Project no. FP6-028038

Palette

Pedagogically sustained Adaptive LEarning Through the exploitation of Tacit and Explicit knowledge

Instrument: Integrated Project

Thematic Priority: Technology-enhanced learning

D.MAN.07 – Open Source Strategy (draft)

Due date of deliverable: 31 August 2007
Actual submission date: 19 November 2007

Start date of project: 1 February 2006 Duration: 36 months

Organisation name of lead contractor for this deliverable: GEIE ERCIM

Project co-funded by the European Commission within the Sixth Framework Programme		
Dissemination Level		
P	Public	**PU**

Keyword List: Licenses, Open Source
Responsible Partner: Karen Marache, GEIE ERCIM

MODIFICATION CONTROL			
Version	Date	Status	Modifications made by
1.0	23-07-2007	Draft	Karen Marache
1.1	25-07-2007	Draft	Vincent Quint
1.2	10-09-2007	Draft	Karen Marache
1.3	29-10-2007	Draft	Karen Marache
1.4	5-10-2007	Draft	Bruno Le Dantec
1.5	12-11-2007	Draft	Vincent Quint
1.6	19-11-2007	Final	Karen Marache

Deliverable manager

- Karen Marache, GEIE ERCIM

List of Contributors

- Karen Marache, GEIE ERCIM
- Vincent Quint, INRIA Rhône Alpes
- Rose Dieng, INRIA Sophia
- Nikos Karacapilidis, CTI
- Denis Gillet, EPFL
- Thibault Latour, CRP-HT
- Bruno Le Dantec, GEIE ERCIM

List of Evaluators

- Christine Vanoirbeek, EPFL
- Vincent Quint, INRIA Rhône Alpes

Summary

This deliverable provides a list of licenses applying/to be applied to the existing PALETTE tools and services, and describes the methodology to be used to define the better Open Source strategy for these services, taking into account the rights and responsibilities of the source code owners as well as those of contributing developers.
The PALETTE consortium decided to collaborate with representatives of the QualiPSo European project, whose goal is to help industries and organizations with Open Source Software strategies (more info, www.qualipso.org).

The findings of this document will impact the PALETTE Exploitation Plan.

The final version of this document will be delivered by end of January 2008.

1 – Introduction

This deliverable describes the situation regarding the tools and services' licenses already granted within the PALETTE project, and presents a methodology to define the better Open Source strategy for the PALETTE services. The project having entered in the second half of its life the question of the dissemination and of the valorisation of the developed tools and services has to be analysed.

The first step consisted in identifying and understanding the different categories of services and the types of Open Source Licenses that could be applied to those services. Currently, most of the PALETTE services are developed by individual partners, but one must not forget that the main objective of the project is to make them integrated and interoperable. For this reason, the Consortium needs to understand the characteristics of existing licenses which could be applied to the PALETTE services. This will be analysed in the first part of this deliverable.

The second part presents a state of the situation concerning the PALETTE services already developed, or in development, in terms of licenses. The motivations of the choice of license, as well as the institutional constraints are provided for each service. This information was collected using an internal questionnaire prepared by INRIA Rhône-Alpes

In order to prepare the best exploitation plan for those results (choice of licences, management of future contributions, etc), the PALETTE Steering Committee decided to cooperate with the QualiPSo European Project. This project works on an IPR tracking methodology of very high importance for PALETTE. It will help to determine if the services prepared by PALETTE are free of protected know how, thus helping us to determine the best Open Source strategy.

Finally, the last part of the deliverable presents the draft questionnaire provided by QualiPSo and used to start the audit of the PALETTE tools.

2 – Open source & free software licensing

This section describes the more common licenses which could be used for the PALETTE services. It is not an exhaustive list and may evolve before the end of the project.

2.1 List of free and non-free software[1]

The main categories of free and non-free software are briefly described hereunder:

Free software or open source software

Free software is software that comes with permission for anyone to use, copy, and distribute, either verbatim or with modifications, either gratis or for a fee. In particular, this means that source code must be available.

