Multiscale Spatiotemporal Visualisation

STREP # FP7-248032

MSV Deliverable

D6.1 - Draft non-profit business plan

Work package 6: Dissemination and exploitation

Debora Testi (B3C)

30/12/2011

SEVENTH FRAMEWORK PROGRAMME

eHealth

DOCUMENT INFORMATION

IST Project Number	FP7-248032	**Acronym**	MSV
Full title	Multiscale Spatiotemporal Visualisation: development of an open-source software library for the interactive visualisation of multiscale data		
Project URL	http://www.msv-project.eu		
Document URL	https://www.biomedtown.org/biomed_town/MSV/associates/reviewers/		
EU Project officer	Ivo LOCATELLI		

Deliverable Number	6.1	**Title**	Draft non-profit business plan
Work package Number	6	**Title**	Dissemination and exploitation

Date of delivery	**Planned**	31-12-2011	**Actual**	02/01/2012
Status	Version v2		Final ☒	
Nature	Prototype ☐ Report ☒ Demonstrator ☐ Other ☐			
Dissemination Level	Public ☐ Consortium ☒			

Authors (Partner)	Debora Testi (B3C)			
Responsible Author	Debora Testi		**Email**	d.testi@scsitaly.com
	Partner	B3C	**Phone**	+39-051-593543

Abstract (for dissemination)	The MSV project result, the MSVTK library, will be released as open-source software to the all biomedical developers community. In this document, the open source model has been analysed together with the MSVTK library potentialities and weakneses with a preliminary SWOT analysis. The report also summarises the good practices that will be put in place to make the MSVTK library a successful initiative after the end of the project. The defined exploitation strategy will be revised in one year time when the MSVTK library will be released.
Keywords	Open-source, licence, SWOT, exploitation

Version Log			
Issue Date	**Rev No.**	**Author**	**Change**
21/12/2011	V1	Debora Testi	First draft with all contents
30/12/2012	V2	Debora Testi	Final version

Project Consortium Information

Disclaimer: This document is property of the MSV Consortium. There is no warranty for the accuracy or completeness of the information, text, graphics, links or other items contained within this material. This document represents the common view of the consortium and does not necessarily reflect the view of the individual partners.

List of Abbreviations

SCS	SCS srl
BED	University of Bedfordshire
UPF	University POmpeu Fraba
KIT	Kitware
AUK	University of Auckland
SWOT	Strenghts, Weaknesses, Opportunities, and Treats
VTK	Visualisation ToolKit
MAF	Multimod Application Framework
GPL	General Public Licence

Summary

1 Introduction

Aim of this document is to provide a preliminary overview on the exploitation plans the consortium has for the project outcome, the MSVTK library.

This document will first summarise the exploitation models, and in particular focus on the open-source project characteristics.

Then, the final MSV product will be described together with the expected end-users, and the decisions already taken by the consortium associated to the exploitation scenario, like the licence model.

This report and the exploitation strategy will be revised and further detailed by the end of the project when the MSVTK library will be ready to be released.

2 Exploitation plans

Exploitation plans are usually considered as business plans, which are formal presentation of a set of business goals, the reasons why they are believed attainable, and the plans for reaching those goals including background information about the organization or team who is trying to reach those goals.

In particular, the aim of an exploitation plan should be to identify one or more “products”, their customers, and the sustainability plan for the releasing of those products.

This plan can be either commercial, thus involving for costs or revenues and a detailed financial analysis, or be, as it is more and more frequent in software development, associated to an open-source program which does not imply any direct commercial revenues by itself.

The MSVTK library will be released to the worldwide biomedical and developers community as ***open-source***; thus, the rest of the document will focus on open-source software models, licences, and sustainability aspects.

3 Open-Source model

3.1 How it works

Software sharing has been around as long as software itself. The term open source was created as alternative to "free", by the Open Source Initiative (OSI)[1], to avoid the potential confusing meaning of the word “free”. From then, the term open source describes practices in production and development that promote access to the end product's source materials. Open-source grew with the rise of the Internet, and the need for massive retooling of the computing source code.

