DATA AND USABILITY EXTENSIONS OF THE VIRTUAL BRAIN FOR ENHANCED EBRAINS INTEGRATION

Concept document for call

“I90 42287839 OV The Virtual Brain”

Table 1 - Details about the proposal author

Entity Proposing	
SME	S.C. CODEMART S.R.L (CDM) www.codemart.ro
PIC (EU)	925125279
Lead	Lia Domide lia.domide@codemart.ro
Contact	Romania 400610, Cluj-Napoca Str. Petofi Sandor, Nr 13, Ap 4 Tel +40744580134 +40364401158 info@codemart.ro

Table of Contents

1. References

TVB

The Virtual Brain project is an open source brain simulation project, with over 35.000 downloads in the neuroscience community. It evolved over the last 10 years, and we are proud to have helped with our software engineering expertise[1].

Since 2010, when we started collaborating with TVB leader Viktor Jirsa from INS-AMU and his team, Codemart has become the main code contributor in TVB project repositories[2]. Codemart deals with software development in all stages: from requirements gathering, analysis, architecture, coding, packaging, testing, optimization, deployment, documentation, then requirement changes and the loop repeats. Even if our contribution does not include the latest developments in neuroscience to the project, we had the opportunity to gather a solid understanding of the relevant neuroscience for TVB.

RELEVANCE: TVB is a Python 3 project, with both a web and a console interface, the latter is intended to be used through Jupyter Notebooks especially, for which we have even some small visualization dedicated tools[3] that can be reused for the current call with very slight adjustments. Also in TVB, we already included data adapter, by integrating with other tools like AllenSDK, BIDS, plus we recently built in TVB a REST API, which will be useful for the current call as precursory work and prove of TVB's adaptability.

TVB-Cloud

The Virtual Brain Cloud[4] initiative is a natural evolution example for TVB product: moving into the cloud, getting new front-ends for access, as well as a new research focus domain (Neuro Degenerative Diseases). This is an H2020 funded project, under the lead of Dr Petra Ritter from Charite-Berlin, a project of four years from Dec 2018 until Nov 2022, and Codemart is a named partner in it.

In this project, Codemart contributes among others tasks by improving modularization of TVB core, new deployments, encrypted storage, in a way that it does not break the release track for current users, but it allows TVB to enter a world where switching between local machines and cloud computing is necessary and smooth.

RELEVANCE: TVB-Cloud is a partner project with HBP, and Codemart (in collaboration with Charite) integrated TVB with EBRAINS by OpenShift deployments of TVB web GUI at CSCS[5] and Juelich[6], Jupyter Hub installations with TVB tools for HBP logged users, Unicore usage for jobs submission on HPCs. These EBRAINS integrations, while not being a direct solution to the current call, are relevant for very particular already understood tasks like usage of EBRAINS AAI (Authentication and Authorization Infrastructure), communication with HPC, data security and protection etc.

We don't include more detailed descriptions of these projects here, nor a rigorous argumentation on why TVB is relevant for this procurement, as TVB is directly mentioned and described in the procurement call.

[1] TVB main site https://www.thevirtualbrain.org/tvb/zwei/teamwork-contributors
[2] TVB's open source code https://github.com/the-virtual-brain/tvb-root
[3] Plots https://github.com/the-virtual-brain/tvb-root/tree/master/scientific_library/tvb/simulator/plot
[4] TVB-Cloud consortium https://virtualbraincloud-2020.eu/tvb-cloud-consortium.html
[5] TVB web in OpenShift at CSCS https://thevirtualbrain.apps.hbp.eu/
[6] TVB web in OpenShift at Juelich https://thevirtualbrain.apps.jsc.hbp.eu/

2. Pre-existing and relevant preliminary work

Work in TVB

We have been contributing with software development into TVB official code repo[7] over the past 10 years. We started by writing a detailed white paper for TVB, together with INS-AMU and Codebox on the architecture and basic technologies, in parallel with developing POCs for proving that the implementation can be done in the way the document was written. Then we had regular (about every quarter of the year) weeks of side-by-side code development with the Marseille researchers team on defining TVB core parts, like the DataTypes, or the simulator API. We worked in an iterative manner on TVB software (applying the SCRUM AgIle methodology most of the time, and Kanban during maintenance periods) with a dedicated Scrum Master role from Codemart, and a professional tasks management system[8] maintained by us. We have been writing and maintaining automated unit-tests for TVB which are run in a continuous manner with every new commit in the repository, and we maintain in synch with TVB code a documentation site for developers, contributors and end-users[9].

Since 2011, Codemart team members have regularly participated in major international neuroscience conferences and exhibitions like SfN, OCNS, or even HBP events. Since 2014, Codemart is an official mentor for the Google Summer of Code and has been invited to the prestigious annual Mentors Meeting several times. Our team members are also contributing to scientific papers in computational neuroscience related to this project (the list is longer, only 3 are mentioned below):

- *The Virtual Brain (TVB): Simulation Environment for Large-Scale Brain Networks*.- Jirsa V., Woodman M.M., **Domide L**. (2021) - In: Jaeger D., Jung R. (eds) Encyclopedia of Computational Neuroscience. Springer, New York, NY. https://doi.org/10.1007/978-1-4614-7320-6_100682-1
- *Toward a Pathway Inventory of the Human Brain for Modeling Disease Mechanisms Underlying Neurodegeneration*. -- Anandhi Iyappan Michaela Gündel Mohammad Shahid Jiali Wang Hui Li Heinz-Theodor Mevissen Bernd Müller Juliane Fluck Viktor Jirsa **Lia Domide** Erfan Younesi Martin Hofmann-Apitius -- Journal of Alzheimer's disease : JAD -- Published on 12 Apr 2016
- *The Virtual Brain: a simulator of primate brain network dynamics* -- Paula Sanz Leon Stuart A. Knock M. Marmaduke Woodman **Lia Domide Jochen Mersmann** Anthony R. Mcintosh Viktor Jirsa -- Frontiers in Neuroinformatics -- Published on 11 Jun 2013