Copylefted software

Copylefted software is free software whose distribution terms do not let redistributors add any additional restrictions when they redistribute or modify the software. This means that every copy of the software, even if it has been modified, must be free software.

Non-copylefted free software

Non-copylefted free software comes from the author with permission to redistribute and modify, and also to add additional restrictions to it.

[1] http://www.gnu.org/philosophy/categories.html

GPL-covered software

The GNU GPL (General Public License) is one specific set of distribution terms for copylefting a program.

Semi-free software

Semi-free software is software that is not free, but comes with permission for individuals to use, copy, distribute, and modify (including distribution of modified versions) for non-profit purposes.

Proprietary software

Proprietary software is software that is not free or semi-free. Its use, redistribution or modification is prohibited, or requires you to ask for permission, or is restricted so much that you effectively can't do it freely.

Shareware

Shareware is software which comes with permission for people to redistribute copies, but says that anyone who continues to use a copy is *required* to pay a license fee.

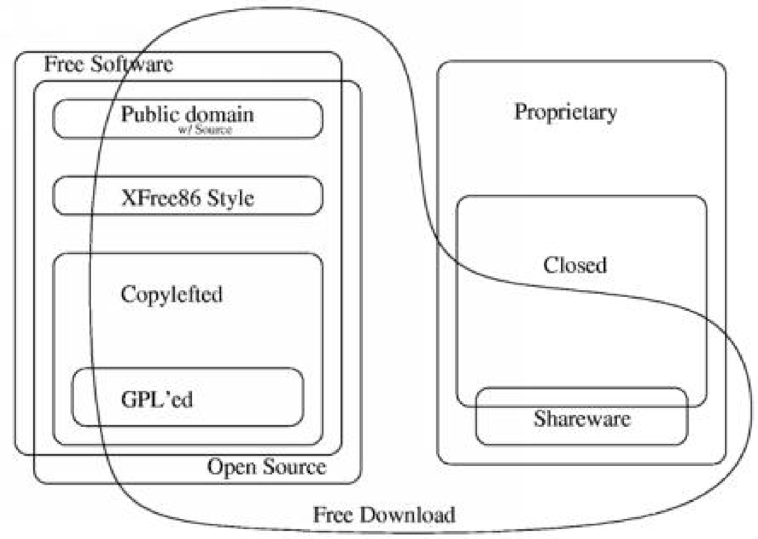
Private software

Private or custom software is software developed for one user (typically an organization or company). That user keeps it and uses it, and does not release it to the public either as source code or as binaries.

Commercial software

Commercial software is software being developed by a business which aims to make money from the use of the software. “Commercial” and “proprietary” is not the same thing! Most commercial software is proprietary, but there is commercial free software, and there is non-commercial non-free software.

The main categories can be visualised by the **Chao-Kuei Diagram**[2]:



All members of the PALETTE Consortium agreed that all services produced or to be produced are meant to be available to a large public. To achieve this goal, all PALETTE services will use an Open Source License.

[2] http://www.gnu.org/philosophy/categories.html

2.2 Open Source licenses

The term “Open Source” is most commonly applied to the source code of computer software that is available to the general public with non-existent intellectual property restrictions. This allows users to develop software content in a public, collaborative manner. It also permits users to use, change and improve the software, and to redistribute it in modified or unmodified form.

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria[3]:

a) Free redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

b) Source code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a pre-processor or translator are not allowed.

c) Derived works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

d) Integrity of the author's source code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

e) No discrimination against persons or groups

The license must not discriminate against any person or group of persons.

f) No discrimination against fields of endeavour

The license must not restrict anyone from making use of the program in a specific field of endeavour. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

g) Distribution of license

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

h) License must not be specific to a product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

[3] The Open Source Definition, updated 2006/07/07 - http://www.opensource.org

i) License must not restrict other software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

j) License must be technology-neutral

No provision of the license may be predicated on any individual technology or style of interface.

