date: 18/05/2011

Title


Caption

Description

File URL
Location of the uploaded file. [Delete](#)

[Save all changes](#)

Figure 7: Example of content post (media) with additional features associated with it



Add New Link

Name

Example: Nifty blogging software

Web Address

Example: <http://wordpress.org/> — don't forget the <http://>

Description

This will be shown when someone hovers over the link in the blogroll, or optionally below the link.

Categories

☒ All Categories
 ☐ Useful links

[Most Used](#)

[+ Add New Category](#)

Figure 8: Example of content post (link) with additional features associated with it

2.7 The Semantic Navigation component

The Semantic Navigation component of the VSCM module will be implemented as an extension to advance search functionality. Since the public must be able to use the functionality offered by this component, it must be made available through the front-end of the ViPi platform.

The majority of WordPress plugins offer two ways of integrating views of components. It is recommended that these two ways are adopted also for the VSCM plugin, as follows:

- **Widget:** the VSCM widget(s) will appear in the Widgets' catalogue of the ViPi (WordPress-based) platform, as shown in Figure 9. The administrator will be able to drag & drop the widget in a sidebar available through the ViPi theme, and provide the necessary settings that will define the presentation of the functionality to the end-user.
- **Quick-code for inclusion within a post or page:** The semantic navigation functionality will be also provided with an option to incorporate within a Web page or single post. To ease the work of the administrator, the plugin will provide a Readme file, providing details on how to do this integration. The embedded-code option, will requires that the administrator writes within a post/page the codified information for the user interface to be loaded properly. E.g. `[semantic_component option1=value1, option2=value2, ...]`

From the user point of view, the semantic navigation component will be either accessible through the left/right side of the ViPi platform pages or from a dedicated page. No matter where the access will be from, the user will be able to launch the navigation component. It is recommended that the navigation tool opens in a new page, to increase visibility. The user will be able to navigate through a graphical (e.g. hierarchical tree-like) representation of the ontology and select one or multiple classes/nodes for which to present results. There are several accessibility issues that need to be taken into account in the implementation of the VSCM plugin. For example, upon opening the navigation page, the screenreader user must be informed, while upon completing the form, it must be made clear to the screenreader user that the flow will return to the previous page after saving.

It is noted that the core of the semantic navigation component will be identical to the semantic annotation component. However, the components will differ in the output they provide. That is, the semantic annotation component will not provide any output but rather a confirmation of successful storing of the annotations, whereas the semantic navigation component will present the discovered results as output on a page. It is recommended that the results are presented as links to their exact location in the ViPi platform, possibly with short description. The objective is to allow for efficient searching with limited amount of results obtained, however, generic queries will still return many results, thus leading to the need of having a paging mechanism. In addition to the paging, an optional filtering functionality can be also envisaged.

Similarly to the semantic annotation component, this component must be made usable and accessible through mobile devices, as well.

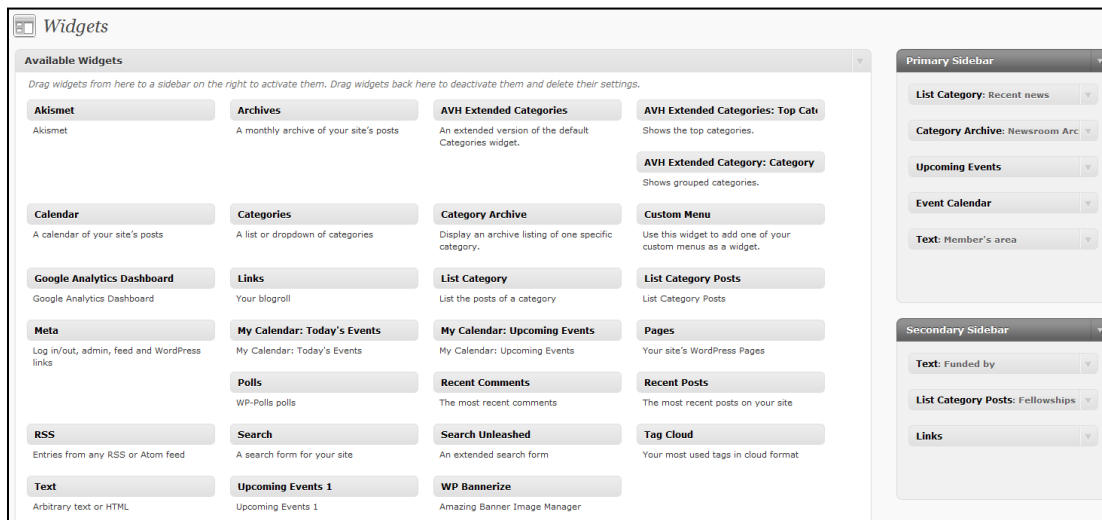


Figure 9: The Widgets' catalogue of WordPress platform

2.8 The Semantic Conceptual Search component

This component is presented as an **optional feature** to further add value to the user's experience when searching for information.

