Virtual Physiological Human Network of Excellence

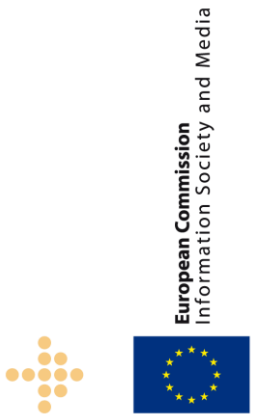
Grant Agreement: 223920

VPH ToolKit Guideline Document

Topic: Tool Characterisation

Version 1.0

22-Mar-11



This page is intentionally blank

Document Information

IST Project Num	FP7 – 2007 - ICT - 223920	**Acronym**	VPH NoE
Full title	Virtual Physiological Human Network of Excellence		
Project URL	http://www.vph-noe.eu		

Document	**Number**	G01	**Title**	Guidance (Tool Characteristic)

Status	Version. 1.0	Final ☑
Dissemination Level	Public ☑ Consortium ☐	

Authors (Partner)	UCL	Haidar, Nash		
	UOXF	Cooper, McCormack		
	CNRS	Thomas, Friboulet, Cervenansky		
	INRIA	Sermesant, Bleuze		
	UPF	Martelli, Omedas		
Responsible Author	Cervenansky		**Email**	Frederic.cervenansky@creatis.insa-lyon.fr
	Partner	CNRS	**Phone**	+33 (0)4 72 43 87 85

Abstract (for dissemination)	This document provides guidance on the attributes required of VPH NoE ToolKit content.

Version Log

Issue Date	Version	Author	Change
30-June-10	0.1	WP3 – Frederic Cervenansky – Denis Friboulet	First version
16-Jul-10	0.2	WP3 – Frederic Cervenansky – Denis Friboulet	Second version with merged comments
30-Jul-10	0.3	WP3 – Frederic Cervenansky – Denis Friboulet	Full draft for internal review
26-Jan-11	0.4	WP3- Frederic Cervenansky – Denis Friboulet	New version with some improvements
22-Mar-11	1.0	WP3	First public release

This page is intentionally blank

Table of Contents

This page is intentionally blank

EXECUTIVE SUMMARY

This guideline is intended to assist potential contributors to the VPH NoE ToolKit in preparing their content for submission.

There are several such guideline documents in this series, covering the full range of issues affecting content providers. They are being developed over a period of time and, once finalised, these guideline may be bound together into a single VPH NoE resource.

Introduction

This document is one of a series that together build to form a complete guide to the ideal content and presentation of materials offered for distribution via the Virtual Physiological Human Network of Excellence ToolKit Portal. The full set of Guideline Documents is summarised below.

Guidance Area	**Description**
Tool characterisation	The attributes important for inclusion in the documentation of Tools, including performance validation
Model characterisation	The attributes important for inclusion in the documentation of Models, including performance validation
Data Characterisation	The attributes important for inclusion in the documentation of Data
Ontological Annotation	The significance, benefits and methods of ontological annotation of ToolKit content
Interoperability	Key attributes concerning the additional specification of predominantly tools and models that will allow operation in a multistage workflow alongside other items of ToolKit content
Ethico-legal issues, provenance	The inherited responsibilities that are attached to any item of ToolKit content – perhaps particularly data – including legal, ethical and territorial restrictions
Licensing	The conditions that apply to the legitimate use of the content from a commercial and intellectual property standpoint
Usability & Training	The factors that are important for the easy use and ready acceptance of ToolKit content, taking into account the environment, the likely users and the need for interoperability. Additionally, the nature of training facilities of all types appropriate to particular content categories.

Tool Characterisation Topic

"*Defense of conviviality is possible only if undertaken by the people with tools they control*". **Ivan Illich**.

Many tools are or will be available through VPH toolkit portal, coming from VPH Initiatives (VPH-I projects, STREPs, Exemplar Projects) or coming from outside projects (well-known software in their application domain or exposed by VPH members).

Beyond the inventory of these tools, an important feature of the Toolkit will be its ability to guide the future users by providing them with a way to choose and evaluate *a priori* the tools exposed in the Toolkit.

In this context, the purpose of this version document is mainly to gather a set of criteria to characterize tools as a function of their application domain and to provide preliminary tracks to use these criteria for assessing tools quality. This version consists, also, in trying to extract from these criteria the most relevant and reliable to put forward high quality contents to VPH Toolkit. The final version of the document will provide some prospective requirements to provide relevant tool characterization ontology.