EBRAINS

While working with Charite team (as subcontractors as well as partners in TVB-Cloud), Codemart team dealt with TVB's more recent requirements of: support for HPC execution of high demanding jobs, increased and well documented security due to managing personal data and the need to comply with GDPR rules. Codemart then implemented these new features into TVB, with integration of technologies like Keycloak (AAI of EBRAINS), Unicore (for jobs submission and communication on HPC), Syncripto, REST, etc.

[7] TVB code contributor's list https://github.com/the-virtual-brain/tvb-root/graphs/contributors
[8] Jira for tasks management https://req.thevirtualbrain.org/
[9] TVB documentation site https://docs.thevirtualbrain.org/

Within EBRAINS infrastructure, we use the OpenShift technology (both for the installation at CSCS[10] and the one in Juelich[11]) to deploy TVB web GUI. Any user with access to an EBRAINS account can then get access to these TVB installations. We separate data by user, thus work of a user is kept private unless the user chooses to share it with other users. We have used a dedicated group for TVB core members to give them further access inside the TVB web app to our settings page.

Still through OpenShift, we did Jupyter Hub installations at EBRAINS, which are using the EBRAINS authentication, and offer a Jupyter kernel with tvb modules installed (tvb core mainly, but also tvb-multiscale[12]).

In TVB software we already have introduced the concept of Adapters[13] for external algorithms, as part of the main architecture. This concept can be used as a base for the interaction between TVB and EBRAINS, for the data sources in particular (through TVB Creators or TVB Importers categories). As a first example we recently implemented (for TVB-Cloud project) the adaptation of the public TVB Tumor[14] dataset from EBRAINS KG into TVB.

SIIBRA

We have been in brief contact with Timo Dickscheid and his team from Juelich, on integration between TVB and the Brain Atlases. We started by analyzing existent documentation[15] of such a possible integration. We recently submitted a tutorial[16] into Siibra Github Repo, after exploring current means of integration between TVB and siibra, and we also identified together a few points where the siibra API could be improved.

We have a proven history of managing TVB architecture and code-base, in which time TVB has grown into a positive example of a well maintained and functional scientific tool with a large user base (>35K downloads since 2012). Having a team of professional developers dedicated to work in a science project, not only brings improved stability in time, but also a high level of the much valued quality for reproducible results, as well as optimizations at the architecture and code level, plus a good efficiency in the implementation time.

[10] CSCS open shift https://okd.hbp.eu/

[11] TVB web GUI entry point towards EBAINS installations https://tvb.apps.hbp.eu/

[12] Jupyter Hub installation with tvb-multiscale https://tvb-nest.apps.hbp.eu/

[13] Adapters module in TVB codebase http://docs.thevirtualbrain.org/py-modindex.html#cap-a

[14] Tumor dataset adapter towards TVB integration https://github.com/the-virtual-brain/tvb-root/blob/master/framework_tvb/tvb/adapters/creators/tumor_dataset_creator.py

[15] HBP atlas data possible map to TVB https://wiki.ebrains.eu/bin/view/Collabs/atlas-tvb-mapping/

[16] Siibra - TVB tutorial https://github.com/dickscheid/siibra-tutorials/blob/main/06-SIIBRA-TVB.ipynb

3. Technical concept

During the past years, the Human Brain Project (HBP) has gathered multiple tools and developed unique multiscale modeling methodologies. These tools and techniques allow the integration of multiscale heterogeneous data, models and algorithms. Successful integration and ease of use of these components is critical for personalizing brain models, for closing the gap between models and empirical brain measured signals, for enabling validations and eventually clinical applications.

We understand the current call is about building a bridge between existing tools, but also between tools and the end-user who in many cases will be safer in a graphical user interface, where the user has enough flexibility to do his research, but not be bothered with too many technical details which are hidden behind nice and reusable widgets.

The challenge we aim to solve here, is to satisfy several heterogeneous constraints comprising the linking and partly enhancing of already existing and well functioning systems like HBP’s KB, Brain Atlases and TVB simulator while still keeping them well functioning independently; the using of an infrastructure with high potential capabilities from EBRAINS like HPC, authentication and authorization, shared drives, data protection and security; and the combining of forces from all of the above with easy querying tools, safely configuring and scheduling GUI components to work together for performing efficient multiscale simulations or other custom operations from analysis to visualizations.

3.1 Methodology

While bidding a project requires a fixed cost, clear time estimations, a set of well specified and accepted requirements, plus concrete deliverables, which we are happy to agree and provide here; on the other hand when implementing a contracted project, we prefer the agile methodology with incremental refinement of the initial requirements, and continuous feedback from involved actors. This is the methodology we will also adopt in this project, and over the years we found this fits very well with the research world, where requirements are far from being a priori fully known and an “easy to assess” piece of information in one step, regardless of how many resources are allocated.

Thus, the following work plan is written as one that has hook points or variable paths, to be assessed over the iterations to come in this project. For example, we cannot finally assess the exact number of GUI widgets to be produced for the second task, but we can confidently say we will cover all described widget categories with more than one such tool.