Unlike the navigation component that requires implementation of new views, this component comprises a normal search-like feature and its implementation can be totally based on an existing search plugin of WordPress.

The difference from a classical search (i.e., enter a phrase in a box, press a button and receive results based on that phrase) will be that when typing in the text-box, the user will be presented with suggested classes and reasoning outcomes from the ontology to accelerate and tailor the search experience. One can imagine the experience like the one when using Google search engine, where potential search options are presented in real-time.

If implemented, the component will be loaded as a widget for direct inclusion by the administrator. In addition, the component can be optionally made available through mobile devices, depending on the available budget and time of the developing team. Otherwise, it is a feature that can be added by other community members that would like to contribute to the effort and this is a great value given by the adoption of an Open Source platform as the back-end tool.

3 Use case scenarios

3.1 Installation and Configuration of VSCM plugin

[INS-01] Find and install plugin	
Version:	1.0
Actors:	Administrator
Preconditions:	The actor has sufficient access rights within the Administration console of the ViPi platform. The plugin has been accepted to be uploaded in the WordPress.org plugin database.
Main Flow:	<ol style="list-style-type: none"> 1. Login as Administrator in the ViPi Administration console 2. Go to Plugins sub-menu 3. Click on "Add new" option 4. In the page that opens, choose to search based on "Term" 5. In the text box, write the name of the plugin (or an associated keyword term) and click enter 6. Locate the VSCM plugin in the list of results and click on "Install now" 7. Click "ok" to confirm installation 8. Click on "Activate plugin" 9. The plugin will appear in the list of installed plugins
Alternative Flows:	Not applicable
Notes:	The use case assumes that the VSCM module is implemented as a plugin of WordPress platform
Issues:	None

[INS-02] Plugin configuration and usage instructions	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights within the Administration console of the ViPi platform - The VSCM plugin has been installed successfully in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Locate the VSCM plugin in the list of installed plugins 2. Click on "Edit" link 3. In the page that opens, the content of the plugin appears with a list of code-pages on the right. 4. Click on ".../Readme.txt" link for the relevant content to appear in the main window 5. Read carefully the instructions about the features and usage scenarios of the plugin, before actually proceeding in configuration
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

3.2 The Administration component

[A-01] Plugin configuration	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights within the Administration console of the ViPi platform - The VSCM plugin has been installed successfully in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Locate the VSCM plugin sub-menu in the left column of the administration panel 2. Click on the "General settings" link 3. In the page that opens: <ol style="list-style-type: none"> a. Define where to store the ontology (e.g. in a table in the WordPress MySQL database or as a single file) b. Upload the default ontology c. Activate/inactivate annotation and search capabilities d. etc. 4. Click on "Save changes"
Alternative Flows:	Not applicable
Notes:	It is recommended that a default ontology is distributed together with the plugin to ease the job of the Administrator. In that case, the Administrator may be asked to confirm the use of that ontology and the storage place, or select another one during settings' configuration.
Issues:	None

[A-02] Upload new ontology	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights within the Administration panel of the ViPi platform - The VSCM plugin has been installed successfully in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Locate the VSCM plugin sub-menu in the left column of the administration panel 2. Click on the "Manage ontology" link 3. Click on the "Upload ontology" tab 4. In the page that opens: <ol style="list-style-type: none"> a. Give the URI of the ontology to be uploaded b. Browse in local file system to locate the zip file containing the ontology c. Click on "Upload" d. Confirm uploading e. (Next step asks the user whether to set the uploaded ontology as the default one. The use case A-08 takes over)
Alternative Flows:	Not applicable
Notes:	None
Issues:	The uploading of ontologies must be handled with high caution. In

	parallel to the basic actions, the plugin needs to take actions also to maintain consistency of the annotations performed with replaced ontology, etc. To avoid loss of information, the administrator will have to provide clear mapping between the concepts of the currently used ontology and the ones of the newly uploaded.
--	---

[A-03] Edit existing ontology	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights within the Administration panel of the ViPi platform - The VSCM plugin has been installed successfully in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Locate the VSCM plugin sub-menu in the left column of the administration panel 2. Click on the "Manage ontology" link 3. Click on the "Edit ontology" tab 4. In the page that opens: <ol style="list-style-type: none"> a. Select one of the existing ontologies b. Click on "Upload new ontology" c. Select either "Replace existing one" or "Store new version" d. Click on "Upload" 5. Confirm uploading
Alternative Flows:	Not applicable
Notes:	None
Issues:	The editing of ontologies must be handled with high caution. In parallel to the basic actions, the plugin needs to take actions also to maintain consistency of the annotations performed with changes in ontology, etc. To avoid loss of information, the administrator will have to provide clear mapping between the concepts of the currently used ontology and the ones of the edited version. The mapping can be performed inside the VSCM plugin or outside, by external tool. In the latter case, the plugin must provide a way to the Administrator to upload the mapping information to be used for updating the stored annotations.