Opening the source code enabled a self-enhancing diversity of production models, communication paths, and interactive communities. Subsequently, the new phrase "open-source software" was born to describe the environment that the new copyright, licensing, domain, and consumer issues created. In particular, open-source software refers to software whose source code is published and made available to the public, enabling anyone to copy, modify and redistribute the source code without paying royalties or fees under centain conditions. Open source software can then evolve to cooperation on the implementation into a jointly owned source code.

3.2 Economic analysis

Even if open-source does not imply any direct commercial revenue, most economists agree that open-source candidates have an information good aspect. In general, this suggests that the original work involved a great deal of time, money, and effort. However, the cost of reproducing the work is very low, so that additional users may be added at zero or near zero cost — this is referred to as the marginal cost of a product.

Being organized effectively as a consumers' cooperative, the idea of open source is then to eliminate the access costs of the consumer and the creator by reducing the restrictions of copyright[2]. This leads to creation of additional works, which builds upon previous work and adds to greater social benefit. Organizations such as Creative Commons[3] have websites where individuals can file for alternative “licenses”, or levels of restriction, for their works. Thus, on several fronts, there is an efficiency argument to be made on behalf of open-sourced goods.

Others argue that society loses through open-sourced goods: because there is a loss in monetary incentive to the creation of new goods, new products will not be created. This argument seems to apply particularly well to the business model where extensive research and development is done, e.g. pharmaceuticals. However, this argument ignores the fact that cost reduction for all concerned is perhaps an even better monetary incentive than is a price increase.

3.3 Software licences

A software license is a legal instrument (like a contract) governing the usage or redistribution of software. A software license grants an end-user permission to use one or more copies of the

[1] http://www.opensource.org/
[2] Copyright is a legal concept, enacted by most governments, giving the creator of an original work exclusive rights to it, usually for a limited time; http://en.wikipedia.org/wiki/Copyright
[3] http://creativecommons.org/

software in ways where such a use would otherwise potentially constitute copyright infringement of the software owner's exclusive rights under copyright law.

Software licenses can generally be fit into the following categories: proprietary licenses (used normally for commercial software and defined case by case by the software owner), and free and open source licenses, which include free software licenses and other open source licenses. The features that distinguish them are significant in terms of the effect they have on the end-user's rights.

Free and open source software

A primary consequence of the free software form of licensing is that acceptance of the license is essentially optional — the end-user may use the software without accepting the license. However, if the end-user wishes to exercise any of the additional rights granted by a free software license (such as the right to redistribute the software), then the end-user must accept, and be bound by, the software license.

Open source licenses generally fall under two categories:

- *Copyleft*: preserving the openness of the software itself; the most known example is the GNU General Public License (GPL). This license is aimed at giving end-users permissions to redistribute, reverse engineer, or otherwise modify the software under the terms of the license. However, these permissions are not entirely free of obligations for the end-user who must comply with certain terms if he/she wishes to exercise these extra permissions granted by the GPL. For instance, any modifications made and redistributed by the end-user must include the source code for these, and the end-user is not allowed to re-assert the removed copyright back over their derivative work. The modified software is therefore also publicly available for further modification by any user.
- *Permissive*: aims at giving freedom to the users of that software. Examples of permissive licences are the BSD and the MIT license, which essentially grant the end-users permissions to do anything they wish with the source code, including the right to take the code and use it as part of closed-source software or software released under a proprietary license.

In addition to granting rights and imposing restrictions on the use of software, software licenses typically contain provisions, which allocate liability and responsibility between the parties entering into the license agreement. In enterprise and commercial software transactions these terms (such as limitations of liability, warranties and warranty disclaimers, and indemnity if the software infringes intellectual property rights of others) are often negotiated by attorneys specialized in software licensing. The legal field has seen the growth of this specialized practice area due to unique legal issues with software licenses, and the desire of software companies to protect assets, which, if licensed improperly, could diminish their value.

3.4 Open-source projects examples

Open-source in itself is a success story as it has become in a relative low number of years a mainstream movement and it has received the attention of both individuals and businesses worldwide.