Thus, we plan to work using SCRUM Agile methodology, with sprints of adjustable lengths (2 - 5 weeks). The first sprint will be dedicated to architecture, tools analysis and basic code design. Every sprint will need to leave the work in a verifiable state: architectural documents ready to share, demo servers with live prototypes on, or early versions with limited features, up to the final product. At least every two months, we will have demos with the latest developments targeting the extended group of involved people.

A close interaction with WP1 teams is intended: collaboration with Viktor Jirsa’s team on TVB core simulator, with Timo Dickscheid's team on the brain atlases and siibra side, and with SDL Neuroscience group in particular for this call. We will synchronize our sprints with their work pace, and hold demos at the end or our sprints with and for them, to gather input as well as feedback.

Throughout the present document we insert multiple footnotes. While in general they are not a necessity for the reader to follow, we consider most of them important in establishing proof of our expertise and competence.

3.2 Implementation

The here proposed work aims to better integrate TVB with HBP world, as a software package as well as with the infrastructure, by implementing adapters between TVB and EBRAINS Multilevel Brain Atlas, and also with its Knowledge Graphs, but also to better integrate other EBRAINS tools in general through the development of GUI widgets for iPython.

The three technical components described by this call's concept document will need to complement each other, as well as to fit into existing TVB and EBRAINS work. These three required components give us the following subchapters, where we describe how we envision their implementation to happen.

Extensions to TVB for data access and integration from the EBRAINS KG and Siibra

We understand that ERBAINS is cataloging and curating a lot of data and relations, currently grouped in the Knowledge Graph. This gives means to query and analyze heterogenous spaces and information about the brain. The Multilevel Brain Atlas holds extremely valuable, but currently unlinked to TVB information for multi-scale, otherwise compatible with TVB simulator. Siibra Python API is a very nice addition to the Brain Atlase, as it opens the atlases information to a stage or reusability, thus we applaud its development from the software developers perspective.

In TVB software we have the concept of Adapters[17] to external algorithms and sources or data, as part of the main architecture. Adapters work with inputs of type DataType and produce results DataTypes. A custom algorithm will then be adapted for usage in TVB. This concept can be reused as a base for the interaction between TVB and EBRAINS data sources. It basically gives us a hierarchy of classes to inherit from (the abstract root ABCAdapter, plus its subclasses like Creator, Importer), but most importantly, we can describe on an adapter class the inputs and outputs structures through objects.

Having Siibra API used in a TVB adapter of type creator, we see an immediate opportunity of retrieving from an EBRAINS atlas, structural or functional connectivities into TVB[18], as the very first step. The user could control the atlas, its version, and other parameters. Launching simulations on such atlas extracted averaged structural connectivity will have value in itself, for studying TVB models, for example. But more interesting will be to map various other Atlas information (cross scales) to certain spatial configurations of the simulator neuronal mass models (e.g. to match a pathology to a personalized brain, or aging a personalized brain in a simulation). For this second stage, we will create other regional level DataTypes like ConnectivityMeasure in TVB, or create new dedicated DataType classes, if more specific structures are identified. Here, at the link with models, the recent modality of RateML[19] in TVB (having the Juelich and INS groups as main authors) proves even more usable for such links to be described scriptically, as the notation is more flexible than in Python.

[17] Adapters module in TVB codebase http://docs.thevirtualbrain.org/py-modindex.html#cap-a

[18] Compute a TVB connectivity in siibra https://github.com/dickscheid/siibra-tutorials/blob/main/06-SIIBRA-TVB.ipynb

[19] RateML in TVB https://github.com/the-virtual-brain/tvb-root/tree/master/scientific_library/tvb/rateML

