Virtual Portal for Interaction and ICT Training for People with Disabilities

Testing Plan and Evaluation Methodology

Outcome No.		19	
Workpackage No.	6	Workpackage Title	Testing and Community building
Authors		George M. Milis (EuroCy), Demetrios Eliades (EuroCy), Karel Van Isacker (PhoenixKM), Panayiotis Tsoris (Steficon), Andrew Burton (NTU), Vilma Butkute (Hiteco), Kosmas Petrides (Hypertech)	
Status (F: final; D: draft; RD: revised draft):		F	
File Name:		WP6-D19-Testing-Plan-and-Evaluation-Methodology-v1.0.docx	

Version History table

Version no.	Date	Dates and comments
0.1	12-09-2012	Initial draft circulated by EuroCy, asking contributions by all involved partners
0.2	21-09-2012	Contributions and feedback received by PhKM and Hiteco
0.3	22-09-2012	EuroCy incorporated the received contributions and also prepared input in the document.
0.4	30-10-2012	All partners provided inputs
1.0	20-11-2012	Final version circulated by EuroCy

Table of Contents

1 Introduction

The deliverable "D17-Testing Plan and Evaluation Methodology" comprises the first outcome of WP6 of the ViPi project. It aims to undertake an extensive and iterative internal testing of the fully integrated ViPi platform, involving the whole consortium as well as external parties where possible, thus ensuring that any bugs or unwanted behaviour is dealt with early. Beyond the platform itself, the testing will be also extended to the developed serious (desktop and mobile) games, which will serve as important Learning Objects (LOs) discoverable within the ViPi portal. The deliverable will serve as the instrument that will guide the ViPi partners in the testing phase and prepare the ground for the pilot evaluation of the ViPi developments, where external stakeholders will be also involved. The purpose of this document is to present the necessary tests that the outcomes must undergo in order for all involved parties to confirm that both functional and non-functional aspects are operational and up to the expected level of quality. Overall, the testing and evaluation plan is a very important deliverable since it will allow the ViPi responsible partners to resolve any issues well in advance of the pilot phase, following the report on the findings of the internal testing phase.

The deliverable focuses on documenting a systematic approach for testing the ViPi outcomes, by defining the testing plan (functional and non-functional), as well as, the evaluation methodology (formative and summative) of the ViPi platform. This involves, among others, the definition of the procedures, the plan, the roles and responsibilities, as well as the identification of test scenarios and test cases.

More specifically, the document, after discussing the evaluation methodology, presents the set of test scenarios and test cases that will be used for validating the individual functional characteristics of each module of the ViPi platform and other outcomes (ViPi portal, e-Learning environment, ViPi mobile application, and desktop and mobile games). The test cases provide a framework for testing precisely the system's full functionality. In addition to the technical outcomes of ViPi, the non-technical outcomes (curriculum, training material and handbook) will be evaluated against the pre-defined objectives and characteristics. Furthermore, the document provides also some non-functional specifications that the ViPi platform (mainly) should meet. Since the ViPi developments are based on already existing and proved middleware platforms, no formal testing is foreseen for these non-functional specifications. However, to ensure that the platform will remain safe, reliable

and available under expected conditions, we will check the ViPi platform installation, focusing on: performance, scalability/upgradability, availability, usability, interoperability, and security.

The recording, monitoring and reporting of defects or failure to meet specifications, will be achieved through the use of the "The Bug Genie" issue reporting tool (ViPi instance accessible at http://www.atlec-project.eu/bugs/thebuggenie/), where all modules of the ViPi outcomes have been recorded and accounts have been created for all ViPi partners, with proper and sufficient access rights for reporting detected issues while testing.

This document is organized as follows:

- Section 2 discusses the available evaluation methodologies and presents the ones adopted in the ViPi project and the way they will be applied
- Section 3 presents the test scenarios and test cases that are defined per ViPi outcome as well as the reporting templates that offer a formal way of recording the test results and further refine the testing procedures, also defining roles and responsibilities of testing staff.
- Section 4, then, concludes the document.

2 Testing and evaluation methodology

To ensure the reliability of the ViPi products, the ViPi consortium, through the internal testing phase, will demonstrate that the level of quality offered complies with the required level of service. The state-of-art in this field proposes a rather wide, but still incomplete, set of methodologies and measurement systems for evaluation and certification activities. Among these methodologies, the most widely known that can be applied to software objects, are the *Capability Maturity Model* (CMM) for Software, by the *Software Engineering Institute* (SEI), and the ISO/IEC 14598 & 9126 series of standards, by the International Standards Organisation.

The CMM is the foundation for systematically building a set of tools, including maturity questionnaires, which are useful in software process improvement. The CMM provides a framework for organising the continuous process improvement steps into five maturity levels. These define an ordinal scale for measuring the maturity of a software process and for evaluating its capability. The CMM methodology is mainly oriented towards the assessment and improvement of software processes, and not towards the evaluation of software products.