[A-04] Delete ontology	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights within the Administration panel of the ViPi platform - The VSCM plugin has been installed successfully in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Locate the VSCM plugin sub-menu in the left column of the administration panel 2. Click on the "Manage ontology" link 3. Click on the "Delete ontology" tab 4. In the page that opens: <ol style="list-style-type: none"> a. Select an ontology to delete

	<ul style="list-style-type: none">b. Click on "Delete"c. Click on "Confirm deletion"
Alternative Flows:	Not applicable
Notes:	The deletion of ontologies will only be allowed if there is no content annotated with that ontology. Otherwise, the annotations will need to be deleted first, utilizing the relevant use case. When an active ontology is deleted, the user must be prompt to select a different ontology, or to upload a new one and also provide the necessary mappings, as described in previous use cases.
Issues:	None

[A-05] Configure reasoning engine

Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights within the Administration panel of the ViPi platform- The VSCM plugin has been installed successfully in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. Locate the VSCM plugin sub-menu in the left column of the administration panel2. Click on the "Configure reasoning engine" link3. In the page that opens:<ul style="list-style-type: none">a. Edit the necessary information to define the connection details with the reasoning engineb. Click on "Save" button.
Alternative Flows:	Not applicable
Notes:	In case the reasoning engine is implemented using a third party tool, there might be a need for installing and configuring that tool on the same server, as deemed necessary. The task will be performed by the Administrator, offline. It is recommended that a default reasoning engine is distributed together with the VSCM plugin. This will be installed and activated transparently, upon installation/activation of the plugin by the Administrator.
Issues:	None

[A-06] Manage suggestions for ontology changes

Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights within the Administration console of the ViPi platform- The VSCM plugin has been installed successfully in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. Check new posts under category "Suggestions for ontology"2. Go to the posts and read their content using the default functionality of the WordPress platform.3. Take necessary decisions and implement them offline4. If applicable, upload new version of ontology
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[A-07] Mapping of ontology concepts	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights within the Administration console of the ViPi platform- The VSCM plugin has been installed successfully in the ViPi platform- There are at least two ontologies uploaded in the platform
Main Flow:	<ol style="list-style-type: none">1. Locate the VSCM plugin sub-menu in the left column of the administration panel2. Click on the "Manage ontology" link3. Click on the "Map concepts" tab4. Select two of the stored ontologies and click on the "Provide mappings" button5. A page opens presenting the classes of the first ontology and drop boxes next to each of them for the user to select classes of the second ontology6. As soon as all classes are mapped, click "Save"
Alternative Flows:	5b. Instead of the online functionality, the user might also be able to upload the mappings in a pre-defined XML-like format.--
Notes:	The mapping must be always performed by an expert person in managing the ontologies of the specific domain.
Issues:	None

[A-08] Set default ontology	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights within the Administration console of the ViPi platform- The VSCM plugin has been installed successfully in the ViPi platform- There are at least two ontologies uploaded in the platform
Main Flow:	<ol style="list-style-type: none">1. Locate the VSCM plugin sub-menu in the left column of the administration panel2. Click on the "Manage ontology" link3. Click on the "Set default ontology" tab4. Select an ontology and click on the "Set as default" button<ol style="list-style-type: none">a. The action will be completed successfully only if the mappings between the previous default ontology and the newly selected have been already provided. A clear warning will be presented to the Administrator, since wrong mappings may lead to inconsistent semantic metadata stored in the database that may negatively affect the search experience of users.5. If mappings ok, then the new ontology is set as "default".
Alternative Flows:	
Notes:	The action must be always performed by an expert person in managing the ontologies of the specific domain.
Issues:	None

[A-09] Update of annotations after mapping	
Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none"> - The VSCM plugin has been installed successfully in the ViPi platform - The administrator has completed a mapping between two ontologies - The Administrator has prompted to replace the default ontology
Main Flow:	<ol style="list-style-type: none"> 1. Locate the specific mapping entry from the relevant table 2. Go through all performed annotations and apply the changes using the mapping information 3. Store the new annotations in the relevant fields in the tables
Alternative Flows:	N/A
Notes:	None
Issues:	None

3.3 The Semantic Repository component

The Semantic Repository Component does not provide any interaction facilities with the end-users, and therefore the usage scenarios are not applicable. Instead, the following comprise system use-cases that aim to support the developers in implementing the proper flow of actions within the plugin.