This document will thus not provide a ranking of tools but a grid allowing ranking

This document is composed of three sections:

- ✓ The first section will expose the meaning, in VPH NOE context, of a "tool". This definition will allow specifying the scope of characterization and its limitations.
- ✓ The second part will describe, in details, the selected items to identify a tool. This description will be, first, general and specific for each, if necessary, VPH domain of expertise: data, fusion, images, models, simulation, ontologies, workflows.
- ✓ The last section will describe different keys to test the tools, document and maintain alive their characterization.
- ✓ An annex is associate to the document illustrating the selected criteria on three relevant tools: slicer3D[1] -useful medical image processing software-, tools coming from ARCH[2] VPH-Initiative project and AHE[3] grid middleware.

[1] www.slicer.org
[2] http://www.vph-arch.eu/
[3] http://www.realitygrid.org/AHE/

Underlying Concepts

Even in the restricted field of VPH areas, the term “tool” is still very loose. The term "tool" used in the present document will concern exclusively functional tools. In order to ensure easy access and dissemination, we moreover propose to consider only software that provides free[4] content (at least for research and non-commercial use).

Beyond this restrictive definition, it is to be noted that the VPH NoE has put forward a number of conceptual tools (models, ontologies, multiscale, etc.). These conceptual tools are compiled and defined in the other section of the VPH ToolKit Guideline Document. Clearly the software tools allowing the implementation and use of these concepts will have to be cross-referenced5.

The characterization of a tool can overlap different definitions such as its components, its functionality, its own definition or its place in a classification, a global reference. Each aspect shall be investigated and drived through different paragraphs.
The purpose of this document is to provide a set of criteria to describe and evaluate functional tools, to allow a guided ranking. The quality of these criteria and their conceivable values will be discussed through the associated guidelines.

For all VPH categories, the available functional tools are API, code, command line application, development framework, executable, library, online application, rich internet applications, script, web service, website and combination of tools.

[4] : If, in order to achieve sustainability, tools emerging from VPH community (VPH-I projects or Exemplar Projects) should be commercialised (as additional libraries for commercial solutions or as stand-alone application), the “free” criteria will be rethought.
[5] : A simple example of this association is ontology as a conceptual tool and Protégé as a software allowing the description and test of an ontology.

Interactions and Dependencies

As previously mentioned, the dependencies between this document and the others section of the Guidelines document have to be carefully cross-referenced: most fields of the VPH can usually be associated with a set of tools:

- For **data**, this document will expose the input-output formats and dimensions for each functional, as documented in the Data Guidelines document. The Data Guidelines document should in turn link a given format to the tools allowing its importation/exportation.

- In the same way, **Models & Ontologies** will be exposed in their associated Guidelines and provide links to the tools allowing their implementation.

- **Interoperability** is a key feature since it should allow building workflows through tools interconnection. In the same way, licensing has to be carefully evaluated since it conditions free use and distribution of the tools. These two items are characterized in their respective section of the Guidelines document, and their categories should be included as features of a tool exposed in the toolkit.

- The **standards** relevant to each VPH category will be described in the other GuideLines items.

Applicable Legislation

Different issues can arise according to the type of tools exposed. These issues can concern:

- ✓ Licence compatibility. The copyleft mechanism, such as described in licence Guidelines, can put some restrictions to the use and diffusion of a tool
- ✓ Data confidentiality. Issues around access to private data, anonymization mechanism, security of data. These are discussed in the Ethico-legal Guidelines document.

Standards and Standards Bodies

Given their broad range of application, the tools exposed in the Toolkit rely on many official (e.g ISO, IEEE, etc.) or de facto standards. The standards used for a given tool will be thus a natural part of its characterisation as described in the core of this document ("Characteristics" section). The main categories[6] or standards are the following:

- **Input/output formats:** The different formats available for the tools are documented in the other GuideLines items
- **Language**: No language is privileged. All languages have their advantages and their limitations. But the restrictions that can emerge are discussed in the Interoperability Guidelines.
- **Operating systems**: Linux, Windows, MacOS, the three major operating systems have each their own advantages and limitations. Nevertheless, a tool enable to run on the different platform ensures a larger diffusion. Cmake[7] and Autotools[8] are two tools (or suite of programming tools for the second) to assisting tools to be portable. .
- **Third parties library**: This short list is the most frequent and open-source libraries used in the field of medical imaging processing. As defined in the Interoperability guidelines, sharing the same libraries can enhance interoperability aspects:
  - *Medical images processing library*: ITK, VTK, MITK, IGSTK, VMTK,
  - *DICOM library*: dcmtk, dicom#, dcm4che, gdcm 1.3, gdcm2, PixelMed Java Dicom Toolkit,
  - *Visualisation library*. Qt, Wxwidget,
  - *Installers:* NSIS[9], InnoSetup[10] or Wix[11] allow to customize very precisely its own installation packages.
  - *Other library*. boost: collection of libraries to extend C++ functionalities, SQLite: management of SQL database.
- **Documentation**: The documentation of a tool is an essential key point for its

[6] A closed list could not be ever reached (new developments, new standards will emerge) but each new iteration of this document will refine these categories.
[7] http://www.cmake.org/
[8] http://airs.com/ian/configure/
[9] http://nsis.sourceforge.net/Main_Page
[10] http://www.jrsoftware.org/isinfo.php
[11] http://wix.codeplex.com/ only for Windows OS.

use, support and diffusion. Doxygen[12] is a powerful documentation generator for programming languages (generally associated to graphviz[13] to generate diagrams), A wiki[14], usually used for collaborative works, is a website allowing to the creation and the editing of webpages.