A marked improvement in the evaluation of software product quality has been made with the issue of the international standard ISO/IEC 14598 (Software Product Evaluation), associated with the ISO/IEC 9126 standard (Software Product Quality). This series of standards has numerous advantages against other known methodologies, since it defines a generic enough evaluation framework that establishes connections with the quality characteristics of software products, embracing all aspects (not only the functional ones), also being adjustable to all technologies for software product development.

2.1 Purpose of Evaluation Methodologies

Evaluation methodologies aim to assist a project in fulfilling its aims, and to measure its outcome, both **formatively**, contributing to design decisions, and **summatively**, looking at the impact and the effects of the project results as a whole. The testing and evaluation methodology will ensure that the ViPi outcomes will meet with ergonomic and other pre-defined specifications and objectives of the project.

2.2 Selection of the ViPi evaluation methodology

The ViPi products to be delivered should be based on standards in order to adequately cover the needs of the end-users and help them migrate towards more efficient stakeholders-interactive

environments. Users (persons with disabilities, their trainers, VET centres, etc.) have more confidence in products and services conforming to international standards. To satisfy directly the prerequisite of product acceptance and thus indirectly to gain user trust, the ViPi products will have to be objectively accredited.

Therefore, the decision of the ViPi consortium is that the software outcomes' evaluation is based on a partial application of the ISO/IEC 14598 and the ISO/IEC 9126 series of standards (at the extent applicable), whereas the evaluation of all outcomes will adopt the "Think aloud protocol" (http://en.wikipedia.org/wiki/Think_aloud_protocol) in a customised manner.

2.2.1 The ISO/IEC 14598 Series of Standards

The *International Organization for Standardization* (ISO) and the *International Electro–technical Commission* (IEC) form a specialised system for worldwide standardisation. In the field of information technology, ISO and IEC have established the ISO/IEC JTC 1 joint technical committee.

The evaluation of software product quality is vital to both the acquisition and development of software that meets certain quality requirements. The relative importance of the various characteristics of software quality depends on the mission or objectives of the system of which it is a part; software products need to be evaluated to decide whether relevant quality characteristics meet the requirements of the system.

The essential parts of software quality evaluation are: (a) a quality model; (b) the method of evaluation; (c) the software measurement; and (d) the supporting tools.

2.2.2 The ISO/IEC 9126 Series of Standards

Each of the previously described ISO/IEC 14598 standards should be used in conjunction with the ISO/IEC 9126 standards providing software quality characteristics and metrics.

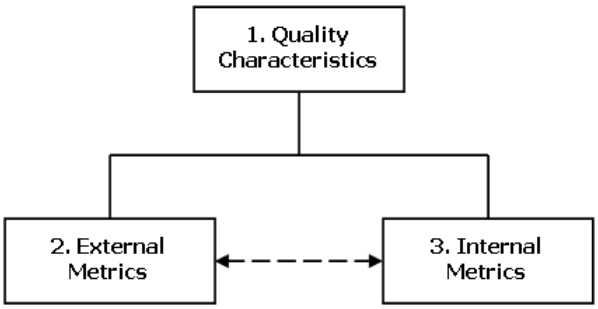

Figure 1: ISO/IEC 9126 Software Quality Characteristics and Metrics

ISO/IEC 9126 is comprised of three parts. Part 1 defines quality characteristics, associated sub–characteristics and the relations between the top two levels of the 9126 quality model. Parts 2 and 3 identify the relationships of the external (metrics that represent the external perspective of software

quality when the software is in use) and internal (metrics that measure internal attributes of the software related to design and code) metrics, to their corresponding characteristics and sub–characteristics.

The quality characteristics and sub-characteristics that are defined in the first part of the standard are given in the following:

- **Functionality** - A set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs.
    - Suitability
    - Accuracy
    - Interoperability
    - Security
    - Functionality Compliance
- **Reliability** - A set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time.
    - Maturity
    - Fault Tolerance
    - Recoverability
    - Reliability Compliance
- **Usability** - A set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users.
    - Understandability
    - Learnability
    - Operability
    - Attractiveness
    - Usability Compliance
- **Efficiency/Performance** - A set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions.
    - Time Behaviour
    - Resource Utilisation
    - Efficiency Compliance
- **Maintainability** - A set of attributes that bear on the effort needed to make specified modifications.
    - Analyzability

- Changeability
- Stability
- Testability
- Maintainability Compliance

- **Portability** - A set of attributes that bear on the ability of software to be transferred from one environment to another.
  - Adaptability
  - Installability
  - Co-Existence
  - Replaceability
  - Portability Compliance

2.2.3 The Think aloud protocol (TAP)

The Think-aloud protocol (or talk-aloud protocol) is a method used to gather data in, among others, usability testing in product design and development. The think-aloud method was introduced in the usability field by Clayton Lewis [Lewis, C. H. (1982). Using the "Thinking Aloud" Method In Cognitive Interface Design (Technical report RC-9265). IBM] while he was at IBM.

Think-aloud protocols involve participants thinking aloud as they are performing a set of specified tasks. Users are asked to say whatever they are looking at, thinking, doing, and feeling as they go about their task. This enables observers to see first-hand the process of task completion (rather than only its final product). Observers at such a test are asked to objectively take notes of everything that users say, without attempting to interpret their actions and words. The purpose of this method is to make explicit what is implicitly present in subjects who are able to perform a specific task.

