Integrating pyArchInit and BraDypUS for field and academic archaeological research

Giuseppe Guarino[1]* and Paolo Rosati[2]

[1] Dipartimento di Storia Culture Civiltà, Università degli Studi di Bologna, Bologna, Italia;
[2] DigiLab, La Sapienza University of Rome, Rome, Italy
*Correspondence: giuseppe.guarino8@unibo.it, paolo.rosati@uniroma1.it

Presented at ARCHEO.FOSS XV 2021: Open software, hardware, processes, data, and format in archaeological research; on-line; November 23rd-26th 2021.

Abstract:

This article presents an attempt to integrate into a single workflow two important tools for the management of archaeological data: pyArchInit and BraDypUS. Although each system has its proper applications and capabilities, their combined use could have brought several advantages. However, it is worth noting that the work currently has some structural limitations that can be resolved only by the modification of the source code. This work summarises and reports on the current methodologies (as of 2021) for operating on both desktop and web platforms simultaneously using specific tools.

Keywords: FOSS archaeological documentation, Archaeological protocol documentation, QField, PyArchInit, BraDypUS

List of FLOSS software used/ data repository and licence: QFIELD (*GNU Public Licence (GPL) Version 2*), BraDypUS (*MIT licence*), PyArchInit (*GNU Public Licence (GPL) Version 2*);
Open dataset and licence: Repository available at https://github.com/Heryx/cfg_from_pyarchinit_to_bradypus, (*MIT licence*).

The contribution of "Open-Source Archaeology" for exchanging data and ideas between researchers and archaeologists.

Within the domain of archaeological research, and particularly in informatics archaeology, the relevant costs associated with software licences might present a concrete obstacle, particularly in situations where research funds are scarce. Furthermore, beyond their proprietary nature, numerous prevalent software applications in archaeology lack explicit design and development for archaeological purposes, thereby falling short of meeting the specific requirements of the discipline. Additionally, certain archaeology-specific software programs are no longer receiving support. As Benjamin Ducke noted:

> One possible solution to this problem lies in embedding software development and research within larger, open-source projects that are not confined to the scope and lifetime of a single research project. This provides a two-fold advantage. On the one hand, paying programmers to implement functionality in open-source software delivers a solid return on investment. On the other hand, it offers countless opportunities for the sharing of investments and resources, reducing the burden on the individual stakeholders. (Ducke 2012, 574)

Open-source software is distributed as readable program source code written in a (usually high-level) programming language (Wilson and Edwards 2015, 1). The availability of source code allows the end user not only to run the software but also to manipulate, change, redevelop and understand how it works.

We use different software tools for our academic research which are often not interoperable with each other. Nowadays, an important objective is finding a meeting point to allow groups and people with different backgrounds to quickly consult the archaeological data, thus increasing the exchange of resources and ideas. Our brief contribution provides an attempt to integrate some of these tools, pyArchinit and BraDypUS to allow for accessible software communication.

The software used

Initiated in 2005, the project "pyArchInit - python for archaeology" was designed to facilitate the management of data derived from archaeological contexts within the well-known and widely used QGIS platform (Cocca and Mandolesi 2013). The overarching goal of the developers was to establish a novel application for archaeologists, seamlessly adaptable to both field and laboratory settings. The envisioned application aimed for user-friendliness and seamless integration with management systems utilised by public administrations. Luca Mandolesi spearheaded the development of Pyarchinit, and Enzo Cocca further implemented it (Mandolesi and Cocca in 2022). The development is based on the commitment to maximum system portability, ensuring ease and immediacy of use, leveraging open-source software for development, creating modules for comprehensive archaeological data management, integrating existing and robust data management systems, and maintaining flexibility for the accommodation of alphanumeric, cartographic, and multimedia data.

BraDypUS stands as an application dedicated to the creation and management of archaeological databases online (Bogdani 2021). Originating from Julian Bogdani's initiatives in 2008 to develop bespoke web-based software solutions for the management of archaeological data, has since expanded its scope, managing, and providing services for significant Italian and international cultural heritage projects. Within the realm of digital humanities, BraDypUS emerges as a valuable tool, particularly in the field of archaeology. The software expedites the processes, methods, surveys, and documentation involved in excavations, often constrained by site-specific requirements. The WebGIS solutions integrated into BraDypUS offer practical assistance in inventorying, understanding, and networking cultural heritage within a given area (Bogdani 2019).