- **Maintenance, versioning**: CVS[15], SVN[16], GIT[17] are three revision control system to allow the management of changes in programs files. A software forge is a collaboration platform allowing collaborative software development (*wikipedia*). It also provides source code management system, mailing-lists or forums, wikis, software versioning, bug tracking.The most known and open-source are GForge[18], KForge[19] and SourceForge[20] (web-based forge).
- **Licence**: The different licences available for the tools are described through Licence Guidelines
- **Certification**: Although the certification of a tool is a long process and need to be investigated in the early time of tool development, a short indication (*from Wikipedia*) of the constraints and the standards associated to this certification can be mentioned:
    - For EU, the directive 2007/47/ec defines a medical device as "*any instrument, apparatus, appliance,* software*, material or other article, whether used alone or in combination, including the* software *intended by its manufacturer to be used specifically for diagnostic and/or therapeutic purposes and necessary for its proper application, intended by the manufacturer to be used for human beings*." The ISO standards for medical devices are covered by ICS 11.100.20[21] and 11.040.01[22]. The quality and risk management regarding the topic for regulatory purposes is convened by ISO 13485 and ISO 14971. Further standard is IEC 62304[23] for medical software.
    - For FDA, the two principal standards are the 510k[24] (Premarket

[12] http://www.stack.nl/~dimitri/doxygen/
[13] http://graphviz.org/
[14] A list and a comparison of different is available here : http://www.wikimatrix.org/.
[15] http://cvs.nongnu.org/
[16] http://subversion.apache.org/
[17] http://git-scm.com/
[18] http://gforge.org/gf/
[19] http://www.kforgeproject.com/
[20] http://sourceforge.net/
[21] http://www.iso.org/iso/products/standards/catalogue_ics_browse.htm?ICS1=11&ICS2=100&ICS3=20&
[22] http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_ics_browse.htm?ICS1=11&ICS2=040
[23] http://webstore.iec.ch/preview/info_iec62304%7Bed1.0%7Den_d.pdf
[24] http://www.fda.gov/cdrh/510khome.html

Notification) and the PMA[25] (Premarket Approval).

More generally, the standards relevant to each VPH category will be described in the other GuideLines items.

Beyond these classical categories, the Toolkit should encourage good practice in software development by putting forward software tools documented through standard development documentation[26] (for classic process[27]) to avoid tools without any documentation:

- ✓ The software requirements specification (**SRS**) is one of them. The SRS is a complete description of the behavior of the system to be developed.
- ✓ The software development design (**SDD**) is used to specify system architecture and application design in a software related project. A design specification provides in-depth detail about the functional and non-functional design requirements including assumptions, constraints, performance, dimensions, weight, reliability and standards. The design specification result should be consistently reproducible providing the same standard outcome.
- ✓ The Verification and Validation (**V&V**) document is the process document to check that a software system meets specifications and that it fulfills its intended purpose. It can be a part of more structured documentation for testing: software test documentation (**STD**).

Furthermore, some results can be associated to these documentations, especially if the results are produced with a unit test mechanism and exposed ultimately through a dashboard[28].Top quality content must be thoroughly tested, both with 100% coverage at the unit test level, but also with high-level functional or acceptance tests. For tools which implement a model, validation against data is also required (link with model guideline document). More generally, algorithms should be tested for correct operation

25

http://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/HowtoMarketYourDevice/PremarketSubmissions/PremarketApprovalPMA/ucm143067.htm

26 All standard documents are referenced at the end of the guideline.

27 With Agile software development methodology, such documents are neither produced nor useful. But some documents should be provided to clarify use-cases and desired functionality (architecture, sequences or models diagrams, backlog documents).

28 The most known is CDash(http://www.cdash.org/) from Kitware company.

Characteristics

This is the core section of the document.
This section will expose the various criteria chosen to characterize the tools. Each criterion must be thought (necessary, available and useful) for each domain (images, ontology,...) and for global view. This part will be used to clarify some concepts for specific domains (function for images domain for example).