There are many successful (which means widely used and known) open source products and projects. It is out of scope to provide a complete list and review all open surce software projects; thus, here just few examples are provided.

- Linux, hand in hand with GNU software as GNU/Linux, born as an OS kernel based on Minix back in 1991. These days, a majority of web servers run Linux, and with Ubuntu it is also starting to make inroads into the desktop market, and it is now also a strong player in the mobile market with Android (which uses the Linux kernel).

- FreeBSD, OpenBSD and NetBSD derived from Berkeley Unix in the 1990s have been well-respected server OS alternatives for a long time; the core for Apple's Mac OS X is derived from FreeBSD.

- MySQL is the most widely used database server in the world, used by a huge amount of websites and services (examples include Wikipedia, and Facebook). It represents the M in the hugely popular LAMP stack (Linux, Apache, MySQL, PHP).

- The Apache HTTP Server has been the most popular web server software in the world since 1996, which is also the year it got started. Apache still has a strong lead, outclassing second runner up IIS in terms of number of deployed websites.

- Firefox 1.0 was launched in 2004 and the browser has since then taken away a huge chunk of the browser market from the previously dominant Internet Explorer.

- Since its launch in 2004, WordPress has become a dominant and hugely popular blog platform. In a survey it was repoted that 27% of the top 100 blogs ran on WordPress.

- BIND (*The Berkeley Internet Name Domain Server*) is the most widely used DNS server software on the Internet. The first version of BIND goes back to the early 1980s and has been the main DNS server on UNIX systems ever since. It can justly be called the world de facto standard DNS server.

4 MSVTK

4.1 The product

The MSV product will be a software library, called MSVTK.

MSVTK is a C++ based library developed as an extension to VTK, and it includes functions and classes to enable multiscale visualisation and interaction of biomedical data. The library will address a variety of challenges (see D3.2 for details), and in particular, it will include specific click-zoom interaction mechanisms and efficient management of out-of-core data.

The MSVTK library is highly innovative as no similar products are available either as open-source or as commercial products. Even if at the end of the project, the consortium will be able to release only an alpha version of the library, thus requiring bug fixing, futher extensiosn etc, MSVTK will already present a number of advantages with respect to the state of the art:

- It will be released with the most open license available; thus allowing anyone to adopt it in their developments;
- It will rely on state of the art software visualisation toolkit like VTK;
- It is not focused on a single use scenario, but, relying on a number of exemplary problems, it will allow to be potentially used in any biomedical domain;
- It will be general enough to be adopted with low effort also in domains other than the biomedical one;
- It will be released with a number of exemplary data sets and demonstrator applications to prove the efficacy of the proposed approach;
- The visualisation frameworks already managed by the project partners (VTK, MAF, GIMIAS) will be the first ones to adopt the MSVTK library providing thus an initial outreach community.

4.2 The end-users

Demonstrations will be developed during the project to show case the MSVTK library potentialities, but the MSV product will be the above-mentioned MSVTK library. Being a software library, end-users will be developers in the computer-aided medicine domain, which would like to enable their specific end-users applications with multiscale functionalities.

4.3 Preliminary SWOT analysis

The SWOT analysis is a strategic planning method used to evaluate the Strengths, Weaknesses, Opportunities, and Threats involved in a project or product. It involves specifying the objective of the project and identifying the internal and external factors that are favorable and unfavorable to achieve that objective.

- Strengths: characteristics of the business, or project that give it an advantage over others;

- Weaknesses (or Limitations): characteristics that place the project at a disadvantage relative to others;
- Opportunities: external chances to improve performance (e.g. make greater profits) in the environment;
- Threats: external elements in the environment that could cause trouble for the project.

Identification of SWOTs is essential in the process of planning the subsequent steps for achievement of the selected objectives. This would allow achievable goals or objectives to be set and also to deploy strategies to overcome the potential treats to the project success.