2.3 The ViPi evaluation process

When evaluating software quality, it is necessary to establish the evaluation requirements, as well as specify, design and execute the evaluation.

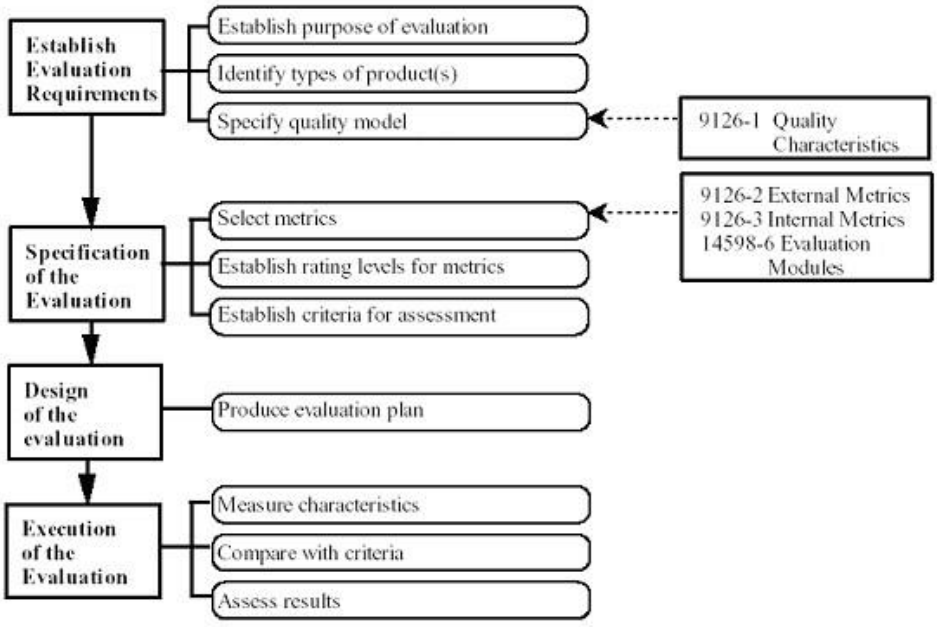

Figure 2: Evaluation Process

The ISO 14598 & 9126 standards and methods presented earlier will be applied at the extent possible and required, in order to match the needs of the ViPi software outcomes. The ViPi outcomes' evaluation process is depicted in Figure 2, and is further adjusted and described in the following sections.

2.3.1 Establishment of Evaluation Requirements

During the evaluation and testing plan phase, the purpose of the evaluation is established, the types of products to be evaluated are identified, and the quality model to be used is defined.

2.3.1.1 Objective

The ultimate objective of the testing and evaluation is to ensure that the product provides the required quality in use – that it meets the stated and implied needs of the users (including operators, recipients of the results of the software, or maintainers of software).

Other objectives of the evaluation process are to:

- identify problems at an early stage so that they can be rectified at a minimum cost
- compare the quality of the ViPi outcomes against stated user requirements
- validate the ViPi outcomes
- ensure that the objectives have been met

2.3.1.2 Identification of the individual outcomes to be evaluated and tested

The following ViPi outcomes are planned to be evaluated and tested:

(a) The ViPi Portal (including the social interaction facilities)

(b) The ViPi e-Learning environment

(c) The ViPi mobile application

(d) The ViPi desktop educational games

(e) The ViPi mobile educational games

(f) The ViPi Curriculum and Training material

(g) The ViPi Handbook

The requirements of the identified Building Blocks are analysed in this document in terms of test scenarios and test cases such as to extract the prioritised (in terms of functionality, reliability/availability, usability, security, performance, maintainability, and portability) needs of the ViPi products.

2.3.1.3 Specification of quality model

A quality model breaks software quality down into different characteristics. As described earlier in this chapter, the ISO/IEC 9126–1 standard provides a general–purpose model that defines six broad categories of characteristics of the software in use: functionality, reliability, usability, efficiency, maintainability and portability. These can be further broken down into sub–characteristics that have measurable attributes.

Internal software product quality attributes are the measurable properties of a software product that influence its ability to satisfy stated and implied needs. One or more attributes can be used to assess a particular software quality characteristic or sub–characteristic. Sufficient internal and external attributes should be identified for each required sub–characteristic.

2.3.1.4 Establishment of Rating Levels for Metrics

Quantifiable features are measured quantitatively using quality metrics. The results of measurement are values on the scales of metrics (e.g. a range 1 to 10, a binary response Yes/No, etc). This value does not itself show the level of satisfaction. For this purpose, the scale has to be divided into ranges corresponding to the different degrees of satisfaction of the requirements. Examples are:

- Dividing the scale into two categories: unsatisfactory and satisfactory

- Dividing the scale into four categories bounded by the current level for an existing or an alternative product, the worst case, and the planned level. The current level is stated in order to control that the new system does not deteriorate from the present situation. The planned level is what is considered achievable with the resources available. The worst case level is a boundary for user acceptance, in case the product does not fulfil the planned level.