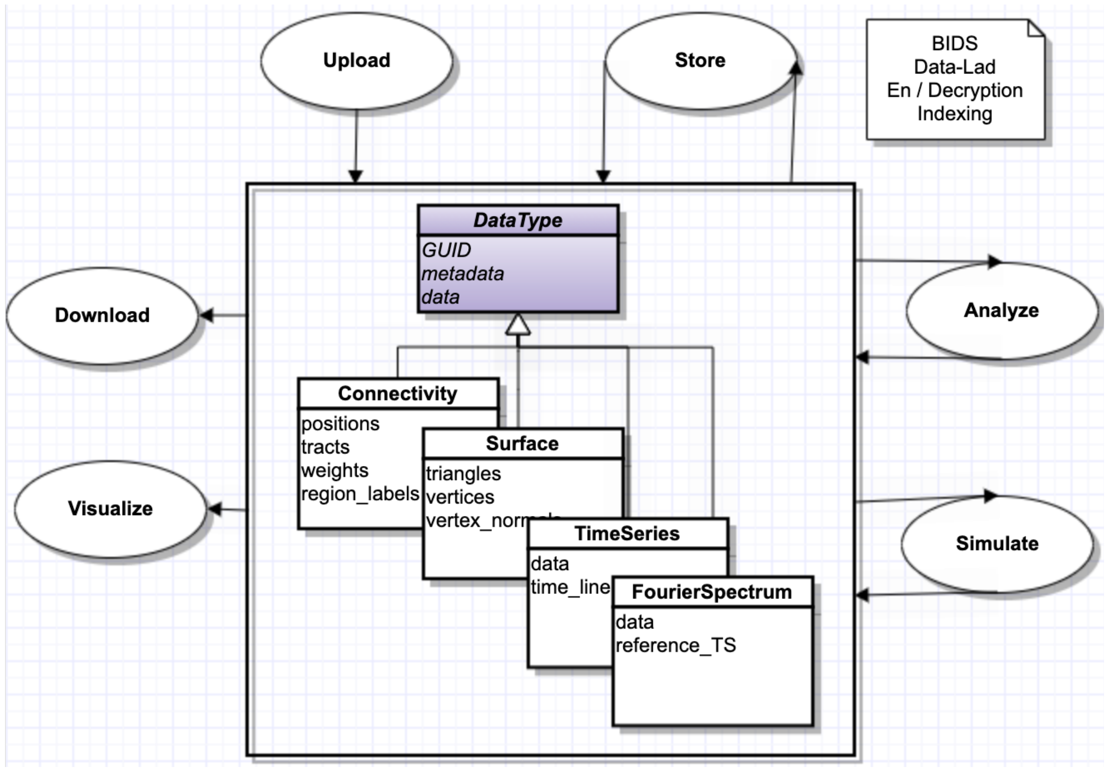


Fig. 1. TVB data centered architecture with algorithms linked to DataType inputs and outputs as "adapter" subclasses. We envision new classes in the oval sections for linking with KG and Siibra

The integration with the KG on the other hand requires a bit of a different paradigm, than the interaction with the multilevel atlasses through siibra, as the KG queries we envision to have a more instant feedback given to the user, both in the write of query usability through autocomplete and suggestions, but also in terms of presenting results as a preview, the not so fixed form of the results, etc. Here we intend to first analyze the potential of existent open source JS modules which already exist (for example at SCAI[20]) in work with KG, and incorporate those on a new web page in TVB GUI. In case none will fit our needs of interaction, then we will take the path of writing new custom widgets. Then, the outcome of such KG queries will still be DataType objects that will go through the normal TVB usage into the simulator configuration.

As TVB web GUI supports integration with EBRAINS AAI, and we can programmatically get the current user's token, this can be further propagated when interacting with other EBRAINS modules, like the KG in this case, for getting access to restricted datasets.

Combining multiple data features will be possible through the persisted DataTypes into TVB, as identified in Fig 1 from above. The user will incrementally gather multiple compatible such DataTypes (produced by existing or new adapters in TVB) all in a TVB Project which can be shared or exported.

New means of validating compatibility of these DataTypes can be envisions (not already part of TVB), but these will become necessary with the inclusion of multiple data sources as described here.

Current TVB Distributions version 1.* or 2.* have a local storage next to them. Where TVB Web CherryPy service runs, there is also a HDD volume with the storage. We need both a relational DB index and an actual folder/project, to store H5 files in (encrypted or not depending on the installation configuration). This is illustrated below in Var A (left side) of Figure 2.

[20] SCAI as producer of JS tools for KG https://neurommsig.scai.fraunhofer.de/

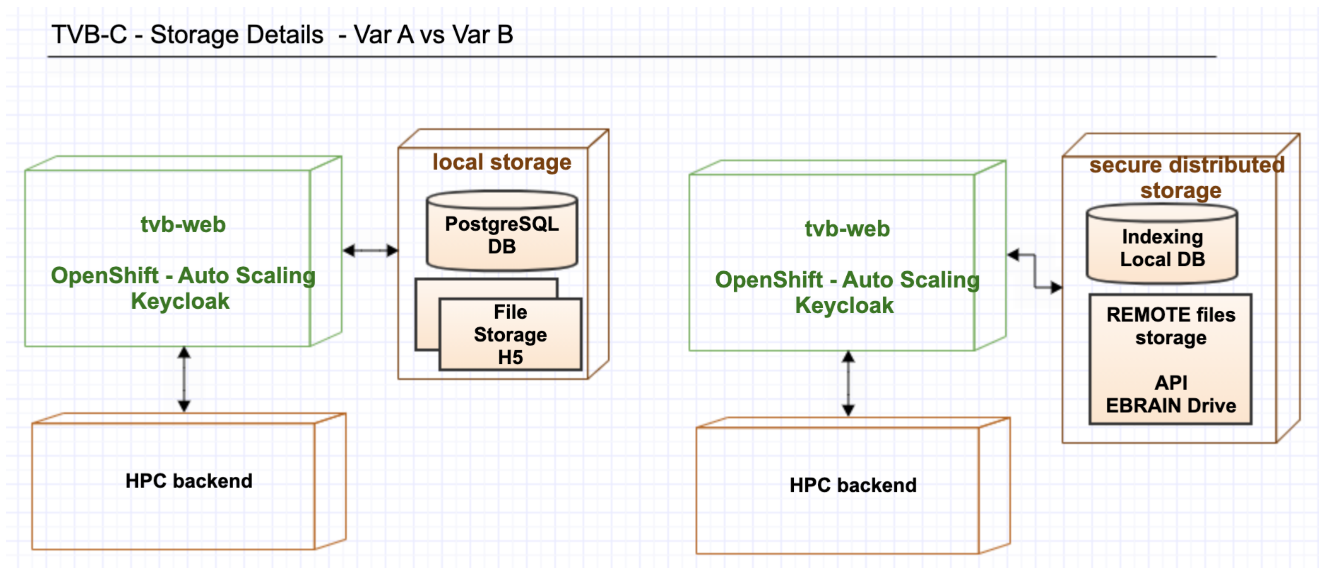


Fig. 2 Proposal to replace the current TVB local storage solution with a secure remote storage, potentially EBRAINS Drive

Var B is something we consider to develop, as a cloud specific architectural feature, for making sure TVB is scalable also from the storage perspective. We've started in the previous versions, by detaching storage functionality of tvb code into a separate package named tvb-storage[21]. While the current tvb-storage versions < 2.4 are based on local storage, in the future, the implementation from the recently detached API will make use of storage backend like EBRAINS Drive[22] or Bucket. Although this feature is not explicitly mentioned in the current call, it might fit nicely with EBRAINS integration, and is something to be considered for further discussions with the Juelich side while planning the concrete sprints of this project.

