VIVO Contributed Software Task Force Recommendations

Context

Many VIVO community members have developed applications and tools to support their VIVO implementations. However these applications and tools are not organized with the distribution of VIVO or clearly identified from project documentation and websites. Other open source projects have documented guidelines and instructions for both developers contributing software and for adopters. This task force will create a three to five page white paper recommending guidelines for accepting, distributing, and using contributed software within the VIVO community.

Areas of recommendation

There are three major areas that this task force will given attention to for the recommendations to follow.

Understanding, developing and standardizing community processes

Currently the VIVO community at large is lacking a clearly documented process for suggesting, organizing and acting on contributions that might benefit the entire community (or sub-communities therein). Without standardized community processes, much of the valuable community-led work that could be integrated into core VIVO documentation and tools go unrecognized or worse become orphaned and eventually fall into disrepair.

Facilitating the exposure and discovery of existing VIVO apps and tools

The VIVO community has a set of existing applications, tools, add-ons and demonstrators that expose the ingenuity and vigor of the active and engaged participants of the community-at-large. Currently, some (but not all) of these applications can be found on the VIVO website or wiki, but these sources are neither up-to-date, nor consistently detailed enough for all of the apps and tools being tracked. Such unmaintained and non-standard pages do not accurately represent the creativity (or activity) of the community and make it difficult for new community members and prospective VIVO adopters to understand the rich set of community-contributed tools that make VIVO a compelling application stack.

Fostering, supporting and stimulating contributions from the VIVO development community

While existing applications and tools are a necessary foundation for the VIVO community, it is the development of new tools and features that keep VIVO active, thriving and growing. Today, it is difficult to understand where community members can begin to work together to develop tools or software that are not part of the core distribution.

Summary

The task force currently makes three broad recommendations :

RECOMMENDATION #1

Develop concrete, visible guidelines through ***identifying gaps in community processes*** *to assist VIVO contributors (current and potential) with organizing and sustaining work.*

Action	Example(s)
Communicate licensing information to contributors.	**Licensing :** What are the acceptable licenses for a contributed software application or tool?
Communicate processes and guidelines around (1) what projects are "community developed and supported", (2) what is the development process, and (3) how one might find answers to common contribution questions.	**Maintenance :** Software has been developed and made available under the VIVO Github[1] organization but for some projects there is no clear maintainer or, in others, a community of users. Efforts should be made to identify community members from multiple institutions to support a project; this will help distribute work, sustain effort, engage the community and create acceptance. Clear succession guidelines should also be established to hand off projects to other community members when the original contributors are no longer involved.
Draft, circulate and promote contribution guidelines and introductory documentation for getting involved.	**Getting started guide :** There are many good examples of contributing guidelines, or "how to get involved?", from various projects and organizations [2]. What is the minimal amount of documentation required for a contributed software? Where should it be made available and how? Each project might approach this differently but guidelines should be available.

RECOMMENDATION #2

Update and promote the current catalog of contributed applications and tools.

[1] Github is used throughout. It should be used as a placeholder for whatever open code sharing tools the project is currently using.

[2] https://blog.newrelic.com/2014/05/05/open-source_gettingstarted/ ; https://guides.github.com/activities/contributing-to-open-source/

Action	Example(s)
Solicit and **assign volunteer maintainers** to the applications and tools list.	The “Apps and Tools” catalog/directory should be maintained by the community with the guidance of the leads of the Apps and Tools working group. Having an updated catalog allows (1) VIVO supporters to use the catalog as training and promotional materials, (2) organizations looking to implement VIVO to have a prominent place to see the showcase of tools to consider when making implementation decisions, and (3) the entire community to have an authoritative and up-to-date place to learn more about active VIVO extensions and supporting software.
Set clear guidelines and criteria for inclusion within the catalog.	Projects should only be listed if there is an active community member willing to be the point of contact and there is freely available software to download that adheres to the community licensing structure. This, for example, could be either a public code repository or a project website with a clear way to obtain the software. Projects that are known to exist but not released to the community should be asked to adhere to the guidelines or not be included (e.g. contact Sue at X university to get the code is a violation of the guideline).
Establish tool categories to identify different types of application and tool contributions inside and outside the VIVO community.	The vivo-docker (https://github.com/gwu-libraries/vivo-docker) work is community developed and maintained and is specifically for VIVO. On the other hand, Karma (http://www.isi.edu/integration/karma/) or OpenRefine (http://openrefine.org/) are other opensource tools developed by different communities but are used extensively by members of the VIVO community.
Identify and appropriately tombstone deprecated, unmaintained or version-incompatible tools.	Projects will undoubtedly be no longer maintained, not work with current versions of the VIVO software or ontology, or be no longer of interest. These should be clearly identified and removed from the Apps and Tools catalog. A separate wiki page should be created to place references to these projects for historical reference. It’s important that people visiting the catalog only find maintained, relevant, and up-to-date information about available tools.