The ISO/IEC 9126 proposes 4 rating levels: Excellent, Good, Fair and Poor. The first three are considered as satisfactory, the last is considered as unsatisfactory. In the case of ViPi, metrics scales, similar to those proposed by the ISO/IEC model will be employed, so as to avail to the user straight–forward questions (test cases) that can be answered in the simplest form possible and to ensure that the answers are valid and contributing to the evaluation. The rating levels of each scale will in turn be associated with grades that when calculated will be providing the final grade of the evaluation product.

However, it has to be mentioned that the user will be also requested at times to share personal views and opinions using physical language (by commenting next to the test cases). Depending on the possible answers, ratings will be established in this case too.

2.3.1.5 Applied evaluation classes

In practice, the evaluation task falls into two different areas:

- **Formative evaluation** (ways to gain feedback on the software product while it is being developed); and
- **Summative evaluation/product evaluation** (ways to show that what was originally proposed about the software product has or still has to be fulfilled).

Another way of conceptualising this is that formative evaluation is concerned with the outcomes of the software development stages. On the other hand, summative evaluation / product evaluation is concerned with the stated objectives the software has to fulfil.

The present evaluation methodology mostly addresses the summative evaluation class, since it aims at evaluating the attributes of the finished products, the consequences, the success or the efficiency of the outcomes of ViPi and the extent to which they meet the set objectives. On a strategic level, summative evaluation will aim to assess the outcomes and impact of the products as a whole. A formative evaluation of all products had taken place during the development phase and by each responsible partner separately. As the two evaluation modes are complementary and the process of formative evaluation may even be an important component in the summative evaluation, some of the test cases will be answered by the developers, with knowledge acquired at the development stage.

2.3.1.6 ViPi testing and evaluation time–plan

This section specifies the timeplan for the testing and evaluation work to be undertaken within the project, containing the evaluation activities, their starting dates, duration and methods to be applied. The ViPi project contract divides activities into ten (10) work packages. The internal testing activities will take place in several stages orientated on these packages: Beginning from the internal formative testing during the development phases in WP3 and WP4, followed by the development of this summative evaluation plan as an early task in WP6, and resulting in the evaluation results by the end of WP6. Then WP7 will deal with the pilot testing of the ViPi products, by external stakeholders.

The above is depicted in the following table:

Stage One Evaluation Activities (WP3&4)

ID	Date	Activity
1.	2012	Internal formative testing during the development of each module

Stage Two Evaluation Activities (WP6)

ID	Date	Activity
2.	Sep 2012-Nov 2013	Elaboration of a Testing plan and evaluation methodology
3.	Sept 2012 - March 2013	Local training workshops where the testing plan will be presented to the internal testers as well as the external stakeholders in view of the pilot phase
4.	Dec 2012 - April 2013	Collection and analysis of the testing and evaluation results

Stage Three Evaluation Activities (WP3&WP4&WP6)

ID	Date	Activity
5.	April 2013	Addressing the findings of the testing phase, by refining the developments.

Stage Four Evaluation Activities (WP7)

ID	Date	Activity
6.	May - October 2013	The pilot test evaluation

The following chapter presents the detailed testing plan, defining the test cases and the test scenarios as well as the test-results reporting templates that will guide the testing and evaluation activities and facilitate the collection of results.

3 Testing and evaluation plan

3.1 Functional testing of technical (software) outcomes

3.1.1 Test scenarios and test cases

The following subsections define all test scenarios and test cases for the ViPi platform and other outcomes, The purpose of these scenarios and cases is to systematically verify the correct functioning of the ViPi outcomes, by executing all test cases of the following sections. The identified test scenarios verify the soundness of the functionality of the respective outcome, through the execution of the defined test cases.

3.1.2 Test scenarios and test cases for the ViPi portal (including the social interaction tools)

Test scenario	Test Case	Comments
[TS-01] User registration	[TC-01] User registers	
	[TC-02] Cancel submission of information for registering	
	[TC-03] Provide insufficient data	
	[TC-04] Register existing user	
	[TC-05] Failed verification of registration data	
[TS-02] User login	[TC-01] Normal login	
	[TC-02] Cancel submission of login information	
	[TC-03] Provide insufficient data	
	[TC-04] Forget password procedure	
[TS-03] Edit profile	[TC-01] Access the profile edit page	
	[TC-02] Fill-in missing profile data	
	[TC-03] Provide insufficient data	
	[TC-04] Annotate profile using conceptual tree (including applying for a specific role(s))	
	[TC-05] Make conflicting selections in the annotations	
[TS-04] Browse public sections of portal	[TC-01] Visit the Home page	
	[TC-02] Visit the About page	
	[TC-03] Visit the Learning Objects page	
	[TC-04] Visit the Activity page	
	[TC-05] Visit the Members page	