(P.R.)

From BraDypUS to PyArchInit and vice versa

Both pyArchInit (https://github.com/pyarchinit/pyarchinit) and BraDypUS (https://github.com/lab-archeologia-digitale/BraDypUS) are specifically tailored for archaeological applications. PyArchInit is predominantly developed in Python and relies on a predetermined database schema primarily designed for spatial data management. BraDypUS employs various web languages such as PHP, Javascript, and CSS, and boasts extensive versatility, offering complete customization for relational databases. The installation of BraDypUS is typically conducted on a web server and is accessible from any device running a modern web browser, while pyArchInit runs on a local machine and requires QGIS. In addressing the challenge of different structures, a resolution was achieved by establishing a connection between the two systems through the utilisation of a Spatial Relational Database Management System (RDBMS). Both applications are capable of interfacing with PostgreSQL and PostGIS, providing a collaborative solution for multiple individuals to concurrently engage in a shared project. Given the rigid database schema inherent in PyArchInit, the process necessitated constructing the database using PyArchInit's dedicated tools.

PyArchInit requires many tables to be created, each specifically designed to adhere to the stringent standards set by the Central Italian Institute for Cataloguing and Documentation (ICCD).

Given that both software packages were originally designed for distinct purposes and implemented using different programming languages, encountering incompatible components is inevitable. One prominent challenge lies in certain database conventions that hinder a straightforward and immediate connection between the systems. For instance:

- BraDypUS needs two mandatory fields (named "**id**" defined as the primary key, and "**creator**") to be added to every PyArchInit table to properly work;
- BraDypUS requires each table name to have a unique prefix separated by a double underscore;
- Yet, pyArchInit is very restrictive on the table naming system, and names such as '**site_table**' and '**US_table**' cannot be changed

To enable a connection between the two platforms, sharing a common database, we recommend the following procedural steps. As a premise, a PostgreSQL/PostGIS, PHP and Apache web server must be available; we will not cover here in detail the setup of a proper environment, since the BraDypUS official guide offers some support for this[1].

In QGIS, the pyArchInit plugin must be used to create and initialize the PostgreSQL database to strictly adhere to the required schema. Subsequently, the QGIS built-in function named DB Manager or an external tool (such as PgAdmin4) can be used to access the database generated by pyArchInit and add to each table the two columns required by BraDypUS, namely 'id' and 'creator'. Both columns should be configured as type INTEGER[2]. This is a critical step to align the PyArchInit database to BraDypUS. After these first steps, a new BraDypUS application can be created by pointing to the already-created database by selecting "pgsql" as the database type and entering the same connection credential used during the creation of the PostgreSQL database in pyArchInit[3]. Consistency between BraDypUS and pyArchInit data is crucial for establishing a reliable connection and ensuring seamless interoperability.

[1] https://docs.bdus.cloud/environment/setup-windows (accessed 20/12/2023)

[2] More information is available at: https://docs.bdus.cloud/conventions (accessed 20/12/2023)

[3] Further details and thorough instructions are available at the official BraDypUS documentation site: https://docs.bdus.cloud/create_app (accessed 20/12/2023).

To ensure easy access and manipulation of data within the BraDypUS environment, it is essential to meticulously create the configuration for each of the pyArchInit tables. BraDypUS integrates a graphical user interface (GUI) for this, which is simple and safe to use.[4] that will be stored in the "Project" folder of BraDypUS. To facilitate this process, we have made available a first draft of the configuration files required to run BraDypUS on a pyArchInit database schema.[5]

As a result, some aspects remain unsolved, the main one being the issue related to the primary key of each table. BraDypUS mandatory requires a field named "id" to act as a primary key, a rather diffused practice in relational databases. On the other hand, PyArchInit uses a proper, uncommon, way of naming the primary key, by prefixing "id_" to the name of the table (e.g., us.id_us, sito.id_sito, tafonomia.id_tafonomia, and so on).

pyArchInit and BraDypUS use also different methods for representing stratigraphic relationships. PyArchInit files all related contexts as an array in a single field, while BraDypUS employs related tables by taking full advantage of the relational model. In the first case, heavy application logic is required to grant data consistency and avoid recursive relationship, something that is solved in the second case by few database constraints that prevent, for example, do define twice a relationship between two contexts.

These are only few preliminary notes that we wanted to share with a broader public of users and developers, in the attempt to facilitate further integration between software solutions that are spreading rapidly in the Italian context, at least.