Knowing these principles, it is obvious that the PALETTE services must be developed under Open Source licenses framework. All the licenses mentioned in the following section are GPL-compatible licenses.

2.2.1 List of licenses qualifying as free software licenses, compatible with the GNU GPL (GNU general Public License)[4]

GNU General Public License, or GNU GPL for short.

This is a free software license, and a copyleft license.

Public Domain

Being in the public domain is not a license; rather, it means the material is not copyrighted and no license is needed. Practically speaking, though, if a work is in the public domain, it might as well have an all-permissive non-copyleft free software license. Public domain material is compatible with the GNU GPL.

Apache License, Version 2.0

This is a free software license, compatible with version 3 of the GPL. Please note that this license is not compatible with GPL version 2, because it has some requirements that are not in the older version. These include certain patent termination and indemnification provisions.

FreeBSD license

This is the original BSD license with the advertising clause and another clause removed. (It is also sometimes called the “2-clause BSD license”.) It is a simple, permissive non-copyleft free software license, compatible with the GNU GPL.

Modified BSD license

This is the original BSD license, modified by removal of the advertising clause. It is a simple, permissive non-copyleft free software license, compatible with the GNU GPL.

CeCILL version 2

The CeCILL is a free software license, explicitly compatible with the GNU GPL.

With CeCILL-B, the authors allow their software to be reused with no restrictions other than an obligation to mention sources (documentation, software interface, web site). CeCILL-C is for the distribution of libraries and software components in general. Anyone distributing an application which includes components under the CeCILL-C license must mention this fact and make any changes to the source code of these components available to the community

W3C Software Notice and License

This is a free software license and is GPL compatible.

[4] Copyright © 1999 Richard M. Stallman (http://www.gnu.org/licenses/license-list.html)

X11 License

This is a simple, permissive non-copyleft free software license, compatible with the GNU GPL. Older versions of XFree86 used the same license, and some of the current variants of XFree86 also do. Later versions of XFree86 are distributed under the XFree86 1.1 license (which is GPL-incompatible).This license is sometimes called the *MIT license*, but that term is misleading, since MIT has used many licenses for software.

3 – State of the situation

Four of the PALETTE partners are developing services and will therefore be directly impacted by the chosen open source strategy: EPFL, INRIA, CTI, and CRP Henri Tudor.

These organizations, so-called "technical" partners, have strong interactions with the "pedagogical" partners for specifying, reviewing, and experimenting the software under development. Strong interactions also exist between the technical partners themselves for integrating and combining their various pieces of software, and also for insuring interoperability between services. Every software falls as much as possible under the responsibility and is implemented by one single partner (with one exception: DocReuse). On one hand, this simplifies the Intellectual Property Rights issue but on the other hand this can lead to potential heterogeneity in software licenses across the Palette project.

These four partners, in close cooperation with the other members of the consortium, based on the identification of the existing licenses and on their institute internal strategy, will identify and apply as much as possible the same license model to all PALETTE services. It has to be understood that some of the services components are software that come with their own licenses, which may be proprietary, commercial or open source.

The following section reviews the partner's internal strategy regarding the choice of the licenses and how this strategy will be applied or not to the developed services in PALETTE. This information has been collected through the following questionnaire:

1/ What is the policy of your institution (EPFL, INRIA, CTI, CRP Henri Tudor), i.e. what are the constraints (or the flexibility) you have to choose a license for the software you are developing for the PALETTE project?

2/ What is the current license (if any) of the PALETTE software you are developing? What are the main reasons for choosing this license? If the final choice is not made yet, what options are you currently considering, and why?

3.1 EPFL

Policy: EPFL supports an open source policy and recommends GPL.

e-Logbook

e-Logbook is a collaborative web-based environment offering mediation and awareness services to Communities of Practice. It consists of an activity-oriented shared space where members can manipulate stored assets.

eLogbook is developed with Ruby on Rails. Ruby uses the MIT license eLogbook requires three external components: Mysql 5, Postfix 2.4.1, and Getmail 4.7.3.