Development of Graphic User Interfaces for iPython Notebooks

Although graphical user interfaces are the most fun to develop parts for a software developer (as one sees something immediately), we are well aware that making tools which are useful with all the functional and nonfunctional requirements listed in this call (ease of use, feature complete, modular, easy to extend, etc) is not a simple task, but based on our experience we are confident that we are up to make this task a success and to comply to all the requirements.

TVB's current web interface, although it has some faults and could use some improvements or a facelift here and there, has mostly stayed online and viable for over 10 years. We will use that experience, but we will not be bound to the existent web components in tvb visualizers for this call. As ipython notebooks are web, we can envision reusage from and with TVB web GUI.

Based on this call's concept document description, we identify these categories of iPython GUI widgets to be developed:

1. Data introspection and selection (selection of data sources, previews of queried data from Siibra and the KG)
2. Operations configurators (Simulation, cohort simulations, parameter explorations, integration of a subset of TVB analysis tools,

[21] tvb-storage package https://pypi.org/project/tvb-storage/

[22] EBRAINS Drive https://github.com/HumanBrainProject/ebrains-drive

3. Operation execution scheduling and tracking (immediate execution, scheduling on HPC, serialization for export of the configuration)

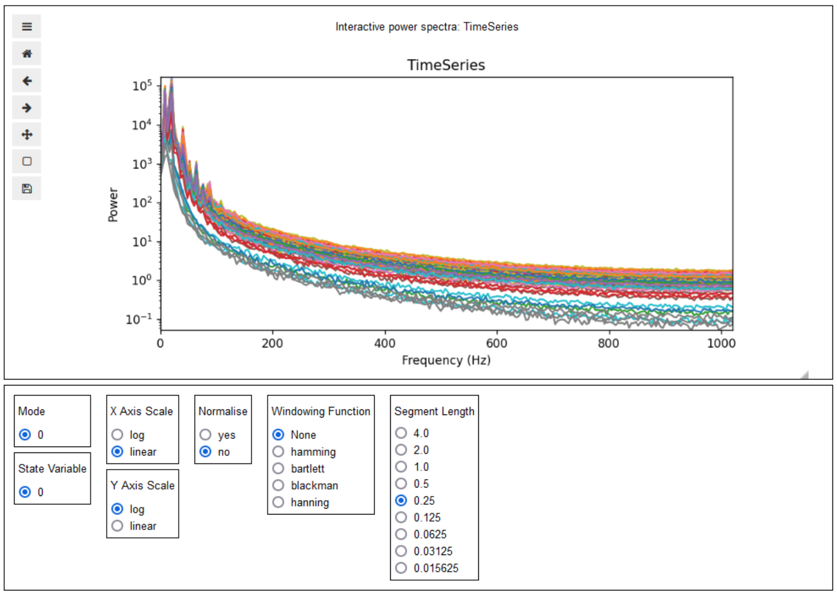
This task will have its own repository, which we will set-up and integrate a Continuous Integration flow execution with it (either a new dedicated one, or the ones currently running with TVB).

As implementation technology, we still need to finalize an analysis report, but we will most probably develop on top of existent Jupyter widgets[23], in combination with other plotting tools like matplotlib or plotly (for the visualization widgets, including previews). We can use Layout templates[24] from ipywidgets for a flexible, customizable but still clean arrangement in the page. The Interact decorator[25] is also important as the configurator new controllers will be expecting user interventions.

We will also focus on the new widgets API (as interface, signature or call manner), to be as clean and clear as possible. We still consider the simple approach that new widgets might depend on only primitive inputs (if they are few - less than 5 - and simple enough - accepting numbers and arrays only), but most probably we will need to go towards describing the widgets API with a set of objects (for inputs and outputs). For this latter case, we suggest repeating the DataTypes centered architecture as used in TVB and described briefly in the previous chapter (Fig 1).

Thus we are after a modular design, easy to extend and integrate into workflows, with DataTypes that allow seamless exchange of information between components, easy to use and arrange GUI components, with the purpose to hide complex operations behind a clever API, and to offer templates of functions to be customly filled by dev-users with scripts needed for their own WP showcases.

These GUI widgets should be well packaged for usage. To be decided if only one Pypi package is ok, or we should divide these into multiple such packages, for the ease of use - mostly it will depend on 3rd party dependencies, and how popular the widgets depending on heavy 3rd party libraries are expected to be. A Spack[26] descriptor for this new module(s) integration with EBRAINS kernels is envisioned as well.



Mode: [✓ 0]

State Variable: [✓ 0]

X Axis Scale: log [✓ linear]

Y Axis Scale: [✓ log] linear

Normalise: yes [✓ no]

Windowing Function: [✓ None] hamming bartlett blackman hanning

Segment Length: 4.0 2.0 1.0 0.5 [✓ 0.25] 0.125 0.0625 0.03125 0.015625

Fig. 3. Example of a reusable visualization widget for notebooks from TVB

[23] Jupyter widgets https://jupyter.org/widgets
[24] Layout templates https://ipywidgets.readthedocs.io/en/latest/examples/Widget%20Styling.html
[25] ipywidgets Interact https://ipywidgets.readthedocs.io/en/latest/examples/Using%20Interact.html
[26] Spack as packaging tool https://spack.readthedocs.io/en/latest/

During the recent EBRAINS online summit, in the internal day presentations, we were made aware of existing proof of concept pieces from EBRAINS groups (Juelich SDL Neuroscience in particular), and we will make close note of those, to incorporate as much as possible existent knowledge and agreed usages, and not reinvent the well, nor loose precious time with rethinking already made decisions.