Tool information

Name: application name. This field shall be enough clear to identify a tool.
Version: Product version of the tool. This element allows to track evolution of a tool. Significant changes shall be written in a changeLog file with the corresponding version. The changelog is used to track bugs correction, features implementation.
Function: Action(s) performed by the tool. This criterion shall allow to identify all actions available through the tool. See next paragraph for more information.
Speciality: it's a generic term to represent the domain of application, functionality not standard. This criterion shall be specified for each sub-categories. See next paragraph for more information.
Input – Output Data: this criterion shall expose all data and their formats accepted and produced by the tools.
Input – Output Data Dimension: the Dimension term shall be considerate as a focus on the level target (Molecule, cell, organ, ...).
Licence: Type of Licence, Open-source type (GNU GPL, CeCILL- B) or proprietary software.
Certification: it refers to an official agreement provided by an external organism. This agreement concerns certain standards, characteristics present in the tool. This certification could be thought as quality assurance.

Tool Specification

Language: Native Programming Language, compiler version.
OS: The knowledge of the OS where the tool is available is a key feature for the characterization. The three majors OS (Windows, Linux, MacOs)
Installation recommendation: How to install the tool with required library, drivers and

software. This part shall be an important section of software documentation. If a bootstrapping technique (through a setup executable) is needed to update some components, the user shall be aware of.

Third party Libraries: The third party libraries to run the tool. If tool needs some additional libraries or software it shall be specified.

Type of tool: the list of functional tools type referenced by VPH NOE Toolkit will be not infinite. For images subsection, only three categories will be allowed: softwares, libraries and development environment.

Type of computation: The criterion shall indicate what kind of computation (if the tool provided this functionality) is available through/ supported by the tool: CPU, GPU, DCI (Distributed Computing Infrastructures), HPC (High Performance Computing).

Tool Description

Short purpose: Some text to quickly explain the tool purpose. This criterion shall be the *abstract* of the tool.

Documentation Link: It refers to any type of documentation: code documentation (Doxygen), tutorials (getting started guide), course, wiki, help system (Frequently asked Question), archives (from mailing list, ...).

Keywords: keywords which can be associated to the tool. The keywords are a good way to summarize tools functionalities and application domains. In first instance, a free-form for keywords can be applied. For specific domains, where the knowledge is clearly identified, a fixed vocabulary, based if available on an ontology, can be proposed.

Citation & Reference papers: It refers to two sub-items: the first for papers about the tool itself and the second for papers which use the tool.

Snapshot(s): It refers to pictures, if available, allowing a better comprehension of the tool.

Long purpose: A long paper or document to explain tool behaviour and available actions.

Testing: It shall expose the different tests performed on the tool (if available or performed), such as low-level tests (unit tests) or high-level tests (functionality). The tests should also expose the proceeding to validate models or results. In parallel, the data or information used to perform the tests should be available.

Download links: It refers to a link where users will be able to download the tool, the documentation and the required libraries. A reference repository shall be proposed to allow certain sustainability.

Tool Context

People involvement: single person, team,

Author(s): It refers to academic or professional aspect of development.

Support: A crucial point of open-source software is the compilation on various computers in various environments. This requires an important level of support.

How many people involved: This criterion shall be used to identify the task force behind the tool and ensure a sufficient level of support to overcome any crucial point (compilation in various environments on various computers for example).

Reactivity: It refers to the reactivity to resolve issues if a "helpdesk" is available or through a traceability system[29].

Type of collaboration: This criterion shall expose the type of collaboration (European project, International Initiative) where tools are coming from.

Funding status: This criterion shall expose the development plan for the future of the tool, in order to reach long term sustainability.

Institute/organization: It refers to the institute, the community behind the tool development (VPH-I project for example).

End-users target: This criterion will expose the final users of the tool (software developers, researchers, clinicians or industrials users).

Development plan: It refers to future improvements, release to implement new features and to ensure the sustainability of the tool.

Website: It refers to the team, project, tool site where additional documentation and information are available.

Use-case: This criterion will refer to an use-case describing tool functionalities and uses.

Training & course: This criterion will refer to any document that can be used to train users.

Rights: Associate to Licence status, the question of tool rights (to use, to promote, to upload, and to modify) should be addressed to avoid any future issues.

Tool Functionality & Speciality

These two items need a special focus as they actually the entry points to several ontologies. Each ontology has been constructed for a specific purpose.

Biomedical Resource Ontology[30] is a controlled terminology for the resource types, area of research, and activity and which are used to improve the sensitivity and specificity of web

[29] As Mantis - http://www.mantisbt.org/ - proposes. The main benefit of a bug-tracking system is to provide a centralized overview of development requests (including both bugs and improvement) and their status.