Finally, the database behind pyArchInit and BraDypUS seamlessly integrates with QField, a GIS software for mobile devices, facilitating the direct storage of archaeological data gathered during fieldwork (Bernasocchi 2022a, 2022b). QField empowers users to visualise and manage GIS projects on smartphones or tablets, leveraging QGIS project themes, labels, and styles, and is currently becoming a rather common practice in the archaeological domain (Montagnetti and Guarino 2021, Montagnetti and Rosati 2019).

(G.G.)

[4] https://docs.bdus.cloud/setup/

[5] https://github.com/Heryx/cfg_from_pyarchinit_to_ bradypus (accessed 20/12/2023).

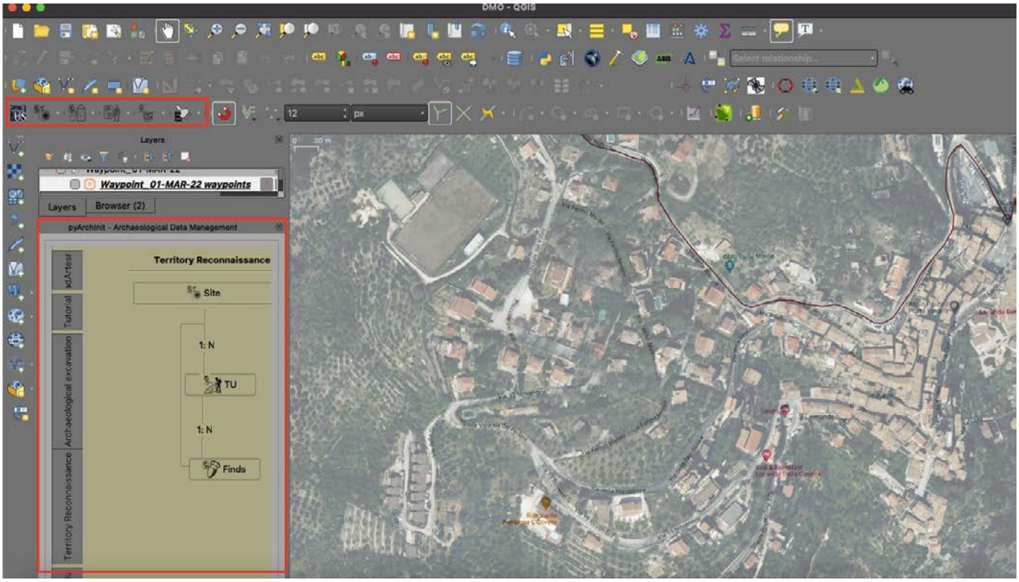
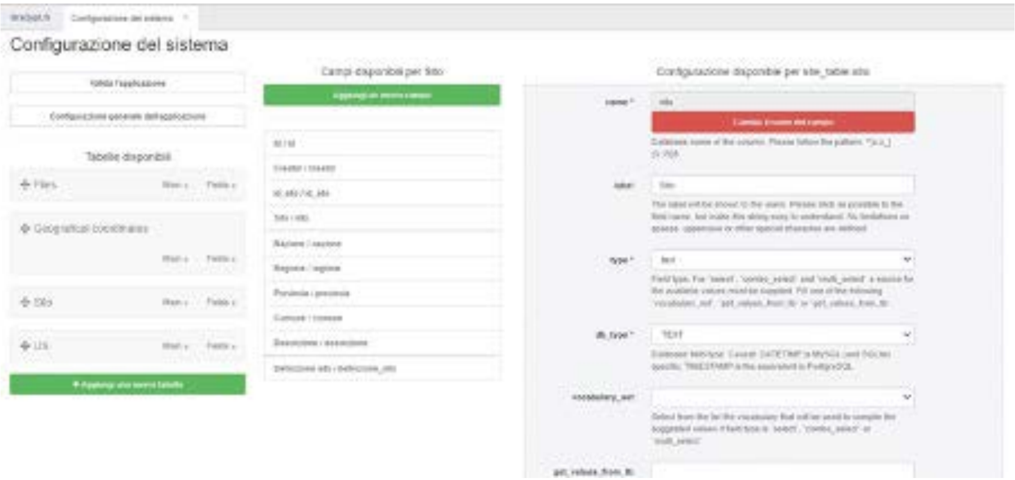


Figure 1. Pyarchinit on Mac, in the red squares the toolset of pyarchinit plugin and the presentation/shortcut bar under the layers panel.