Here below the preliminary MSVTK SWOT analysis is reported based on the current status of implementation of the library. This will be revised in terms of strength and weaknesses at the end of the project, when MSVTK will be ready, before finalising the exploitation plan. The SWOT will be then completed also taking into consideraration the MSVTK community building issues described in next section.

SWOT-analysis		**Internal factors**	
		Strengths - New SW tool with respect to state of the art. - General enough to be used/adopted in other domains outside the biomedical one. - Examplary data publicly available to test the library.	**Weaknesses** - Uncomplete and still under development (alpha release).
External factors	**Opportunities** - Partner developing the library will also be early end-users - Partners are all already running open-source projects - MSVTK can be used in any biomedical domain	*The MSVTK library will be integrated in the partners open source frameworks as soon as possible; thus providing both dissemination channels and the continuing of the development after the end of the project.* *The different domains coverd by the use cases and demonstrators will allow showcasing the MSVTK potentialities to different potential end-user communities.*	*Different prototypes will be implemented to showcase the MSVTK potentialities in different biomedical domains with example data so to facilitate the uptake from external new adopters.*
	Threats - No adoption from end-users external to the project consortium	*The innovative characteristic of the library will allow to disseminate it at relevant conferences and prestigious papers; thus fighting the reluctancies of new comers.* *Special attention will be posed to community building issues relying also on the previous experiences of the MSV partners.*	*Strong dissemination strategies will be put in place involving also organisation of seminars and workshops.* *Specific dissemination and outreach activities will be put into place towards the contributior s of the CommonTk initiative (CTK[4]).*

[4] http://www.commontk.org

5 MSV exploitation strategy

5.1 Licence model

In order to have as much adopters as possible, the MSVTK library will be released as open-source under BSD-like licence, Apache 2.0 licence[5]. The license allows proprietary use, and for the software released under the license to be incorporated into proprietary products. Works based on the material may be released under a proprietary license or as closed source software.

Here below it is reported a standard version of the Apache 2.0 licence which will be associated to MSVTK.

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

[5] http://www.apache.org/licenses/LICENSE-2.0.html

Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

You must give any other recipients of the Work or Derivative Works a copy of this License; and

You must cause any modified files to carry prominent notices stating that You changed the files; and

You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

5.2 Building the MSVTK community

It is not easy to have open-source software of success. Despite the famous successful examples there are also a lot of attempts, which fails. In this section, we will report the main issues, which should be addressed to make MSVTK library a successful project.

As already mentioned, the main factor of success in an open-source project is the creation of an active community, which will imply to get both support on the development from external contributors and long-term sustainability of the library after the end of the EC-funded project.

5.2.1 Making the project inviting

The presentation and packaging of the library should be addressed very early so to reduce the barrier to entry to new comers. This involves some specific steps like writing documentation, setting up a project web site, simplify as much as possible of the software compilation and installation, etc. It is also important to clearly organise the information with a simple and effective graphic layout for the website so to guide new comers to the most relevant information.

The mission statement should be concrete, limiting, and above all, short. It has to be defined very clearly by including what the project is really about and its limitations.

Then, before anything else, it should be clearly stated the project is open source and provide information on the exact license.

There should also be a brief list of the features the software supports (including what is "planned" or "in progress"), and the kind of computing environment (both hardware and software) required to run the software.

5.2.2 Effective project management

Management in an open source project is not visible, but in the successful projects, it is usually happening behind the scenes in some form or another.

An open source project consists of a random collection of programmers, who have most likely never met each other, and who may each have different personal goals in working on the project.

The aim of management is thus to ensure that the project is coherently working, by setting standards for communications, moderate the public communication like in forums, and priotitise and define a roadmap for future developments and releases.

5.2.3 Development Status

A new comer would also be interested in knowing the project status, which is: for new projects, the gap between the project's promise and current reality, while for mature projects, how actively it is maintained, how often it puts out new releases, how responsive it is likely to be to bug reports, etc. The status should be reported in conventional ways clearly mentioning the library release.