	[TC-06] Visit the Groups page	
	[TC-07] Visit the Contact page and submit a message	
	[TC-08] Perform search with text in different languages	
	[TC-09] Change display language	
	[TC-10] Visit the links on the footer of the page	
[TS-05] Search for learning objects (not logged-in)	[TC-01] Perform a simple text-based search	
	[TC-02] Perform a simple text-based search with text in different language than the one currently displayed	
	[TC-03] Perform guided search by making selections on the tree	
	[TC-04] Make conflicting selections	
	[TC-05] Select multiple top-level nodes	
	[TC-06] Select multiple nodes within a top-level node	
[TS-06] Search for learning objects (logged-in)	[TC-01] Perform simple text-based search	
	[TC-02] Perform same search but choosing also to consider your profile (not annotated the profile)	
	[TC-03] Perform same search but choosing also to consider your profile (having annotated the profile)	
	[TC-04] Perform guided search	
	[TC-05] Perform guided search but choosing also to consider your profile (not annotated the profile)	
	[TC-06] Perform guided search but choosing also to consider your profile (having annotated the profile)	
	[TC-07] Perform guided search, considering also your profile but making conflicting selections	
[TS-07] Browse through LO results	[TC-01] Open an LO from the results-set	
	[TC-02] Browse through the contents of an LO	
[TS-08] Create and submit learning objects	[TC-01] Access the LO creations page	
	[TC-02] Fill-in the required data and click submit	
	[TC-03] Provide insufficient data and click submit	
	[TC-04] Annotate the LO using the conceptual tree	
	[TC-05] Make conflicting selections	
	[TC-06] Select multiple top-level nodes	
	[TC-07] Select multiple nodes and leafs within a top-level node	

	[TC-08] Search for the currently submitted LO (not yet approved)	
	[TC-09] Search for the currently submitted LO (already approved)	
[TS-09] Social communication with other community members	[TC-01] Read through the current activity on the portal by community members	
	[TC-02] Browse through the other members of the portal	
	[TC-03] Check existing community groups on the portal	
	[TC-04] Access an LO and submit a comment	
	[TC-05] Access an LO and rate it	
	[TC-06] Access the same LO and try to rate it again	
	[TC-07] Create a group	
	[TC-08] Join group	
	[TC-09] Interact within a group	
[TS-10] Approve LO to be published	[TC-01] Login as admin	
	[TC-02] Access the pending LOs for moderation	
	[TC-03] Open and check the content of the LO	
	[TC-04] Approve for publication	
	[TC-05] Reject publication	
[TS-11] Create/Approve user	[TC-01] Access the Users page in the admin panel	
	[TC-02] Create a new user	
	[TC-03] Check list of pending profile-roles for approval	
	[TC-04] Remove user	

3.1.3 Test scenarios and test cases for the ViPi e-Learning environment

Test scenario	Test Case	Comments
[TS-12] User login/logout	[TC-01] User logs into ViPi Portal (see use-case [VP-02])	User has already a proper account in the ViPi portal.
	[TC-02] User clicks on “Access to the online training environment”	User is logged-in the ViPi portal.
	[TC-03] User logs out from within the learning environment	User is logged-in the ViPi portal and has visited the e-learning environment.
	[TC-04] User returns to the ViPi Portal and logs out from there	User is logged-in the ViPi portal and has

		visited the e-learning environment.
[TS-13] Browse through training content	[TC-01] User browses through courses and performs several tasks	User is logged-in the ViPi portal and has visited the e-learning environment.

3.1.4 Test scenarios and test cases for the ViPi mobile application

Test scenario	Test Case	Comments
[TS-14] Application download and installation	[TC-01]	
	[TC-02]	
	[TC-03]	
[TS-15] Start application	[TC-01]	
	[TC-02]	
[TS-16] Edit settings	[TC-01] Edit username/password and save	Already have an account on ViPi platform
	[TC-02] Edit language settings	
	[TC-03] Edit preferences (comments notification and save)	
	[TC-04]	
[TS-17] Browse through general ViPi content	[TC-01] Click on “Other pages” button	
	[TC-02] Browse through the content of the pages	
[TS-18] Guided search of LOs	[TC-01] Click on “Learning Objects” button	
	[TC-02] Select nodes from the ontology	
	[TC-03] Activate search without asking to consider user’s profile	
	[TC-04] Activate search with asking to consider user’s profile	
	[TC-05] Browse through the search results	
[TS-19] Accessing learning objects	[TC-01] Sharing the LO to social media	
	[TC-02] Access the online version of the LO	
	[TC-03] Comment on the LO	
[TS-20] Quick search	[TC-01] Click on “Quick search” button	
	[TC-02] Enter keyword(s) and activate search	
	[TC-03] Browse through the search results	
[TS-21] Comments	[TC-01] Click on “Comments” button to access	

	the comments	
	[TC-02] Browse through the comments of specific LOs and pages	
	[TC-03] Reply to comment	
	[TC-04] Delete comment	
[TS-22] Interface components	[TC-01] Dashboard button	
	[TC-02] Synchronization button	