Configurazione del sistema

Tabelle disponibili

Campi disponibili per Sito

Configurazione disponibile per site_table sito

name *: sito

label: Sito

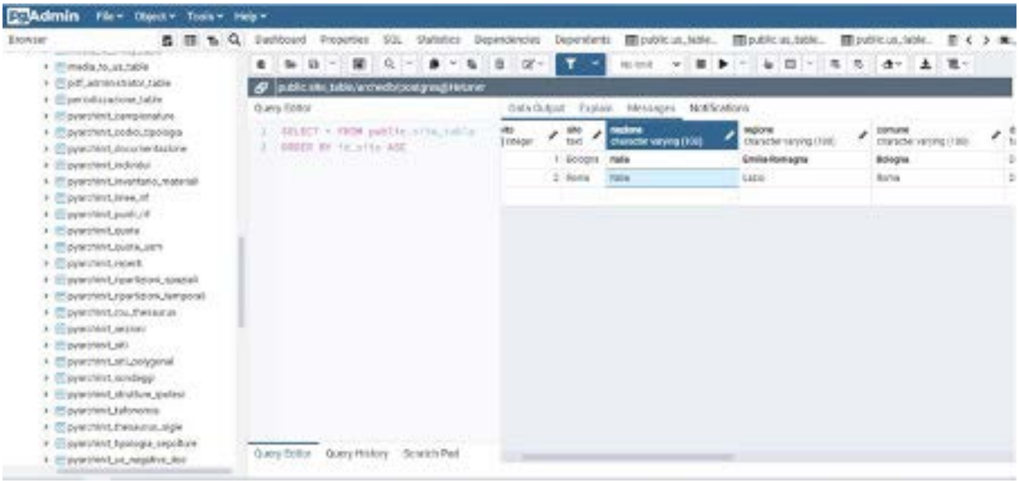
type *: text

db_type *: TEXT

vocabulary_set:

get_values_from_tb:

Figure 2. The BraDypUS configuration systems page



public.site_table/archedb/postgres@Helsinki

```
SELECT * FROM public.site_table
ORDER BY id_sito ASC
```

id_sito integer	sito text	nazione character varying (100)	regione character varying (100)	comune character varying (100)
1	Bologna	Italia	Emilia-Romagna	Bologna
2	Roma	Italia	Lazio	Roma

Figure 3. The pgmyadmin main page with the config folder

References

Bernasocchi M. 2022a. *QField* (version 1.10.00). https://github.com/opengisch/QField.

Bernasocchi M. 2022b. *Qfield Cloud* (version 0.13.2). https://github.com/opengisch/qfieldcloud.

Bogdani, J. 2019. “Archeologia e tecnologie di rete. Metodi, strumenti e risorse digitali”. in *I Cardini di Groma*, 4. Roma 1st ed.: BraDypUS Communicating Cultural Heritage

Bogdani, J. 2021. *BraDypUS* (version 4.1.2). https://github.com/lab-archeologia-digitale/BraDypUS.

Ducke B. 2012. “Natives of a Connected World: Free and Open Source Software in Archaeology”. In *World Archaeology*, 4 (44): 571–79.

Cocca E. and L. Mandolesi L. 2013. “Pyarchinit: gli sviluppi dopo Archeofoss 2009”. In *Archeologia e Calcolatori* Suppl. 4:128–38.

Montagnetti, R., and G. Guarino. 2021. “From Qgis to Qfield and Vice Versa: How the New Android Application Is Facilitating the Work of the Archaeologist in the Field” *Environmental Sciences Proceedings* 10, no. 1: 6. https://doi.org/10.3390/environsciproc2021010006

Mandolesi L. and E. Cocca. 2022. *PyArchInit* (version 3.9.12). https://github.com/pyarchinit/pyarchinit.

Montagnetti R. and P. Rosati. 2019. “Georiferire la stratigrafia archeologica”. *Archeologia e Calcolatori* 30: 463–66

Edwards, Benjamin and A.T. Wilson. “1 Open Archaeology: Definitions, Challenges and Context” In *Open Source Archaeology* edited by Andrew T. Wilson and Ben Edwards, 1-5. Warsaw, Poland: De Gruyter Open Poland, 2015. https://doi.org/10.1515/9783110440171-002