Documentation

We plan for the following documentation to be written mostly during the development, and only left for final review towards the end of the project:

- Inline code documentation for every public class, and method. This documentation will be automatically extracted from the new code and merged with TVB main documentation through Sphinx; it will be readable directly in Github but also on TVB docs site[27]. This type of documentation (together with appropriate variables and method names, plus a good and clean code design) ensures that *developers* can reuse code, and future changes in requirements will be easier to solve.
- Also *for developers*, a tutorial or blogpost article should be provided to describe how to extend with potential new features these new components. Its place seems to be on TVB docs site, in the Developers section[28], but crosslinks should be added also on EBAINS collabs.
- For *operators and high level support teams*, instructions on how to install and maintain the new modules will be left in the online documentation of TVB. Although a dedicated PDF could easily be generated also - similarly with how we do it for other manuals, and handouts in TVB, as the RST documentation of TVB[29] is already modular and versatile - we don't foresee this necessary, as the installation/maintenance should be as simple as installing/upgrading a Pypi or conda-forge module. Anyhow, if this will prove necessary, the manual can easily be generated as a separate document.
- *For end-users* we intend to have a dedicated EBRAINS collaboratory (plus crosslinks in existent official TVB Collab[30]) with easy to understand tutorials on how to operate the new GUI components, and similarly with existing TVB documentation, few jupyter demos on TVB official documentation page [31].

3.3 List of services and deliverables breakdown

We consider the following main services (or technical tasks) to be perform by Codemart as part of this project, across all deliverables:

- Write a detailed analysis of requirements document and enumerate a few more technical details needed to clarify before implementation.
- Manage, plan and supervise the work in an Agile manner:
    - Maintain and offer access to Juelich partner to a tasks management system - most probably the current one from TVB[32] which is Jira. We are aware there is an end of life for the self hosted version of Jira and we are currently planning to migrate.

[27] TVB code extracted documentation https://docs.thevirtualbrain.org/py-modindex.html
[28] Developers section on TVB docs site https://docs.thevirtualbrain.org/doc_site/top_developers.html
[29] source code for doc https://github.com/the-virtual-brain/tvb-root/tree/master/tvb_documentation
[30] Collab on TVB main https://wiki.ebrains.eu/bin/view/Collabs/the-virtual-brain
[31] TVB demos-doc section https://docs.thevirtualbrain.org/demos/Demos.html
[32] Jira installation for TVB project management https://req.thevirtualbrain.org/

    - Working with iterations or sprints, repeating the development phases like analysis, implementation, testing and deploying in small incremental steps will allow us to get early feedback from Juelich partner and to converge fast towards a usable solution. Thus the risk of having something incompatible with EBRAINS at the very end will be minimised.
    - Provide a Codemart person to act as Scrum Master for the period of the project.
- Setup and maintain a dedicated GIT repository under https://github.com/the-virtual-brain for parts of this call (the new GUI widgets - the rest will go most probably under the existent TVB repo https://github.com/the-virtual-brain/tvb-root).
- Write the code needed to implement the tasks in this call:
    - develop adapters between TVB and Multilevel Brain Atlases with Siibra, by creating new code in TVB and potentially influence siibra API;
    - develop new GUI widgets to be used in notebooks.
- Benchmark, do continuous builds and write automated tests. We proved that the use of continuous integration[33], automated tests and performance benchmarks[34] a scientific project has a great value long term, by doing it methodically within TVB. We intend to extend these for the current project as well.
    - The effort for writing unit-tests is considerable (about 30% added to the development time), but critical if we want reproducible results in incremental versions of a software.
    - For the new adapters we estimate that we will use mockup designs when writing unit-tests, for really separating the tools at test time, and be able to speedup the builds and development velocity.
- Hold demo meetings for the Juelich team and potentially other HBP teams interested to participate.
- Hold sprint planning sessions where the Juelich team will be invited especially for the Product Owner role.
- Write and deploy documentation while working in this project and also at the end.
- Work with containers for easy and stable deployment. For TVB we already started a tradition of deploying through Docker[35], and we want to propagate this tradition for the current proposed project also. It will fit nicely with EBRAINS and HPC or even cloud deployments, because it is oriented on portability, self-contained and well encapsulated modules.
- Package in appropriately sized and feature based python modules on Pypi plus conda-forge. Potentially create Spack descriptors for EBRAINS kernels.

The deliverables for this project have been described by the contractor as part of the official call concept. In this chapter we took those deliverables and divided them into smaller technical steps, so that we can easier work in an agile manner on them. We kept the official "Due month" for the larger deliverable and did not establish a more exact deadline for the deliverable subparts yet, as in some cases these smaller parts will be developed in parallel (e.g. D1.A and D1.B) or their exact due date will be better to establish in every agile sprint planning session that we plan to have, and to be assumed at that time by the entire team. We consider bounding to respect the deadlines for the top deliverables. And we reiterate here what was written in the previous chapter 3.1 "Methodology", that work will be done in sprints of variable length, but we envision at least one / month deliverable to be ready for inspection and testing by the partnering HBP groups, plus periodic demos for the team to gather and incorporate relevant feedback.