- Mysql 5 and Getmail 4.7.3 are under GNU General Public License (GPL) Version 2
- Postfix 2.4.1 is under IBM Public License Version 1.0 -- Secure Mailer

As a consequence, e-Logbook should most probably be published under GNU General Public License (GPL), which is consistent with EPFL's policy. The MIT license could also work, but e-Logbook should then be distributed separately from the components mentioned above.

DocReuse

DocReuse is a tool enabling the semi-automatic reuse of web documents within Communities of Practice. DocReuse takes as input a set of source documents, adapts them to user needs and constraints in order to produce a new document. To automate such a process, DocReuse makes use of document structure and explicit representation of semantic information.

No license yet. Note that DocReuse is developed jointly by EPFL and University of Fribourg

3.2 INRIA

Policy: For open source software INRIA is promoting the CeCILL family of licenses (these licenses were created jointly by three public research organizations in France: CEA, CNRS and INRIA http://www.cecill.fr/index.en.html).

Corese

Corese is a semantic search engine. It is an RDF engine based on Conceptual Graphs (CG). It enables the processing of RDF Schema and RDF statements within the CG formalism.

For Corese the CeCILL-C license was preferred in order to take benefit of any changes to the source code by the community.

Sewese

Sewese is a semantic web application development platform built upon the Corese engine. This framework provides such functionalities as generation of interfaces for requests, edition and navigation, and management of the transverse functions of a portal (presentation, internationalization, security ...). An ontology editor, a generic annotation editor, and a basic rule editor are parts of the Sewese platform.

Being built on top of Corese, Sewese uses the same license, CeCILL-C.

ECCO

ECCO is an environment for cooperative ontology development. It implements the ontology development methodology defined in Palette and uses building blocks of the Palette KM services, such as knowledge creation, annotation, retrieval, and visualization.

The final license for ECCO has not been formally set yet, but it is expected to be CeCILL-C.

SweetWiki

SweetWiki is a new wiki engine written in Java. It has been developed around the semantic web technologies.

The final license for SweetWiki has not been formally set yet, but it is expected to be CeCILL-C.

Amaya

Amaya is a Web editor, i.e. a tool used to create and update documents directly on the Web. Browsing features are seamlessly integrated with the editing and remote access features in a uniform environment. This follows the original vision of the Web as a space for collaboration and not just a one-way publishing medium.

The development of Amaya started as a joint INRIA-W3C project. It is therefore distributed under the W3C Software Notice and License. The specific developments made for Palette are a templating mechanism and a new user-interface suited for Communities of Practice.

LimSee3

LimSee3 is a multimedia authoring tool based on extensive use of templates and adaptable user interface.

LimSee3 uses the CeCILL-B license.

3.3 CTI

Policy: CTI does not impose or recommend any particular policy for licenses.

Cope-it!

CoPe_it! is a web-based system that attempts to assist and augment collaboration being held among members of Communities of Practice by facilitating the creation, leveraging and utilization of the relevant knowledge. The system follows an argumentative reasoning approach, which complies with collaborative principles and practices.

No license is currently decided, but everybody will be able to access the source code developed by CTI through a CVS base.

The following software products are needed for installing and running Cope_it!, but are not distributed by PALETTE:

- Internet Information Server (IIS):
  Purpose: Web server hosting CoPe_it! Web application
  Licensing: proprietary (comes with Windows Operating systems. For non-server versions, the number of sessions is limited to 10).
- Microsoft .NET Framework 3.0. Needed for executing the CoPe_it! Web application.
  Purpose: run/execute application logic (written in C#)
  Licensing: free download, install and deployment of the framework for commercial or non commercial uses.
- Microsoft SQL Server 2005
  Purpose: database back-end.
  Licensing: commercial with various licensing options (per CPU etc).