[SR-01] Store ontology	
Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none"> - The VSCM plugin has been installed successfully in the ViPi platform - The Administrator user has submitted a request to store an ontology in the VSCM repository
Main Flow:	<ol style="list-style-type: none"> 1. Based on the input provided by the view layer, construct the XML-like content that contains the ontology <ol style="list-style-type: none"> a. Both in the case of a new ontology and in the case of a new version, the content is already available in the proper format. It is recommended that the format to be adopted is the RDF, although the use of OWL would also be a valid option. 2. Generate an ID for the new ontology or retrieve the ID of the existing ontology 3. Store the content of the ontology along with the necessary metadata for easy management of the versions of ontologies.
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SR-02] Delete ontology	
Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none"> - The VSCM plugin has been installed successfully in the ViPi platform - The Administrator user has submitted a request to delete an

	ontology from the VSCM repository
Main Flow:	<ol style="list-style-type: none">1. Based on the input provided by the view layer, locate the ID of the ontology to be deleted2. Check whether there are annotations stored using the specific ontology.<ol style="list-style-type: none">a. If yes, show a message of inability to delete unless the relevant annotations are deleted first.b. If no, show a message asking for confirmation and then remove the ontology from the repository.
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SR-03] Store annotations

Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none">- The VSCM plugin has been installed successfully in the ViPi platform- The Administrator user has submitted a request to store new content annotations in the VSCM repository
Main Flow:	<ol style="list-style-type: none">1. Based on the input provided by the view layer, construct an XML-like content for the new set of annotations2. Retrieve the ID of the specific content entity.3. Store the annotations, together with the ID of the content entity in the relevant table4. Show success operation message.
Alternative Flows:	Not applicable
Notes:	Instead of storing the annotations in a dedicated XML-like format, the default means of storing tagging information in WordPress can be exploited.
Issues:	None

[SR-04] Edit annotations

Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none">- The VSCM plugin has been installed successfully in the ViPi platform- The Administrator user has submitted a request to edit existing content annotations in the VSCM repository
Main Flow:	<ol style="list-style-type: none">1. Retrieve the ID of the annotations set stored for the specific content type2. Based on the input provided by the view layer, construct an XML-like content for the updated set of annotations3. Update the annotations in the table.4. Show success operation message.
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SR-05] Delete annotations	
Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none">- The VSCM plugin has been installed successfully in the ViPi platform- The Administrator user has submitted a request to delete existing content annotations in the VSCM repository
Main Flow:	<ol style="list-style-type: none">1. Retrieve the ID of the annotations set stored for the specific content type2. Show message asking for confirmation of the deletion3. Update the annotations table by deleting the relevant entry.4. Show success operation message.
Alternative Flows:	Not applicable
Notes:	None
Issues:	The content type (in case classified as learning object) will not allow the update option in case no annotations are provided. The semantic annotations are a mandatory input to the content type.

[SR-06] Store mappings	
Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none">- The VSCM plugin has been installed successfully in the ViPi platform- The Administrator user has submitted a request to store a new mapping of ontologies in the VSCM repository
Main Flow:	<ol style="list-style-type: none">5. Based on the input provided by the view layer, construct an XML-like content for the new set of mappings6. Retrieve the ID of the two mapped ontologies.7. Store the mapping information, together with the IDs of the ontologies in the relevant table8. Show success operation message.
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SR-07] Edit mappings	
Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none">- The VSCM plugin has been installed successfully in the ViPi platform- The Administrator user has submitted a request to edit existing mapping of ontologies in the VSCM repository
Main Flow:	<ol style="list-style-type: none">1. Retrieve the mapping information based on the IDs of the selected ontologies and present it to the user in editable fields.2. Based on the input provided by the view layer (after changes by the administrator), construct an XML-like content for the updated mapping3. Update the mappings in the table.4. Show success operation message.
Alternative Flows:	Not applicable
Notes:	None

Issues:	None
---------	------

[SR-08] Delete mappings	
Version:	1.0
Actors:	System
Preconditions:	<ul style="list-style-type: none"> - The VSCM plugin has been installed successfully in the ViPi platform - The Administrator user has submitted a request to delete existing mappings in the VSCM repository
Main Flow:	<ol style="list-style-type: none"> 1. Find the entry of mappings that corresponds to the Ids of the selected ontologies 2. Show message asking for confirmation of the deletion 3. Update the mappings table by deleting the relevant entry. 4. Show success operation message.
Alternative Flows:	N/A
Notes:	None
Issues:	None

3.4 The Content Annotation component

[CA-01] Annotate new content entity – Web	
Version:	1.0
Actors:	Administrator, Content provider
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as a Content Provider, within the Administration Panel of the ViPi platform - The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. From within the content entity creation page, locate the box dedicated to the semantic annotation. 2. Click on “Add new annotation” 3. Browse through the ontology and mark the classes with which to annotate the content. <ol style="list-style-type: none"> a. It is possible to add multiple annotations simultaneously. 4. Click on the general “Update” button of the content entity.
Alternative Flows:	Not applicable
Notes:	The ontology that opens for the annotation is always the one defined as default through the administration panel
Issues:	The functionality must be made compatible to WCAG 2.0