[30] http://bioportal.bioontology.org/visualize/43043/?conceptid=BRO:Data_Acquisition_Software

searches.

BIRNLEX (ie the BIRN Project lexicon) provides entities for data and database annotation for the BIRN project, covering anatomy, disease, data collection, project management and experimental design. The project has evolved into Neurolex.

NEUROLEX[31] (*NIFSTD version 1.9.5*) is being constructed to help improve the way that neuroscientists communicate about their data, so that information systems like the NIF can find data more easily and provide more powerful means of integrating data that occur across distributed resources. The metadata structure is simpler than other ontology to avoid problems associated to the inappropriate use of metadata fields and to the intensive time consuming of both the curation effort and the training of curators.

Software Ontology[32] (SWO) has the scope of describing types of software used in Bioinformatics. The SWO covers areas such as the software type, the manufacturer of the software, the input and output data types and the uses (i.e. objectives) the software can be put to.

SW Tool Ontology[33] is the ontology used for the characterization of medical images tools in neuroscience community.

OntoNeuro[34] designs of a common semantic model providing a unified view on all data and tools to be shared between NeuroLOG project partners, focusing on neurology images processing.

Different issues can arise from these ontologies:

- The inconsistencies in annotations can make searching for resources a very difficult task, all the more if the structure is complicated and not intuitive to the end user.
- The resources need to be well defined because trying to catalog every resource useful to VPH community can be a tedious task if a resource is too narrow, such as a plug-in. For example, it will take time for a curator to annotate software library with hundreds of software applications. However if each plug-in to each program should be considered as a resource, the task will become too large and be not likely to help user.

[31] http://neurolex.org/wiki/Resources_With_Definitions
[32] http://bioportal.bioontology.org/visualize/45014
[33] http://www.cma.mgh.harvard.edu/iatr/vocabulary.php
[34] http://neurolog.polytech.unice.fr/doku.php?id=public_namespace:ontology

Tool Usability

As usability is a key point to ensure a large diffusion and use of a tool, the criteria associated to usability are discussed on their own guidelines.

Methods of Verification

The characterization of tools cannot support accurate verifications for all tools meanwhile the verification performed as certification criteria by official organism (EU, FDA) or industrial norms shall be mentioned.

But even if the verification is limited, supplier should provide use-cases to illustrate the functionalities[35]. Verification will be performed with the community who will provide examples to estimate the pertinence of described functions. In particularly, if some building community, who provides helps and strong examples for a tool, is available[36], the links with this community should be exposed.

[35] comprehensive tests used for the tool verification can be provided as use-cases
[36] Such as Matlab exchange (http://www.mathworks.com/matlabcentral/fileexchange/)

Ownership

This document exposes a set of criteria allowing to identify and characterize functional tools in the VPH NOE community context. In this context, it seems that it should be no restriction in terms of authorship.

Training

The training for a tool is strongly linked to the tool community. Some workshops, for specific tools, already exist based on regular meetings. These workshops explore tools usability, development and intensive bug tracking for example (cf. ITK workshop[37]).

As training is an efficient way of growing a community, it would be good practice that the new tools emerging from VPH should be associated to a training policy.

[37] http://www.na-mic.org/Wiki/index.php/2010_Summer_Project_Week

Maintenance

The standards tools are based upon are constantly evolving and some area need well established standards to promote interoperability. Examples are workflow language/environment where initiatives such as the Interoperable Workflow Intermediate Representation (IWIR) are on the way. In the same way initiatives exists toward interoperability of image processing development environments, such as CommonTK.

Thus, maintenance will be needed for the characterisation of tools where these new standards are emerging.

Ranking

A homogenous ranking cannot be proposed with the large amount of tools available and the infinite combination of criteria to identify them. Meanwhile, some features (**sustainability, usability)** are the key elements for the strong diffusion of a tool. These features are directly linked to the presence or not of some components:

- ✓ **Documentation**. A strong documentation is an important element to ensure a large diffusion and use. Evaluate accurately the quality and quantity (relevant, all functionalities are described, amount of, tutorials ...) of documentation is not an easy task but propose a simple metric of a rough evaluation could be the first steps for a global ranking.
- ✓ **Updates**. Frequents updates to correct issues and implement new features to keep tool alive. This could be demonstrated by a history of release dates[38], or through a link to an issue tracker, coupled with plans for future releases. The existence of such a system should be highly advised.
- ✓ **Community**. As previously discussed, this term is very general. But mailing lists, forums, FAQ webpage associated to a tool are the first step to create a group of users and experts around this tool (such facilities will be available through VPH portal too). This set of elements will provide the sustainability of the tool.
- ✓ **Open-source**. Open-source is not the bailsman for the reliability, sustainability and the future changes for a tool. But, allow to the community to participate to its development, to track bugs and propose patches to resolve them is a good way to allow a large diffusion.
- ✓ **Support (Funding)**. To elaborate a long term development, some investments are needed, both monetary and community good will. The informations (how many, permanent positions, devoted time ...) about people involved in the development and the component (no more updates ...) are a good indicator for a strong sustainability.
- ✓ **Use-case**. Associated to a tool, some use-cases will be a good estimation for the sustainability of the tool.
- ✓ **Training**. Provide some training course will smooth the first contacts with the tool.
- ✓ **Vox populi.** But, for each tool, the ultimate criterion is the end-users and their belief in a tool.