3.1.5 Test scenarios and test cases for the ViPi desktop games

Test scenario	Test Case	Comments
[TS-23] Download and install	[TC-01] Download Escapology game installer	
	[TC-02] Install Escapology game	
[TS-24] Play the game	[TC-01] Select language	
	[TC-02] Start game	
	[TC-03] Browse through all questions (question, clue and rationale)	
	[TC-04] Keyboard usage	
	[TC-05] Mouse usage	
	[TC-06] Outcome of playing	
	[TC-07] After finalization of game-level, redirection to previous menu	
[TS-25] Download and install	[TC-01] Download Rob the Mob game installer	
	[TC-02] Install Rob the Mob game	
[TS-26] Play the game	[TC-01] Select language	
	[TC-02] Start game	
	[TC-03] Browse through all questions (question, possible answers and rationale)	
	[TC-04] Keyboard usage	
	[TC-05] Mouse usage	
	[TC-06] Outcome of playing	
	[TC-07] After finalization of game-level, redirection to previous menu	
[TS-27] Download and install	[TC-01] Download the game editor	
	[TC-02] Install the game editor	
[TS-28] Use the editor	[TC-01] Select language	
	[TC-02] Select game	
	[TC-03] Select fields to be completed	
	[TC-04] Edit (also remove) questions and categories and save	

	[TC-05] Edit (also remove) answers and clues of specific questions and save	
	[TC-06] Activate shortcuts on desktop for the games	
[TS-29] Download and install	[TC-01] Download the correct language version of the Yes or No game	
	[TC-02] Install the downloaded game	
[TS-30] Play the game	[TC-01] Keyboard input	
	[TC-02] Mouse usage/input	
	[TC-03] Browse through the questions	
	[TC-04] Outcome of playing	
	[TC-05] After finalization of game-level, enter name and save outcome as pdf	
	[TC-06] Go back to previous menu	
[TS-31] Download and install	[TC-01] Download the correct language version of True of False game	
	[TC-02] Install the downloaded game	
[TS-32] Play the game	[TC-01] Keyboard input	
	[TC-02] Mouse usage/input	
	[TC-03] Browse through the questions	
	[TC-04] Outcome of playing	
	[TC-05] After finalization of game-level, enter name and save outcome as pdf	
	[TC-06] Go back to previous menu	
[TS-33] Download and install	[TC-01] Download the Stay Safe installation package	
	[TC-02] Install the downloaded package	
[TS-34] Edit option	[TC-01] Select language	
	[TC-02] Select TTS	
	[TC-03] Select display resolution	
	[TC-04] Access the Author (game editor)	
[TS-35] Play the game	[TC-01] Keyboard input	
	[TC-02] Mouse usage/input	
	[TC-03] Browse through the questions (provide all possible different answers to each question)	
	[TC-04] Check explanations and instructions given for each question	
	[TC-05] Outcome of playing	
	[TC-06] Go back to previous menu	
[TS-36] Game Editor 1	[TC-01] Access conversation editor	
	[TC-02] Select language	

	[TC-03] Select conversation and hub	
	[TC-04] Select a question hub	
	[TC-05] Edit answer boxes	
[TS-37] Game Editor 2	[TC-01] Access screen translation tool	
	[TC-02] Select language	
	[TC-03] Select screen and item	
	[TC-04] Edit text in boxes	
[TS-38] Access Fly Swat game	[TC-01] Go to Fly Swat URL	
	[TC-02] Play version using mouse	
	[TC-03] Play version using keyboard	
	[TC-04] Check outcome of game	
	[TC-05] Check use of buttons on intro screen to change game modes (sound, fly speed, target, fly movement mode, swat mode)	
	[TC-06] Check use of buttons in game play to change game modes (sound, target)	

3.1.6 Test scenarios and test cases for the ViPi mobile games (Memobile series of games)

Test scenarios	Test Case	Comments
[TS-39] Mobile Settings	[TC-01] Device settings for installations	Assume the device is not accepting APK (outside Google Play Store) The user enables the displayed option:Settings >> Applications >> Unknown sources. The user enables the displayed option: Settings >> Security >> Unknown sources.
[TS-40] Application Download	[TC-01] Download APK file from the Google play	Assume MEMOBILE.APK file is not present on the device. The user downloads MEMOBILE.APK file through internet browser on the device from *http://www.putaddresshere.com*
	[TC-02] APK file transfer	Assume MEMOBILE.APK file is not present on the device. The user copies MEMOBILE.APK file on to device from a different source (PC as direct download, email as attachment, mobile device as file transfer).

[TS-41] Application Installation	[TC-01] Installation of application	Assume MEMOBILE application is not installed in the device. The user access to: *Settings >> Applications >> Download* and executes MEMOBILE application. The user has to answer to: "*Do you want to install this application?*", by clicking *Install* button, then clicking *Open*.
	[TC-02] Installation of Adobe Air	Assume Adobe Air application is not installed in the device. The user has to accept Adobe Air installation request: *"This application requires Adobe AIR. To continue, install Adobe AIR on this device"* by clicking *Install.* Android Market will open the user has to select *Install*, then *Accept*.
[TS-42] General settings	[TC-01] Disclaimer	The user clicks on *Disclaimer* button and then on *Close* button.
	[TC-02] ESC button	The user clicks on *ESC* button in every page.
	[TC-03] GO HOME button	The user clicks on *GO HOME* button in every page.
[TS-43] Matching Pairs game	[TC-01] How To Play	The user clicks on *How To Play* button and then on *Next* button three times.
	[TC-02] Play Game button	The user clicks on *Play Game* button.
	[TC-03] Matching Pairs game	The user clicks on *Matching Pairs* button.
	[TC-04]	The user clicks on *Play Game* button.
	[TC-05]	The user plays the game clicking on the twelve numbered boxes on screen.
	[TC-06]	The user clicks on *Learn And Play* button and then on *Next* button thirteen times to go through the instruction pages. In each page the button *Skip* takes the user to *Matching Pairs* game start screen.