[CA-02] Edit existing annotations – Web	
Version:	1.0
Actors:	Administrator, Content provider
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as a Content Provider, within the Administration Panel of the ViPi platform - The VSCM plugin is active and operational in the ViPi platform

Main Flow:	<ol style="list-style-type: none">1. From within the content entity edit page, locate the box dedicated to the semantic annotation.2. Click on "Edit annotation"3. Select and delete one or multiple annotations4. Browse through the ontology and mark the classes with which to annotate the content.<ol style="list-style-type: none">a. It is possible to add multiple annotations simultaneously.5. Click on the general "Update" button of the content entity.
Alternative Flows:	Not applicable
Notes:	The ontology that opens for the annotation is always the one defined as default through the administration panel. Some of the annotations might be associated with previous versions of the ontology.
Issues:	The functionality must be made compatible to WCAG 2.0

[CA-03] Delete existing annotations – Web

Version:	1.0
Actors:	Administrator, Content provider
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights as a Content Provider, within the Administration Panel of the ViPi platform- The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. From within the content entity edit page, locate the box dedicated to the semantic annotation.2. Click on "Delete annotation"3. Select and delete one or multiple annotations4. Click on the general "Update" button of the content entity.
Alternative Flows:	Not applicable
Notes:	There is no option to update the content entity with no annotations available. Therefore, if all annotations are to be deleted, it is recommended to delete the content entity itself.
Issues:	The functionality must be made compatible to WCAG 2.0

[CA-04] Suggest changes in ontology – Web (optional)

Version:	1.0
Actors:	Content provider
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights as a Content Provider, within the Administration Panel of the ViPi platform- The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. From within the content entity create/edit page, locate the box dedicated to the semantic annotation.2. Click on "Submit suggestions"3. Fill-in the form with the suggestion details4. Click on "Submit" button of the content entity.
Alternative Flows:	The same functionality must be provided also through the Web services interface and implemented for the mobile devices.
Notes:	None.
Issues:	The functionality must be made compatible to WCAG 2.0

[CA-05] Automatic annotation of media of textual type – Web (optional)	
Version:	1.0
Actors:	Content provider
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as a Content Provider, within the Administration Console of the ViPi platform - The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. From within the Media content create/edit page, locate the box dedicated to the semantic annotation. 2. Click on “Automatic annotation” 3. (A call is made to a separate service, sending also the currently active ontology) 4. As soon as a relevant message appears informing on the completion of the operation, check the annotations appearing in the box. 5. Click on the general “Update” button of the Media entity.
Alternative Flows:	
Notes:	In case implemented, it is recommended to be available only through Web and not through mobile devices.
Issues:	The functionality must be made compatible to WCAG 2.0

[CA-06] Annotate new user – Web	
Version:	1.0
Actors:	Administrator, Content Provider, End-user
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform or as an end-user through the registration functionality - The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. From within the user registration page, locate the box dedicated to the semantic profile annotation. <ol style="list-style-type: none"> a. Similar box will be available for the public user in the registration page 2. Click on “Add new annotation” 3. Browse through the presented part of the ontology and mark the classes with which to annotate the user’s profile. 4. Click on the “Register” button for the new user to be created.
Alternative Flows:	Not applicable
Notes:	<p>The ontology that opens for the annotation is always the one defined as default through the administration panel.</p> <p>The semantic annotation of the profile of the user is recommended to be an optional feature.</p>
Issues:	None.

[CA-07] Edit/Delete annotations of user – Web	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform or she is a registered user of the platform. - The VSCM plugin is active and operational in the ViPi platform

Main Flow:	<ol style="list-style-type: none"> 1. From within the user profile page, locate the box dedicated to the semantic annotation of profile. 2. Click on "Edit/Delete annotation" 3. Delete any annotations that need to be deleted 4. Browse through the ontology and mark the new classes with which to annotate the user's profile. 5. Click on the "Update/Save" button of the user profile.
Alternative Flows:	Not applicable
Notes:	The ontology that opens for the annotation is always the one defined as default through the administration panel
Issues:	None.