[38] As provided by the versioning tools.

For the tools that fulfil these components with high quality inputs, a VPH approval status should be proposed. With this feature, a tool will be clearly identified with highest ranking and providing a high reliability. With this approval, VPH NOE will have **the opportunity to put the focus on tools emerging from its own community.**

For each tool, a simple graphic can allow to visualize the ranking:

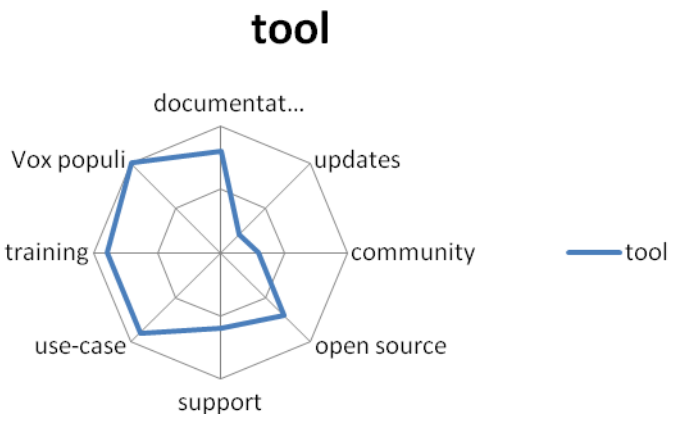


Figure 1 : Tools ranking

Documentation, Reporting Templates

For software and especially in open source community, it is now expected to provide documentation about tool's code generated with Doxygen[39] application.

As previously cited, some development documents should be available too.

For most of the criteria described above guidance will be provided to indicate the level of information wanted.

[39] *www.**doxygen**.org*

Others Initiatives

Different initiatives are providing similar classification with a more or less developed tools characterization in function of the application domain.

IATR[40] provides a centrally available listing of all image analysis tools that are available to the neuroscience community in order to facilitate the development, identification, and sharing of tools that are of use to the general community.

iTools[41] provides an infrastructure for classification, categorization and integration of different computational biology resources across space-and-time scales, biomedical problems, computational infrastructures and mathematical foundations. Three types of resources can be referenced through iTools – data, software tools and web-services.

IdoImaging[42] is a rough searchable database of free applications, with details of platforms, authors, and version, dedicated to medical images. More than 250 programs are referenced.

NITRC[43] facilitates finding and comparing neuroimaging resources for functional and structural neuroimaging analyses with the possibility to add use-cases and comments on a tool.

Medfloss[44] provides a comprehensive and structured overview of Free/Libre and Open Source Software (FLOSS) projects in the domains of medical informatics and health care. Moreover it offers an open content platform to foster the exchange of ideas, knowledge and experiences about these projects.

Plume[45] is a French project created to Promote economicaL, Useful and Maintained software for higher education and research communities. The site references software, mainly Free/Libre Open Source Software (FLOSS) from French universities and national research organisations, (CNRS, INRA)

i2b2[46] (Informatics for Integrating Biology and the Bedside) contains a large amount of biomedical software resources. More, it is developing a scalable computational framework to address the bottleneck limiting the translation of genomic findings and hypotheses in model systems relevant to human health.

[40] http://www.cma.mgh.harvard.edu/iatr/about.php?page=introduction
[41] http://cms.loni.ucla.edu/iTools/
[42] http://www.idoimaging.com
[43] http://www.nitrc.org/
[44] http://www.medfloss.org
[45] http://www.projet-plume.org/
[46] https://www.i2b2.org/

Further Information

Readers seeking more details should be able to find sources listed and explained here.

For good development practices, please refer to SRS, SDD and Agile documentations already cited.