	[TC-07]	The user clicks on *Next* button.
	[TC-08]	The user clicks once on *Play Again* and then once on *Main Menu* button.
[TS-44] Starter Kit game	[TC-01]	The user clicks on *Starter Kit* button.
	[TC-02]	The user clicks on *Learn and Play* button.
	[TC-03]	The user clicks on *Learn* button and then on *Next* button three times to go through the instruction pages. In each page the button *Skip* takes the user to the *Starter Kit* game start screen.
	[TC-04]	The user clicks on *Learn* button and then on *Next* button three times to go through the instruction pages. In each page the button *Skip* takes the user to the *Starter Kit* game start screen.
	[TC-05]	The user clicks on *Play Game* button.
	[TC-06]	The user clicks on each button and plays the three different levels of the game. If the maximum score is set, a *Next* button is shown
	[TC-07]	The user clicks once on the *Play Again* and then once on the *Main Menu* button.
[TS-45] Press And… Action game	[TC-01]	The user clicks on the *Press And… Action* button.
	[TC-02]	The user clicks on the *Learn and Play* button.
	[TC-03]	The user clicks on the *Learn* button and then on the *Next* button nine times to go through the instruction pages. In each page the button *Skip* takes the user to the *Press And… Action* game start screen.
	[TC-04]	The user clicks on *Play Game* button twice.
	[TC-05]	The user plays *Press And… Action* game by selecting the correct answer to fifteen questions.

		If the maximum score is set, a *Play Extra Level* button is shown
	[TC-06]	The user clicks once on *Play Again* and then once on *Main Menu* button.
	[TC-07]	The user clicks on *Play Extra Level* button, then on *Play Game* button. The user plays *Extra Level* by selecting the correct answer to fifteen questions
	[TC-08]	The user clicks once on *Play Again* and then once on *Main Menu* button.

3.2 Testing of non-technical outcomes

3.2.1 Test cases for the ViPi Curriculum, Training Content and Handbook

These are outcomes of the ViPi project that do not comprise software developments. They comprise documents which aim to serve specific purposes for their readers. The testing of these outcomes is not directly linked to the methodologies defined for the rest of the outcomes. Instead, a list of pre-defined objectives/specifications has been established, together with instructions for measuring the success towards meeting each of these objectives. The testers will be called to provide success/failure statement, also giving their comments while using the non-technical outcomes.

Outcome	ViPi Curriculum, training content and handbook

Pre-defined objectives	**Measurement execution**	**Other Comments**
Broad coverage of basic ICT Skills (only relevant for the Curriculum and Training content)	Assessing per-unit with core stakeholders	
Accessibility	Is the content of the document fully accessible (assess with end-users with different disabilities)	

Pre-defined objectives	Measurement execution	Other Comments
Assessment questions (only relevant for the Curriculum and Training content)	Are they at suitable difficulty level? Do they cover the offered training content?	
Completeness	Is level of detail for each document section, adequate? Any topics missing?	
Understand-ability	Is level of detail adjusted to the target user groups? Is the language used appropriate for the target groups?	
Localization (e.g. translation to local language)	Was the content in local language understandable?	

3.3 Test reporting templates

The purpose of the test reporting templates, as presented in this section, is to offer a formal way of reporting the test results of the test cases defined in the previous section. These templates will be used during the testing phases and filled-in by the testing groups, such as to be included in the final testing-results reports.

3.3.1 Test reporting template for the technical outcomes

The following is a sample table. A separate table must be created for reporting the test results per test scenario defined in the previous sections.

Test Scenario	**[TS-01]**				
Test Cases	ID	Date	Tested by	Results (Excellent-E, Good-G, Fair-F and Poor-P)	Think-aloud comments and issues.
	[TC-01]	01 Apr 2013	George Milis	**Functionality** Suitability: G Accuracy: G Compliance with specs: G	I had difficulty in operating the ... due to the issue I have with my hands...

				Usability Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G	

				Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	
				Functionality Suitability: G Accuracy: G Compliance with specs: G **Usability** Understandability: G Learnability: G Operability (including accessibility): F Attractiveness: G Compliance with specs: E	

3.3.2 Test reporting template for the non-technical outcomes

The following table is simply a sample. Separate tables will need to be filled-in for each testing session and for each tested outcome or part of it. The outcomes that are tested with this template are the: ViPi Curriculum, ViPi Training Content and ViPi Handbook.