3.5 The Semantic Navigation component

[SN-01] Configure Widget – Web	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform - The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Go to the "Appearance" sub-menu of the Administration panel. 2. Expand and click on "Widgets" link 3. Drag the "VSCM navigation" widget and drop it in the preferred Sidebar. 4. Fill-in the necessary information for the configuration of the widget. <ol style="list-style-type: none"> a. The minimum configuration information is expected to be: Title of widget (visible to the public), Navigate by (tree or graph or other methods, if more than one available), Save button, Close button. 5. Click on "Save".
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SN-02] Create page/post with semantic navigation component – Web	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform - The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Go to the "Pages" or "Posts" sub-menu of the Administration panel. 2. Click on "Add new" 3. In the main content area, write the codified information of the component, that is: <i>[semanticNavigation, option1=value1, option2=value2,...]</i>. 4. Provide all other necessary information for the creation of

	the post/page 5. Click on "Publish".
Alternative Flows:	Not applicable
Notes:	It is possible to include other content in the same post/page, above or below the embedded semantic navigation component.
Issues:	None

[SN-03] Remove widget – Web

Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform- The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. Go to the "Appearance" sub-menu of the Administration panel.2. Expand and click on "Widgets" link3. Drag the "VSCM navigation" widget from a sidebar and drop it either in the left-top part of the page (not retaining the configuration details) or in the left-bottom part of the page (to retain configuration details for later use).
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SN-04] Remove component from post/page – Web

Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform- The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. Go to the "Pages" or "Posts" sub-menu of the Administration panel.2. Click on "Posts"/"Pages" link3. Click on the title of the post/page that contains the semantic navigation component4. In the main content area, locate the codified information of the component and delete it.5. Click on "Publish" to publish the page with the rest of the content.
Alternative Flows:	5a. In the "Publish" box, click on "Status" and choose "Draft" for the page to go offline and be deleted later.
Notes:	None
Issues:	None

[SN-05] Navigate to find content – Web/Mobile

Version:	1.0
Actors:	Administrator, Content provider, Public
Preconditions:	<ul style="list-style-type: none">- The VSCM plugin is active and operational in the ViPi platform

Main Flow:	<ol style="list-style-type: none"> 1. From within the place where the semantic navigation component is provided, click to launch the navigation component. E.g. with a button "Launch navigation". 2. Browse through the ontology and use the provided functionality at each navigation level to request for results 3. Choose whether to consider the user's semantic profile or not. <ol style="list-style-type: none"> a. Choosing to consider the user's semantic profile, results in more intelligent and custom search within the available learning objects. The outcome is a list of results suitable to the user's situation and preferences. 4. Click on "Show results" 5. Browse through the results 6. Click on the title of a result-set entry to open it in its original location.
Alternative Flows:	<p>Access through mobile</p> <ol style="list-style-type: none"> 1.a Access the navigation functionality through the mobile device interface. 2.a and 6.a: Same as in main flow <p>Update results</p> <ol style="list-style-type: none"> 2.b Browse again through the ontology and use the provided functionality at each navigation level to request for results 3.b Click again on "Show results" 4.b Browse through the updated list of results
Notes:	None.
Issues:	The functionality must be made compatible to WCAG 2.0

3.6 The Semantic Conceptual Search component

[SCS-01] Configure Widget – Web	
Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none"> - The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform - The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none"> 1. Go to the "Appearance" sub-menu of the Administration panel. 2. Expand and click on "Widgets" link 3. Drag the "VSCM Conceptual Search" widget and drop it in the preferred Sidebar. 4. Fill-in the necessary information for the configuration of the widget. 5. Click on "Save".
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SCS-02] Remove widget – Web

Version:	1.0
Actors:	Administrator
Preconditions:	<ul style="list-style-type: none">- The actor has sufficient access rights as an Administrator, within the Administration Panel of the ViPi platform- The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. Go to the "Appearance" sub-menu of the Administration panel.2. Expand and click on "Widgets" link3. Drag the "VSCM Conceptual Search" widget from a sidebar and drop it either in the left-top part of the page (not retaining the configuration details) or in the left-bottom part of the page (to retain configuration details for later use).
Alternative Flows:	Not applicable
Notes:	None
Issues:	None

[SCS-03] Search to find content – Web

Version:	1.0
Actors:	Administrator, Content provider, Public
Preconditions:	<ul style="list-style-type: none">- The VSCM plugin is active and operational in the ViPi platform
Main Flow:	<ol style="list-style-type: none">1. From within the place where the semantic conceptual search component is provided, start entering text in the search box.2. (The system will be presenting in real-time a list of ontology classes relevant to the currently inserted search text)3. Use the arrows to choose one (or more) of the proposed ontology classes<ol style="list-style-type: none">a. Choose whether to consider user's profile in searching or not.4. Click on "Show results"5. Browse through the list of results obtained6. Click on the title of a result-set entry to open it in its original location.
Alternative Flows:	<p>Update results</p> <ol style="list-style-type: none">1.a Write new text in the search box3.a Choose different set of classes4.a Click on "Show results"5.a Browse through the updated list of results
Notes:	None.
Issues:	The functionality must be made compatible to WCAG 2.0

4 Selection of tools

For a broad array of technologies and tools available to engineers for designing applications compatible with the Semantic Web concept, the reader is referred to <http://www.w3.org/2001/sw/wiki/Tools>.

4.1 Recommended technologies and tools

4.1.1 View layer

The View layer of the VSCM module provides three different view options. The first one is the Widget, the second one is the codified version for posts/pages and the third one is the view for the mobile devices.