An article from the British Computer Society for the good practice in software developments:

http://www.bcs.org/upload/pdf/cop.pdf

References

Standard Development documents:

- ✓ SRS:http://standards.ieee.org/reading/ieee/std_public/description/se/830-1998_desc.html
- ✓ SDD: http://standards.ieee.org/reading/ieee/std_public/new_desc/se/1016-1998.html
- ✓ STD:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4578383&isnumber=4578382
- ✓ V&V:http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1488512
- ✓ Agile software development methodology : http://www.agilealliance.org/

Some developments in VPH NOE already following these standards:

- ✓ DCM API SRS:http://www.vph-noe.eu/vph-repository/doc_download/140-dcm-api-specifications
- ✓ DCM API SDD: http://www.vph-noe.eu/vph-repository/doc_download/165-dcm-api-design-document

Preliminary documents, still in VPH NOE framework, have previously tried to characterize tools for the domain of medical imaging:

- ✓ **GUIDE SRS** : http://www.vph-noe.eu/vph-repository/doc_download/142-guide-specifications
- ✓ **VPH Technology Database Requirements :** https://vphnoewiki.comlab.ox.ac.uk/VphTechnologyPortal?action=AttachFile&do=get&target=VPH+Technology+Database+Requirements+Final+20090430.doc

Annexes

	Slicer 3D (external tools)	archTK (VPH-Initative)	AHE (grid middleware)
Tool Information			
Name	slicer or slicer 3D	archTK	Application Hosting Environment
Version	3.6	1	2
Function	Annotation - Interaction; Filtering; Mesh generation; Quantification; Registration; Segmentation; Measurements;Visualisation	Vascular network modelling (solver and end-user editor)	
Speciality	medical images processing	Hemodynamics	Access to grid resources and clusters
Input-Output Data	DICOM		
Input-Output data Dimensions	2D - 4D	0, 1	
Command line Support	Yes	Yes	yes
Graphical Interface	Yes	Yes	yes
Tasks description	Slicer is a "point and click" end-user application. Slicer is used as a vehicle for delivering algorithms to computer scientists, biomedical researchers and clinical investigators. Slicer is distributed under an open source license without a reciprocity requirement and without restrictions on use.	Solver: simulate hemodynamics in vascular networks using lumped or 1D wave propagation models, including user-defined non-linear elements. Editor: define and modify vascular networks as undirected graphs, define properties on nodes and edges, run simulations and inspect results interactively.	submit, monitor, terminate and download application execution on grid and clusters resources.
Licence	BSD licence http://www.slicer.org/pages/LicenseText	Apache 2.0	OMII open source
Certification	None	None	
Tool Specification			
Language	C++	C++ and Python	Java and Perl
OS	Windows, Linux, MAC OS	Windows, Linux, Mac OS	Windows, Linux, MAC OS
Installation Recommendation	Stable Releases: Pre-compiled stable Slicer Releases for Linux, Windows, Mac and Solaris. This is what most people will want to download. See also the release notes. Snapshots: Custom built Slicer binaries, in various states of completion, i.e. some features might not be stable. Nightly builds: This contains a week's worth of nightly builds. Nightly builds are experimental and sometimes unstable. x86 means Intel or AMD processors, Darwin is for Mac OS X, PPC means PowerPC processors. Mac: Darwin is the OpenSource software environment for Apple's Mac OS X Hardware/OS requirement: Either Windows XP or more recent, Linux (x86 or x86_64), Mac OS X (ppc or Intel), min 2 GB of RAM and a dedicated graphic accelerator with at least 128 MB of on-board graphic memory (512 or more recommended). Shared memory graphics will result in slow render speeds. X11 for Mac: On Mac OS X you will need to install X11 from the CD. As an alternative, we had good experience with xquartz. Collaboration with the University of Szeged in Hungary has resulted in a port of slicer3 to the current generation of the Oracle (formerly Sun) Solaris operating system. More information,	Installing from source: Solver: prerequisites are a recent version of Python 2.x, Numpy and Matplotlib (both Python modules). The solver can then be run directly. Editor: prerequisites are a recent version of Qt (>4.7). The source code is compiled with no extra dependencies from QtDesigner or the command line. Binary releases will be available since April 2011 for Windows and MacOS.	
Third party Library	ITK, VTK, QT	Numpy, Matplotlib, Qt	Axis 2 Web Services
Type of tool	software	Software	
Type of computation	CPU - GPU	CPU	
Tool Description			
Short purpose	Slicer is a community platform created for the purpose of subject specific image analysis and visualization. - Bidirectional interface for devices - Multi-modality imaging including, MRI, CT, US, nuclear medicine, and microscopy. - Multi organ from head to toe. - Expandable and interfaced to multiple toolkits.	archTK is an open source tool targeted towards researchers and clinicians in training for simulating hemodynamics of vascular networks using lumped and 1D wave propagation models. It is made up of two independent components, a solver written in Python and a Qt-based editor, which can be used independently from the former. The tool is flexible with respect to the underlining mathematical models of vessels and can be easily extended through the definition of new element classes. It allows the specification of non-linear relationships between physical quantities at the users' level, with no need of subclassing. Vascular networks can be generated and imported directly from the Vascular Modeling Toolkit (www.vmtk.org). Starting April 2011, the solver will be integrated with a case repository including vascular networks for a set of exemplar applications, namely upper extremities (for vascular access for hemodialysis applications), circle of Willis, coronary tree.	AHE is a user-centric light weight middleware that allows users to run complex applications (e.g. simulation, workflows) on grid resources and local clusters.
Documentation Link	http://www.slicer.org/slicerWiki/index.php/Slicer3:Developers	archtk.github.com	
Keywords			
Citation & Reference papers	Elhawary H, Liu H, Patel P, Norton I, Rigolo L, Papademetris X, Hata N, Golby AJ INTRAOPERATIVE REAL-TIME QUERYING OF WHITE MATTER TRACTS DURING FRAMELESS STEREOTACTIC NEURONAVIGATION Neurosurgery. 2010 Dec. Covington K, McCreedy ES, Chen M, Carass A, Aucoin N, Landman BA NTERFACES AND INTEGRATION OF MEDICAL IMAGE ANALYSIS FRAMEWORKS: CHALLENGES AND OPPORTUNITIES Annu ORNL Biomed Sci Eng Cent Conf. 2010 May 25;2010:1-4 Daniel Hähn INTEGRATION OF THE VASCULAR MODELING TOOLKIT IN 3D SLICER Student Research Project 2009 Apr		A. N. Haidar, S. J. Zasada, P. V. Coveney, A. E. Abdallah, P. Y. A. Ryan, , B. Beckles, J. M. Brooke, and M .A. S. Jones. "Audited Credential Delegation - A User-Centric Identity Management Solution for Computational Grid Environments". In Sixth International Conference on Information Assurance and Security (IAS), pp. 222–227. (doi: 10.1109/ISIAS.2010.5604067), IEEE Computer Society June 2010
Snapshot	http://wiki.slicer.org/slicerWiki/index.php/File:Slicer3-Architecture.jpg		Interfaced to the IMENSE system developed in ContraCancrum project and to Audited Credential Delegation Security plug in
Long purpose	http://www.slicer.org/publications/item/view/1835		
Testing	http://www.slicer.org/slicerWiki/images/5/51/Slicer3MinuteDataset.zip http://www.slicer.org/slicerWiki/images/6/61/Slicer3VisualizationDataset.zip http://www.slicer.org/slicerWiki/images/2/20/HelloWorld_Plugin.zip http://www.slicer.org/slicerWiki/index.php/File:EditorTutorialDataset.zip		
Download link	http://www.slicer.org/pages/Special:SlicerDownloads	http://archtk.github.com/download.html	