RECOMMENDATION #3

Create a “VIVO labs” organization to facilitate co-development of new projects.

Action	Example(s)
Create a "VIVO labs" Github organization to facilitate the community development of VIVO contributed software.	**VIVO “labs” :** The goal of the labs site is to engage community members with contributed software, both in terms of developing and in using it, and also to gain acceptance. The Hydra and Fedora Projects’ [3] use of a separate “labs” Github organization to facilitate development of non-core applications and tools, as well as more generally the Apache Foundation’s incubation process[4], can serve as models. VIVO labs would create co-development space for VIVO organizations to work together to solve common problems and share tools with the community. A key prerequisite of a labs project should be that two or more organizations agree to work together on a project. This will encourage idea sharing and improve opportunities.
Establish workflow process recommendations for VIVO Labs projects.	**The “Hydra Labs” model :** The Hydra project "graduates" tools from labs to core when they reach a certain level of adoption and sustainability. VIVO should also develop a policy around graduating projects from labs. The Apache Incubator model : The Apache Incubation process (http://incubator.apache.org/incubation/Process_Description.html)l has the idea of project community "acceptance" and "engagement", which might facilitate deeper contribution and encourage broad community support, thus boosting project success rates. See the sample workflow in appendix B for an example of how this process might work.

[3] http://projecthydra-labs.github.io/
[4] http://incubator.apache.org/incubation/Process_Description.html

Appendix A - Contributed software: A definition

Contributed software is something intended to work in conjunction with a VIVO implementation but independent from a VIVO release. For example, a PHP client library developed by an implementation team for the VIVO SPARQL Update API could be considered contributed software. The ORCID to VIVO Python web application that George Washington University is developing is an example of a supporting application initiated and developed by community members. These apps or tools help VIVO sites get work done but aren't a core component of a VIVO software release.

Over the years there has been significant effort in this area but to-date there haven't been guidelines or procedures to identify this work, encourage development across institutions, or to include contributions as part of the VIVO suite of software.

Appendix B - Example workflow for creating a "labs" project

The Brown and Dartmouth team meet and decide to jointly develop a Python client library for the Harvard Catalyst Disambiguation Engine. This library will support publication ingest and approval workflows at their sites and could have broader appeal.

- a VIVO labs contributor creates a new repository under VIVO labs called "catalyst-engine-client"
- announce on the mailing list the initiative
- add an open license, e.g. MIT, to the project
- write a clear README identifying what the project is, why it is helpful for the community, and who is developing and maintaining it
- begin collaborating
- when a minimal level of stability is reached, add the project to the Apps and Tools catalog
- after a period of time (e.g. three months for smaller projects or six months for longer projects) provide a community summary of the status of the project and future plans, either through email or by speaking during the VIVO Apps and Tools call
- several VIVO sites use the client library for a period of time (six months or longer)
- at this point, two things could possibly happen
    - the project is in use at three or more institutions and "graduates" to a fully supported
    - the client library is no longer developed (e.g. something else comes along to replace the disambiguation engine) and no community members are using the tool. A message to the community confirms no use. The project is marked as deprecated.

Appendix C - task force members

The following contributed to discussions and drafting the recommendations:

- Chris Barnes, University of Florida
- Paul Friedman, Northwestern University
- Ted Lawless, Brown University
- Keith Maul, National Center for Atmospheric Research (NCAR)

Jim Blake, Cornell University, also participated in the discussions.

Submitted 6/30/2015.