The relevant (recommended) selection of tools is presented in the following:

- Widget: Implemented following the widget implementation guidelines of WordPress community. The language to be used must be PHP
- Codified version: Implemented in PHP, adopting the good practices proposed by other plugin developers.
- For mobile device: Implementation based on the Android operating system, using an appropriate language and implementation guidelines. The call to the platform services will be through SOAP messages

4.1.2 Middleware layer – Business logic

The main part of the business logic of the VSCM module will be implemented in PHP scripting language, as it will be based on WordPress platform. The development guidelines provided by the WordPress community (http://codex.wordpress.org/Developer_Documentation) will be studied and adopted.

The parts that correspond to the browsing through the ontologies for semantic navigation or annotation, are discussed in next sections.

The interface to the mobile devices or any other client of the plugin services, is recommended to be implemented as Web services. We recommend the implementation of Web services for the functionality necessary to be exposed to mobile devices, as can be identified by the Chapter 3—Use Cases

The following link provides information on how to develop Web services with PHP:

<http://phpwebservices.blogspot.com/2008/08/make-your-wordpress-blog-web-service-in.html>

The relevant detailed guide is directly accessible at:

<http://www.dimuthu.org/blog/2008/08/11/make-your-wordpress-blog-a-web-service-in-few-step/>

In addition, the plugin “WordPress Web Service” available from <http://wordpress.org/extend/plugins/wordpress-web-service/screenshots/>, provides a ready mechanism to deploy Web services for a wordpress blog, by providing the access information to the relevant WSDL files.

Alternatively, the XML-RPC protocol can be used for exporting functionality to mobile devices

4.1.3 Storage layer

The storage layer of the VSCM will be implemented within the core database of the ViPi platform. New tables will be created as detailed in previous sections. The Database Management System to be used is the MySQL (<http://www.mysql.com/>), which is the one adopted by the WordPress community.

4.1.4 Create ontology offline

The tool that is proposed for the creation and offline management of the ontologies, is the **Protégé** (<http://protege.stanford.edu/>). This is the preferred tool by the majority of ontology engineers, as also revealed from a recent survey (M. Rahamatullah Khondoker, Paul Mueller – “Comparing Ontology Development Tools Based on an Online Survey”²).

A decision will be made at a later stage about the format of the ontology to be used. In any case, it is recommended to adopt either the RDF Schema (widely adopted by Semantic Web engineers) or the OWL (some advance features comparing to RDF) format.

4.1.5 Alternative tools for the ontology management:

An extended list of Ontology Editors can be seen at http://en.wikipedia.org/wiki/Ontology_editor.

The selection of the specific tool must be based on the price (open source Vs proprietary), as well as, on the application domain (some tools are more appropriate for specific domains). According to the survey mentioned above, the following are some frequently used tools (following Protégé):

- SWOOP
- Top Braid Composer
- Internet Business Logic
- Onto Track
- IHMC Cmap Ontology Editor
- OntoWiki: Collaborative generation of knowledge bases following a wiki-style approach

² http://dSPACE.icsy.de:12000/dSPACE/bitstream/123456789/272/4/WCE2010_pp188-192.pdf

4.1.6 Methodology for creating the ontology

There is no one single and correct way or methodology for developing ontologies. In order to keep things as simple as possible, we adopt the methodology described in section “3 A Simple Knowledge-Engineering Methodology” of the publication Natalya F. Noy and Deborah L. McGuinness, Stanford University, Stanford, CA, 94305, available online at:

http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html

According to this methodology, the steps to create an ontology are the following:

- Step 1. Determine the domain and scope of the ontology: In VSCM case, the domain is twofold: i) Learning Objects Content and ICT Skills, ii) Learning Objects Content and Disabilities
- Step 2. Consider reusing existing ontologies: In VSCM case, as described in the following section, we make use of parts of ASKIT ontologies for the disability groups and content characterisation.
- Step 3. Enumerate important terms in the ontology
- Step 4. Define the classes and the class hierarchy
- Step 5. Define the properties of classes
- Step 6. Define the properties of the data
- Step 7. Create individual instances

4.1.7 Reasoning

The reasoning on the ontology, that will facilitate the extraction of information from the ontology, for annotation and search, can be performed in either of the following two ways:

i) Implementation of a new reasoning tool. That is, the ViPi consortium can develop a simple and custom set of scripts with which to parse the ontology structure/content and retrieve the necessary information for the annotations and/or search.

ii) Integrate and use one of the established Description Logic Reasoners. A short list of such popular reasoners is given in:

<http://protege.stanford.edu/plugins/owl/api/ReasonerAPIExamples.html>

and

http://protegewiki.stanford.edu/wiki/Using_Reasoners

The advantage of using an existing reasoner engine is that the tool will be scalable and not require development effort when the ontology emerges in structure. A disadvantage is that the effort for the integration might be too much, depending on the selected tool. In general, there are reasoning tools developed in C++, Java, Python, etc.