Installation of Microsoft SQL Server 2005 requires clients to purchase licenses.
For more details see: http://www.microsoft.com/sql/howtobuy/default.mspx
http://www.microsoft.com/sql/howtobuy/faq.mspx
However, the SQL Server 2005 Runtime permits redistribution of SQL Server 2005 along with CoPe_it! to clients without obligating the client to purchase licenses. In this case, the SQL Server 2005 Runtime is permitted to be used only by the application (CoPe_it!). The client of the application cannot use this SQL Server product to run other applications or to develop new applications, databases, or tables. The vendor (CTI is vendor for the CoPe_it! current installation) must have a Product Integration Agreement so that he is able to redistribute the SQL Server 2005 Runtime. For more information see:
http://support.microsoft.com/kb/917400

- TIBCO General Interface v3.4.1
  Purpose: javascript library
  (http://www.tibco.com/software/rich_internet_application/general_interface/default.jsp)
  Licensing: Open source BSD License
- Script.aculo.us v1.7.0
  Purpose: javascript library, http://script.aculo.us/
  Licensing: free software under MIT license. Allows reuse within proprietary applications as well as in free and open source software:
  http://wiki.script.aculo.us/scriptaculous/show/License

3.4 CRP Henri Tudor

Policy: CRP Henri Tudor does not impose or recommend any particular policy for licenses.

Generis

Generis is a knowledge management tool working as a web platform treating knowledge as information within some context. It is an ontology server able to work in a distributed way. Generis enables collaborative creation, edition and management of models representing the concepts of a particular domain as well as relations between these concepts and annotations of web resources according to these concepts and relations.

Generis includes GPL licenced components. Hence, its final license will be GPL version 2 (excluding higher versions).

Generis current package includes MySQL and Apache server which is not GPL, but this will have a minor impact since the distribution can be dissociated. In addition, Generis can use other RDBMS (Oracle was already tested) and web servers.

There is however one current restriction regarding the distribution of the source code of Generis. The current release is a beta release not for distribution. Generis has been developed in the framework of another national project aiming at developing a computer-based assessment platform called TAO. So far, all partners using TAO (and thus its Generis kernel) must sign a collaboration agreement including non-disclosure clauses preventing them from redistributing the code under development that are valid only on the beta release (there is an automatic termination of the non-disclosure clause when public release in issued). During the Palette project, there might be no problem since CRP Henri Tudor can host the application. The question may arise after the project when the future distribution of the software will be discussed, in respect with the status of the release at that time.

Other tools

Other tools will be developed by CRP Henri Tudor independently from Generis (knowledge evaluation service and SoA-related components in WP5).

No decision has been taken yet for licensing these tools. It is however assumed that GPL version 2 (excluding higher versions) will be adequate too.

4 – Collaboration with the QualiPSo European Project[5]

From the above internal review of the PALETTE services, it appears that different licenses are currently used and all of them are compatible with GPL, if not GPL itself. Some tools do not have a clear license yet, but the first impression is that most services could adopt one of the licenses already in use in PALETTE.

Despite this first report, the way to determine conditions under which PALETTE software may be distributed is still unclear and needs some refinements. After some discussions, the Consortium decided to search for European projects faced with the same questions and decided to get in touch with the QualiPSo project.

The goal of the QualiPSo IP (Quality Platform for Open Source Software) is to help industries and governments fuel innovation and competitiveness with Open Source software. To meet that goal, the QualiPSo consortium intends to define and implement the technologies, processes and policies to facilitate the development and use of Open Source software components, with the same level of trust traditionally offered by proprietary software.

Contacts started between ERCIM representatives and INRIA. Two meetings took place in Sophia Antipolis and it was finally decided that the PALETTE project will act as a beta user of the QualiPSo methodology, by answering a second questionnaire prepared and discussed during these two meetings (Annex 1).

5 – Conclusion

The PALETTE “Services Mediators[6]” will fill in the QualiPso questionnaire in November 2007. Collected answers will be compiled by ERCIM and the results jointly analysed by PALETTE and QualiPSo representatives.