The term *alpha* usually means a first release, with which users can get real work done and which has all the intended functionality, but which also has known bugs. The main purpose of alpha software is to generate feedback, so the developers know what to work on. The next stage, *beta*, means the software has had all the serious bugs fixed, but has not yet been tested enough to certify for release. The purpose of beta software is to either become the official release, assuming no bugs are found, or provide detailed feedback to the developers so they can reach the official release quickly.

5.2.4 Downloads

The software should be downloadable as source code in standard formats. The distribution mechanism should be as convenient, standard, and low-overhead as possible. Software should conform to standard build and installation methods; the more it deviates from the standards, the more potential users and developers will give up.

Downloading source packages is fine for those who just want to install and use the software, but it would not be enough for those who potentially want to debug or add new features. Thus, end-users will need real-time access to the latest sources, and a version control system.

5.2.5 Guidelines and documentation

If someone is considering contributing to the project, he/she will look for developer guidelines and documentation.

The basic elements of guidelines are:

- pointers to forums for interaction with other developers,
- instructions on how to report bugs and submit patches, and
- some indication of how development is usually done

Documentation should be available (even if uncomplete) from two places: online (directly from the web site), and in the downloadable distribution of the software. Developer's documentation is written to help programmers understand the code, so they can use and also extend it.

Visitors might need to get in contact with the persons involved with the project. Thus, the addresses of mailing lists, and any other forums where others involved with the software can be reached should be provided.

5.2.6 Example Output and Screenshots

Data and code examples/demonstrators or visual presentations like screenshots can help in understading the working of the library and the results that can be get out of it.

5.2.7 Announcing and Dissemination

After the website and all the material is up and running, the announcing of the project is fundamental, together with an effective dissemination strategy, in order to engage as much as possible with new end-users.

The MSVTK library release should be posted first of all via all partners' dissemination channels, then in specialised forums/mailing lists and presented into relevant conferences/journals.

5.3 Early adopters

As already mentioned one of the main outreach objective will be achieved by having the project partners adopting the MSVTK in the development of specialised biomedical application based on their specific frameworks. The result in terms of exploitation will be twofold:

- disseminate MSVTK within the respective partners' communities, and
- further support and contribute to the development and maintanence of MSVTK after the end of the funded project.

Few lines on the characteristics of each of the partner framewroks is reported in the next sections.

5.3.1 VTK

The Visualisation ToolKit[6] (VTK), developed by Kitware, is an open-source, freely available software system for 3D computer graphics, image processing, and visualisation. VTK consists of a C++ class library and several interpreted interface layers including Tcl/Tk, Java, and Python. VTK supports a wide variety of visualization algorithms including: scalar, vector, tensor, texture, and volumetric methods; and advanced modeling techniques such as: implicit modeling, polygon reduction, mesh smoothing, cutting, contouring, and Delaunay triangulation. VTK has an extensive information visualization framework, has a suite of 3D interaction widgets, supports parallel processing, and integrates with various databases on GUI toolkits such as Qt and Tk. VTK is cross-platform and runs on Linux, Windows, Mac and Unix platforms.

5.3.2 MAF

The Multimod Application Frameworks[7] (MAF), developed by B3C, is an open source freely available framework for the rapid development of applications based on the Visualisation Toolkit and other specialised libraries like Qt. It provides high-level components that can be easily combined to develop a vertical application in different areas of scientific visualisation. The framework allows the development of multimodal visualisation application where different views of the same data are synchronised and, when the position of an object changes in one view, it is updated in all the other views. The last version of MAF is cross-platform and runs on Windows, Mac and Unix platforms.

5.3.3 GIMIAS

GIMIAS[8], developed by UPF, is a workflow-oriented environment for solving advanced biomedical image computing and individualized simulation problems, which is extensible through the development of problem-specific plug-ins. In addition, GIMIAS provides an open source framework for efficient development of research and clinical software prototypes integrating contributions from the Physiome community, while allowing business-friendly technology transfer and commercial product development.

[6] http://www.vtk.org
[7] http://www.openmaf.org
[8] http://www.gimias.org

6 References

Karl Fogel, “Producing Open Source Software. How to Run a Successful Free Software Project”, producingoss.com/