Outcome	ViPi Curriculum, Section 2.7			
Pre-defined objectives	**Date**	**Checked by**	**Results (Excellent-E, Good-G, Fair-F and Poor-P)**	**Think-aloud comments and issues.**
Coverage of basic ICT Skills	01 Apr 2013	George Milis		
Accessibility Is the content of the document fully accessible (assess with end-users with different disabilities)				
Assessment questions Are they at suitable difficulty level? Do they cover the offered training content?				
Completeness Is level of detail for each document section, adequate? Any topics missing?				
Understand-ability Is level of detail adjusted to the target user groups? Is the language used appropriate for the target groups?				

Localization (e.g. translation to local language) Was the content in local language understandable?				

3.4 Non-Functional testing

The following sections provide brief description of the non-functional characteristics for which the ViPi technical outcomes should offer a good level of compliance. No formal specifications have been defined in the ViPi contract for such characteristics, however, the consortium decided to address them at the extent possible, in order to ensure that the ViPi technical outcomes will adhere to high standards and be sustained in the long run.

The non-functional characteristics are taken from the metrics of the standard ISO-9126 quality metrics standard. As an exception, the interoperability and security sub-characteristics, from the functionality characteristic of the standard, are addressed in this section, since these could not efficiently and easily be tested by the formal functionality testing.

3.4.1 Functionality - Interoperability

The hardware which is used for hosting the ViPi platform is purchased by a well established hosting provider, who adopts sufficient procedures for complying with the market standards for hardware interoperability.

Concerning the software of the platform, all ViPi developments are based on the adoption of open source software (WordPress, BuddyPress and ATutor platforms, Android) and open standards (SCORM for learning objects, XML-RPC, etc.) which ensures the interoperability of the provided solutions with solutions available in the market and also with any further developments to be assumed by the open source software community.

No critical interoperability deficiencies are expected to be experienced during the operation of the ViPi outcomes.

3.4.2 Security (Functionality)

There will be several thorough tests to be performed by the ViPi partners in order to verify that the system is as secure as possible. The objective of the tests will be to immediately identify and rectify “security holes” of the system hardware, configuration and/or software. In parallel, the security tests will be able to identify any security weaknesses, forming the basis for future corrective steps.

The security tests will focus on trying to access the software functionalities with user-roles with no sufficient access rights, trying to inject malicious code through the points of interaction with the platform, etc.

To ensure that no security holes remain open, the ViPi operating team will always keep the underline platforms up to date. This will require that the ViPi developments are also kept updated to comply with new versions of the platforms, e.g. Wordpress core, and ATutor core.

3.4.3 Reliability

As described in the ISO-9126 standard, the reliability of a software outcome is, further to the compliance with any particular specifications, associated with the maturity of the software, its fault-tolerance and its recoverability.

No formal reliability specifications have been set in the ViPi contract. However, since the developments are based on very mature open source platforms that are supported by active communities, the maturity and fault-tolerance of the tools is secured.

Moreover, the hosting of the ViPi platform and other tools, is undertaken by professional and well-known and reliable hosting providers. These providers take care also of backing-up the hosted data. In addition, the ViPi portal will use its own plugin for email back-up of the content database, which will allow full recoverability.

3.4.4 Efficiency-Performance

This item mainly applies to the ViPi platform modules (portal and e-Learning environment). The software installation must be able to achieve sufficient response times for the expected number of concurrent users (e.g. 50), which form reasonable performance quality metrics for the platform. The ViPi consortium and external stakeholders will be asked to make use of the platform simultaneously, performing light but also heavy tasks (e.g. guided LO search or submission of LOs). Then they will all asked to report back with their feelings with respect to the time behaviour of the system.

Further to the above, stress testing will be done by Hypertech (main development partner), using also available testing software tools.

3.4.5 Maintainability - Scalability

The scalability of the ViPi platform is ensured since all developments are based on mature and highly supported open source software platforms. The communities behind these platforms are active and perform updates and different types of changes from time to time. These changes are offered to all users of these platforms in the form of batch updates.

The ViPi developments took into consideration that such updates will be performed during the lifetime of the services, therefore, all functionality was offered in the form of plugins to allow updating without breaking.

The ViPi exploitation strategy will contain also details on how the ViPi developments will be maintained by the involved partners.

3.4.6 Portability

A detailed handbook is provided as part of the ViPi outcomes, which offers guidance on how to install and configure the ViPi games, mobile application, etc., and also how to use the ViPi portal and learning environment (no installation will be required by end-users for the latter).

No formal testing is foreseen on the issue of portability.

4 Conclusions

The present document presented the adopted methodology for the testing of the ViPi outcomes, also providing a set of templates to be used by the testing staff to report their findings. All testers are expected to use these templates, complete them while testing and then hand them back to the responsible persons for collecting the results.

An effort was made such as to avoid designing a very complex testing and evaluation plan, while at the same time ensure the high quality evaluation of the project outcomes. As can be seen in the document, the ISO-IEC-9126 quality metrics standard, as well as the Think-aloud protocol have been adopted in the plan, however, only parts of them are actually used in the templates, based on their actual necessity for the type and scope of the ViPi outcomes.

The results of the testing phase, will be separately reported in due time, in the deliverable D21 of the ViPi project.