At the end of this process, we expect QualiPSo to provide advises regarding the Open Source Strategies for the set of services offered by the PALETTE project. The consortium will then review all possibilities in the light of the exploitation/valorisation plan and will validate one of them.

[5] QualiPSo Fact Sheet available at http://cordis.europa.eu/fetch?CALLER=PROJ_IST&ACTION=D&DOC=19&CAT=PROJ&QUERY=1187189361220&RCN=80465

[6] The Service Mediator is the person in charge of a service, being the only interlocutor for this service within the project

ANNEX

QualiPSo questionnaire for PALETTE project

Collecting information for auditing code

The goal of this document is to describe information to be collected in order to conduct the first step of an investigation (audit) to determine the conditions upon which a given software can be distributed.

Context

The situation we consider is rather general: you (an organization or individual) wish to make available to third parties (to distribute) a software package. This package may just include an executable program or it may contain its source code, possibly together with the source code of other needed software, and various documents.

The aim is to determine the conditions under which the software package may be distributed i.e. its possible licenses (if any). Especially in the case where it uses Open Source components. Determining these conditions may be quite complicated. As a first step it is necessary to collect some information. This information can be split into four categories:

- on external components and tools
- on your code
- on included literary and artistic works
- on peripheral IPR (Intellectual Property Rights)

The same information is needed even if the software is not really distributed but made available through a server.

1. Information on external components and tools

A rough definition of external components and tools could be: everything needed by your code at runtime or compile time. It may include libraries, servers, clients, frameworks, compilers and their runtime libraries, development environments, code generation tools, etc.

Information to be gathered is at least the name (and version) of each component or tool, a brief description of the way it is used, the license under which you got it (for commercial products it may be necessary to have the exact license text or contract as they may not be available under a standard, unique license), whether it will be distributed with your code or not (either in source or binary form, possibly statically linked into a program) and whether you modified it or not.

2. Information on your code

Borrowing information

Your code may include code you borrowed from someone else (outside your organization). You may have modified or translated it (from another programming language), or just extracted some piece of it. This code should be identified together with its license if any and the kind of transformation you made (extraction, modification, translation...).

Authorship information

We will define author as any individual who wrote some part of the code, or some part of a specification or preparatory material (for example, things like precise or non trivial description of software architecture). An author may not have really written code but may have directed another person to write. An author may have none, one or more affiliation organizations (organizations that pay it or direct it).

The needed information is the list of authors, together with several elements:

- whether they write code or specification or preparatory material (and in the latter case what was it)
- their affiliation organizations
- what kind of rights the affiliation organizations have and on what grounds (work contract, national law, special status...)

This information is necessary also for the authors of the possible modifications to external components or tools.

If you have borrowed (reused, modified, translated) part of previously existing code in your organization the authors or this pre-existing code should also be listed.

Contractual context

The information needed is the set of contracts (or grants) under which part of the code (including previous version of the code) or modifications to external components or tools may have been written. Of interest are the IPR, confidentiality, publication or exclusivity clauses in these contracts.

As these contracts may be confidential, you can at least provide a list of their types and numbers (for example one contract with an industrial partner, one European contract and two state grants).

The information on previous distributions of (part of) your code (including previous versions) or modifications to external components or tools is also needed. It may be Open Source or ad-hoc licenses or contracts.

3. Information on literary and artistic works

The same kind of information (as in the previous section) is needed for all the artistic and literary works included in the software package to be distributed. This includes documents such as user's manual or technical notes, logos, icons, images, drawings, animations... whatever their formats are. Also note that individual elements inside documents may need to be identified (for example a picture or drawing inside a text document at least when its author is not one of the authors of the document).

4. Information on peripheral IPR

On patent: a list of the patents (pending or granted) that to your current knowledge may cover your code.

On trademarks: a list of the names that you may use for your code or in connection with your code.