4.1.8 Semantic navigation and annotation

The navigation through the ontology for annotating content or for performing semantic search queries, is recommended to be implemented in an as simple as possible way.

As mentioned earlier in the document, the way the WordPress platform manages the classification of content into Categories can be explored. The ontology can be presented in a hierarchical tree-like view, allowing the user to easily browse through the classes, expand and collapse the levels as deemed necessary and mark one or more classes.

In the cases of classes that pose properties, the marking of a class will pose a request to the user (e.g, by opening a small window or other more accessible alternative) to further clarify her search requirements.

The implementation of the above can be done with direct development in PHP (or some other scripting language) and dedicated calls to the Reasoning engines for the annotation and/or search capabilities.

Alternatively, the navigation through the ontology can be performed by integrating third-party graphical ontology browsing and annotation/tagging tools. Such tools will require integration within the VSCM plugin. Below we provide a non-exhaustive list of browsing and annotation tools:

- RDF Gravity (<http://semweb.salzburgresearch.at/apps/rdf-gravity/index.html>)
- OntoMat (<http://annotation.semanticweb.org/ontomat/index.html>): ontology browsing and annotation of text parts (creation of individuals in the knowledge base)
- Open Ontology Forge (<http://sourceforge.net/projects/ontoforge/>)

The decision on whether to integrate a third-party tool for the annotation of the learning objects will be made at a later stage. However, in that case, the integrated tool should be opened in separate window, in order to increase visibility and lower the integration effort. A very important aspect to consider is whether the third-party tool can be made accessible, in accordance with the WCAG 2.0.

At the moment of writing this document, we suggest that a very simple ontology browsing and annotation functionality should be developed, utilizing the WordPress features as much as possible.

4.2 VSCM Ontology

As mentioned in previous chapters, the VSCM ontology is being developed using the Protégé software tool. According to the specifications of the VSCM within the ViPi platform, the ontology should at least cover the following sub-domains:

- Content (Learning Objects)
- Users with preferences as properties
- Impairment Types: Associated to content as well as users
- Assistive Devices: Associated to content as well as users
- ICT Skills: Associated to content as well as users

The part of the VSCM ontology that concerns the content types, the users, their preferences and the devices, is implemented as a custom combination and adaptation of a sub-list of the ASK-IT project (<http://www.ask-it.org/>) ontologies (see references at the end).

The part of the VSCM ontology that concerns the ICT Skills, is implemented using relevant extractions from the EURES (<http://ec.europa.eu/eures/home.jsp?lang=en>) taxonomy as starting point but thoroughly customised to be in line with the ViPi Curriculum developed under WP3 of the ViPi project.

Since the VSCM ontology can be classified within the broader domain of ontologies for accessibility, it is recommended to adopt the COF (Common Ontological Framework) developed by the OASIS project (www.oasis-project.eu). This will facilitate the potential integration and reuse of the VSCM ontology with other ontologies developed under the same Framework.

4.2.1 References for the VSCM Ontology:

- [1] ASK-IT-general
- [2] ASK-IT-personal
- [3] ASK-IT-social
- [4] ASK-IT—TourismAndLeisure
- [5] ASK-IT-wbedu
- [6] OASIS EU project Common Ontological Framework (COF)
- [7] EURES Taxonomy (extracted ICT-related Skills)
- [8] The skills for life survey: a national needs and impact survey of literacy
(http://books.google.com/books?hl=en&lr=&id=uFW_Ruq_RbIC&oi=fnd&pg=PA9&dq=ict+skills&ots=EJzm3IDi7Y&sig=GrCM9p4pDaqPfw_m_qjCCyZqeY_k#v=onepage&q&f=false) - Source: US DOC (2002) based on Bureau of Labor Statistics (BLS), Occupational Employment Statistics (OES) (2002).
- [9]
http://www1.ets.org/Media/Tests/Information_and_Communication_Technology_Literacy/ictreport.pdf
- [10]http://www.tda.gov.uk/teacher/returning-to-teaching/routes-to-returning/~media/resources/teacher/return-to-teaching/rtt_ict_skills_audit.pdf
- [11] ViPi Curriculum

5 Discussion

The document comprises an initial high-level design effort for the Semantic Content Management (VSCM) module. The information contained in the document is the analysis of the requirements of the module so as to support the development process. As such, it is considered as an attachment of D13 – Integrated ViPi platform, which results from WP4 of the ViPi project.

The steps that need to be followed in the framework of D13 are as follows:

- Refine the requirements of the several components
- Further study the available tools and define potential usage scenarios and integration possibilities
- Continue and finalise the engineering of the ontology to be utilized by the VSCM module