	Slicer 3D (external tools)	archTK (VPH-Initative)	AHE (grid middleware)
Tool context			
people involvement	team	Simone Manini (Mario Negri Institute), Luca Antiga, Michele Colombi (Orobix Srl - ARCH project subcontractor)	
Author(s)	Steve Pieper, Julien Fillet, Jean Christophe Robin	ARCH VPH project	Stefan Zasada and Ali N. Haidar
Support		5	
How many people involved	10	ARCH project context	2
Funding status	Major funding for Slicer was provided through a variety of federal and private funding sources, including NCRR, NIBIB, Roadmap, NCI, NSF, DOD and others.	ARCH VPH project	
Institute/organization	The National Alliance for Medical Imaging Computing (NA-MIC) www.na-mic.org	Mario Negri Institute	
Type of collaboration	American Initiative		
End users Target		Scientists, developers and clinicians.	Scientists, developers and clinicians
Website	www.slicer.org	archtk.github.com	
Development plan	Slicer is expected to evolve dynamically in architecture and implementation by drawing on the expertise and effort of the wider NA-MIC, NAC, Harvard Catalyst and NCIGT communities. The NA-MIC software engineering methodology, as applied to the problems which Slicer has historically addressed, is expected to result in a cleaner architecture that is easier for developers to support and extend.	First official release in April 2011, source code already available	Full Java version with usable security features to be released in Sept 2011
Use-case		Research on hemodynamics and wall mechanics on idealized	
Reactivity	mailing list very active: http://massmail.spl.harvard.edu/mailman/listinfo/slicer-users		
Training & course	http://www.slicer.org/slicerWiki/index.php/Slicer_3.6:Training	Not planned yet.	
Rights	no restrictions on the use of the software	Copyright regulated by the Apache 2.0 license.	
Security Tools Characterization			
Number of Credentials		0	1
Adding Users accounts		No	included
Removing Users Accounts		No	included
Setting Permissions		No	included
Acquiring Credentials		No	included
Easy to Install		Yes	instantly
Changes to Security Policy		No	none
Scalability		Yes	