[33] TVB Continuous Integration Build setup https://jenkins.codemart.ro/view/TVB/ tester/pass
[34] TVB benchmarks (ex. methodology) http://docs.thevirtualbrain.org/doc_site/benchmarks/top_benchmarks.html
[35] TVB repos on Docker Hub https://hub.docker.com/orgs/thevirtualbrain/repositories

Table 2 - Deliverables and their breakdown

Deliverable Identifier	Deliverable Description and breakdown	Estimation story points	Due Month
D1	Software modules for data access and integration merged into the TVB repository.	8	M10
	A. New adapters in TVB codebase B. Unit-tests for each of these adapters with potentially mock implementations for the remote source C. Possibly new datatype classes in TVB (or arguments why existing ones are enough) D. GUI integration in TVB of KG queries		
D2	GUI modules available in an EBRAINS repository.	8	M14
	A. Repository setup and initialization with links towards a Continuous Integration build system B. Analysis report on technologies and basic widgets API C. GUI widgets for data selection D. Widgets for operations configuration E. Widgets for operations scheduling, execution and monitoring		
D3	GUIs and data access modules integrated into the scientific showcases.	2	M17
	A. Deploy in EBRAINS Jupyter Hub the new widgets B. Update TVB web openshift installations at EBRAINS with the new data adapters C. Updates based on feedback from relevant HBP users D. Create a Spack description to include the new GUI modules in the official EBRAINS images for the jupyter kernels		
D4	Full documentation, tests and quality assurance reports following the EBRAINS technical coordination guidelines.	1	M18
	A. Aedicated Collab space for the new GUI modules B. Deployed new pages in TVB main documentation site describing the new modules for developers, end-users and operators C. New unit-tests and integration tests running successfully in CI workflows D. Progressive summary reports E. End of work quality assurance report		

In the above table with deliverables, we included a column with estimations in story points[36]. We evaluated this by deliverable difficulty, risks, size of the work obviously but also the prioritization

[36] Story Points
https://www.atlassian.com/agile/project-management/estimation#:~:text=Story%20points%20are%20units%20of,work%2C%20and%20risk%20or%20uncertainty

of the work needed. We assigned 1 story point to the task which seemed to have fewer risks, then in the fibonacci style we estimated the others as well. When translating in physical time the estimations might not entirely reflect these proportions, as the risks will be mitigated.

3.4 Effort breakdown

Based on the above list of deliverables and their details, we compute the following effort.

Table 3 - Effort and cost

Resource		D1 - M10 - Software modules for data access and integration merged into the TVB repository.	D2 - M14 - GUI modules available in an EBRAINS repository.	D3 - M17 - GUIs and data access modules integrated into the scientific showcases.	D4 - M18 - Full documentation, tests and QA reports following the EBRAINS guidelines.
Expert Developer		2	2	1	1
Senior Developer		7	8	3	1
Developer		5	5	2	1
	Total Months	14	15	6	3

Included as covered by this cost are developer workstations and licenses for development software.

We estimate a cost in Euros/developer/day, to be multiplied with the estimated time for each of the deliverables. We consider 19 days to make a work month.

4. Available resources and skill profiles

Codemart, a Romanian software company founded in 2004, is a closely knit team of specialists, focused on delivering high quality custom based software solutions. As proof it stays our LLoyd's Register LRQA issued ISO 9001 certification for developing client-oriented software solutions, which we maintain since early 2015.

Due to the short length of the current project, and the relatively high ramp-up time needed for a software developer to join a science development group, we propose for this call a team from only existing employees of Codemart[37], all selected from the ones with TVB experience. If it will prove necessary due to an undesired change of plans (e.g. people leaving the company or the project starts late and we need more parallel tracks) we can also hire new people by our regular recruiting channels (head hunting collaboration, internships), and that will be ok, as long and the majority of the team has experience with TVB already.

We estimate that we need for this project's implementation one software architect, one senior developer and one other dedicated developer. One of these first two will also take the role of Scrum Master, participating and organizing all Scrum Agile meetings and making sure the project management and scrum methodology work well. We already have people in the company for each of these positions.

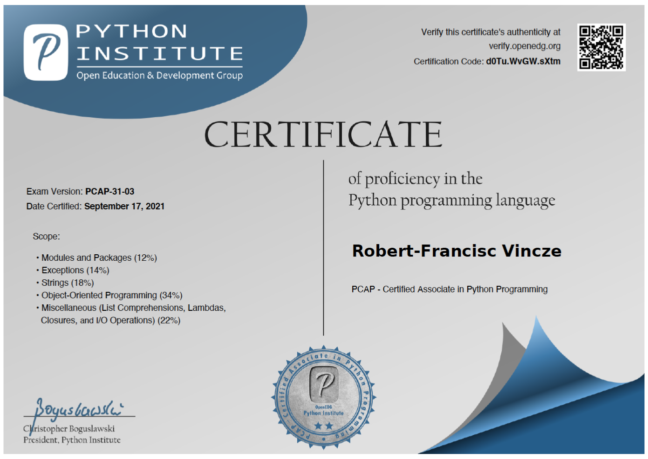
The **lead person** proposed for the current project is Lia Domide.

- *Lia Domide*[38] is one of the co-directors in Codemart. She studied computer science in Cluj-Napoca (Romania), and finished her diploma "magna cum laude" in 2007. Her first major contact with the science world was in 2007, while at a scholarship at the Eidgenössische Technische Hochschule Zurich (Switzerland). Lia has decided to take the road of the software industry and since then works as a developer and software architect, but still keeps close contact with the science world, through the nature of the projects she is coding for. She is a certified Scrum Master. She acted as a software technical lead in TVB project for over 9 years. We propose Lia as main contact from Codemart on this, thus her CV in PDF form will be attached to the submission.

For the **development**, we nominate:

- *Paula Prodan (former Popa)* graduated from Technical University in Cluj-Napoca în 2016. Paula shows great interest in applying her studies in a field with potential for making the medical world better, so she joined TVB team soon after joining Codemart(2015), with focus on the preprocessing pipeline, for extracting a structural large connectivity from MRI patient scans. Her enthusiasm is a great value for those algorithmic parts, while her optimism and gay spirit a great value for the team itself. Paula acts as lead developer in TVB-Cloud project so far.
- *Rober Vincze* is the prodigy of the team. He recently graduated in the summer of 2020 from the Technical University in Cluj-Napoca, but he joined the Codemart team one year before, as part-time. His fresh view is a great addition to a slightly old codebase such as TVB's. He recently got a Python certification and is on the way to get the more advanced one soon.

[37] Codemart team https://codemart.ro/codemart/zwei/crew
[38] CV online for Lia Domide https://www.linkedin.com/in/liadomide/



PYTHON INSTITUTE
Open Education & Development Group

Verify this certificate's authenticity at verify.openedg.org
Certification Code: **d0Tu.WvGW.sXtm**

CERTIFICATE

of proficiency in the Python programming language

Exam Version: **PCAP-31-03**
Date Certified: **September 17, 2021**

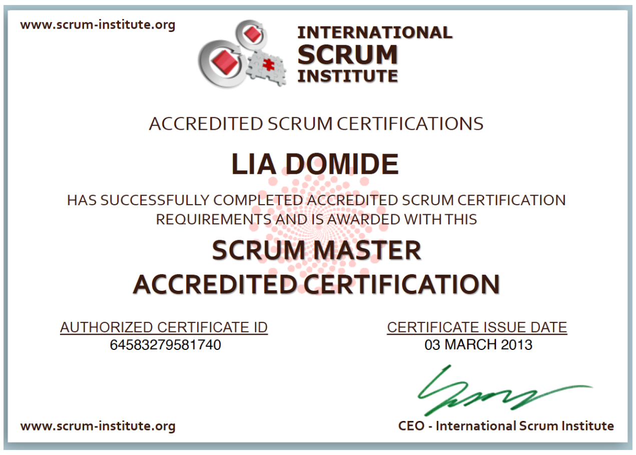
Scope:

- Modules and Packages (12%)
- Exceptions (14%)
- Strings (18%)
- Object-Oriented Programming (34%)
- Miscellaneous (List Comprehensions, Lambdas, Closures, and I/O Operations) (22%)

Robert-Francisc Vincze

PCAP - Certified Associate in Python Programming

Christopher Boguslawski
President, Python Institute



www.scrum-institute.org

INTERNATIONAL SCRUM INSTITUTE

ACCREDITED SCRUM CERTIFICATIONS

LIA DOMIDE

HAS SUCCESSFULLY COMPLETED ACCREDITED SCRUM CERTIFICATION REQUIREMENTS AND IS AWARDED WITH THIS

SCRUM MASTER ACCREDITED CERTIFICATION

AUTHORIZED CERTIFICATE ID
64583279581740

CERTIFICATE ISSUE DATE
03 MARCH 2013

www.scrum-institute.org

CEO - International Scrum Institute

Fig. 4. Two certificate samples from proposed team members

The work won't be equally distributed in time, but it will be based on the project needs. While the architect's role will be more prominent at the beginning, for designing the adapters, the new datatypes and the ipython widgets, it will also be necessary to keep her involved part-time during the entire project-time (in all sprints) to make sure all engineering solutions are matching the architecture.

4.1 Equal opportunities

Codemart has an overall gender distribution of 30% women in the full team. This is not low, considering the latest figures[39] in the field. The Codemart science division is slowly fluctuating from within the Codemart big team, depending on the project needs. Our current team for the science projects has a gender based distribution of 50% women and 50% men. While writing code is not a directly gender influenced activity, we think that a good gender balance inside a work team makes everybody behave better and creates a nicer atmosphere overall.

At Codemart we have employees between 22 and 55 years old, ranging from Junior to Expert level in computer science expertise. We highly encourage both young and older people to join the team through internship programs, flexible schedules and extra benefits for maintaining people in the company over the years. We also have periodic social events within the company, nowadays mostly happening outdoors or online, where people can discuss outside of the regular project meetings, know each other a bit better, and thus also improve the social aspect of the work we later do,as being creative work such non-technical aspects are also important.

[39] Women in computer science summary https://en.wikipedia.org/wiki/Women_in_computing#2010s

5. Rights of use, licenses and intellectual property

Codemart is a software company with ISO 9001 certification for developing client oriented software solutions. While this ISO certificate is not licensing, rights of use or intellectual property oriented directly, it has chapters for all these aspects, which Codemart is complying with.

While working with TVB so far we mostly used open source software, and we intend to continue so. TVB itself being GPLv3, any module based on TVB will be GPL as well (by intention, but also due to GPL restriction). The new code part of TVB will go though TVB main repo, which is public. Also the tasks management system[40] has public access.

The new notebook GUI widgets will have their own repository, to be established together with the Juelich partner where to sit and how to be handled exactly (even if we already are in favour of a setup as outlined in 3.3). In accordance with the requirements of the call and compatibility to the existing packages this will be put under a GPL license, copyrighted to the current authoring groups.

Codemart has signed with each of its employees non-disclosure agreements regarding data they have contact with, while working at Codemart, and a period of time after they leave the job.

Also, for this project, we will not store or directly manipulate sensitive patient or other institutions data directly, we can develop the software with mock data or anonymized data.

We pay licenses for development tools (like Pycharm IDE[41] for writing code), and especially as we are being producers ourselves, we take any intellectual property very seriously.

[40] Jira installation for TVB project management https://req.thevirtualbrain.org/
[41] Pycharm IDE for writing code https://www.jetbrains.com/pycharm/