An Architecture and Protocol for Management of Multimodal Experimental Data

Ovidiu Gherman
MintViz Lab / MANSiD Research Center
Ştefan cel Mare University of Suceava
Suceava, Romania
ovidiug@eed.usv.ro

***Abstract*—Data collection for controlled experiments is a critical step in ensuring that the correct protocol is followed. This stage usually requires specialized devices and proper recorders. Moreover, the data – in different formats and from various sources - must be stored, processed, and used for various stages of the experiment. A good workflow for data management is important both for experiment's implementation and for historic preservation of the relevant information (for later analysis, meta-analysis, or various other uses). In this regard, the article proposes an experimental architecture that leverages the modern technologies to allow easy deployment of data acquisition tools in the field, the collection and classification of data and storage and custom retrieval of information for various purposes. The details of the platform will be further discussed, and its advantages highlighted.**

Keywords—microservices; containers; data aquisition; distributed architecture; experimental data

I. INTRODUCTION

Most of the experiments made today require the acquisition of a multitude of data types, to allow meaningful analysis and obtain significant conclusions [1], [2], [3].

Usually, this type of lab equipment is composed of a series of field sensors and a computing node (PC or laptop) for recording, storing information or on-demand processing. If the complexity of the recording equipment is high (high volume of data recorded, fast sampling times etc.) or the preprocessing steps are complex (de-noising, complex mathematical transforms) the computing node can fail or lag in executing all operations. This can lead to data corruption or may cause data loss [4] that will alter the quality of the acquired signals. Moreover, logistical problems can plague the equipment deployed in the filed: multiple sensors require enough ports on the PC/laptop in use (laptops are used in this role because of their portability, but usually lack a large enough number of USB ports – the most used type of connection) or the equipment can be distributed in a big area and requiring long, impractical cables or even multiple computing systems. Using multiple computing nodes solves most of the presented problems but will also introduce new problems such as desynchronization of data streams or failures not properly signaled towards the main processing node.

In this regard, to ensure data integrity and coherence, a series of steps can be taken:

- Data is synchronized and the acquisition devices will always be in sync between them. Subsequently, data will be processed correctly ensuring that the conclusions derived from the analysis are valid. This could be a problem if the acquisition device or various components of the platform is spread on multiple computing nodes.
- Monitoring services can run and validate individually each type of data from the multimodal dataset. Appropriate measures can be taken if the sensor malfunctions or the recordings are not correctly done or corrupted.
- Data can be centrally stored (at once or when an Internet connection is established, via a - potentially encrypted - syncing service) for immediate or latter processing.
- Data can be served historically for various tasks in the future, at once (the entire multimodal dataset) or segmented (only certain blocks can be made available/served). The hub can be queried to offer relevant data for certain following experiments/analysis. A key step is ensuring that multimodal data retrieved from a single experiment is synchronized (timestamps are a potential solution in this case).
- Stored data could be represented or visualized "as is" using the proper libraries. This allows to extract novel information from recorded dataset or to verify that the recorded information is of proper quality/valuable for experimenter. Quality checks can run automatically and detect data that is not in expected ranges (for example extreme or anomalous values that can be discarded from analysis).

978-1-7281-6870-8/20/$31.00 ©2020 IEEE

To improve this situation, a novel approach is proposed in this article: creating a platform that will allow easy management of deployed gear and of the data recorded. This platform will also offer features for later management and validation of acquired data.

II. The Proposed Setup

A. User requirements

In traditional approaches, the ecosystem of various physical tools and software platforms is either custom-build for a given task, or contains a generic equipment used in conjunction with predefined software, the workflow being supported - usually - manually. This approach is inefficient and prone to mistakes because of the various steps that must be taken in a predefined order. To avoid such pitfalls, a more sustainable approach is needed.

Usually, the recorded data is processed (error-checked, formatted, normalized, etc.) and then analyzed using various statistical and analytic tools (for example Matlab). Although the last step - the terminal analysis - cannot in many cases be completely automatized (the potential cases are hard to cover via automatic workflows in some cases), the previous steps can be controlled via automatic workflows that will ease operator's work in a significant degree.

Generally, users require some form of acquisition device (preferably based around consumer devices like laptops) that can collect relevant data via sensors and connected devices, pre-process it and then either require to manipulate it according to implemented algorithms or to be stored for further analysis (locally or after being sent to a remote site). Additionally, there is a problem for archiving data (with appropriate safeguards like encryption and timestamping), proper data access (for sensitive situations), data monitoring and auditing, visualization and other on-demand services for end-users that can be implemented.

A specific test case is found in VErGE [5], a multi-modal platform for collecting experimental biologic data. In the normal use-case, the data is acquired via various acquisition channels (in this case an EEG headset [6], an eye gaze recorder, a touch gesture acquisition device and an audio/video system) and stored as XML files and then manually uploaded on a central storage system. Although the system is built to be used in the field, by an operator that will run the experiment with volunteers – and so is designed to be easy to use – deploying the appropriate software stack (drivers, acquisition software, data recorders, backup software etc.) and maintaining it in use requires technical expertise. Moreover, the acquired data is processed completely manually using a dedicated processing pipeline based on Matlab and Java applications (the specifics of acquired data and the end product after data processing can be seen in Fig. 1), after the experiments are done. If the setup is deficient or the equipment does not work as intended, the experiment must be repeated because the validity of the data cannot be ascertained in a timely manner.

B. The Proposed Design

To respond to the user requirements, a modular approach is recommended. This way, the platform can be re-configured if needed to be a better match to the current experimental setup and various use-cases needed for the experiment.

The proposed architecture is described in detail in Fig. 2. As seen, the platform involves three main components:

- An ecosystem of containers (and the companion infrastructure for deployment) that will run bundles of software (libraries, interface modules and drivers) tailored for specific data acquisition equipment (for example various EEG headsets, microphones, eye gaze devices, heartbeat monitors etc.). The main advantage of the proposed architecture is that it allows pushing - via a deployment tool - the proper containers on the target laptop from a central control hub. In this regard, the device (equipment and laptop) becomes "plug'n'play" and the operator does not need to install various software packages, debug install problems or configure various devices; at most the drivers for the equipment must be deployed if unable to do so from the container. The correct container is selected from a dashboard and is pushed onto the devices. Once started, if the equipment is connected (usually on a USB port), the device will work as intended. The setup stage is reduced by a significant margin. This can be replicated for various devices - even if the said devices are installed on the same laptop. This component will be named from now on as "customer facing component".
- A cloud system based on Linux (for easy deployment and operation, considering that Linux-based system are ubiquitous in technologies involving cloud operations) that will employ various services (also container based) that can help the operator with (a) the management of containerized bundles of software ready to deploy on customer (field) devices and (b) the management of the acquired data (storage, analysis, various types of processing, plotting, etc.). In this regard, an programable API can be used to link the data repositories with various tools that can obtain desired information from relevant data, automatically (for example Octave can be used instead of Matlab as an automatic data processor for certain pre-programmed operations).
- A control system (dashboard) to allow the experimenter to interact with the entire system. The dashboard can be used to deploy container bundles to various targets (either customer devices or the micro services cloud), to load new modules/containers prepared by the developer or to visualize/plot/extract data from the historical archives. A monitoring system must be also employed to allow for automatic integrity checks and data validation. This kind of tool can offer periodic updates toward the experimenter, ensuring that the system and the stored data are in best condition.

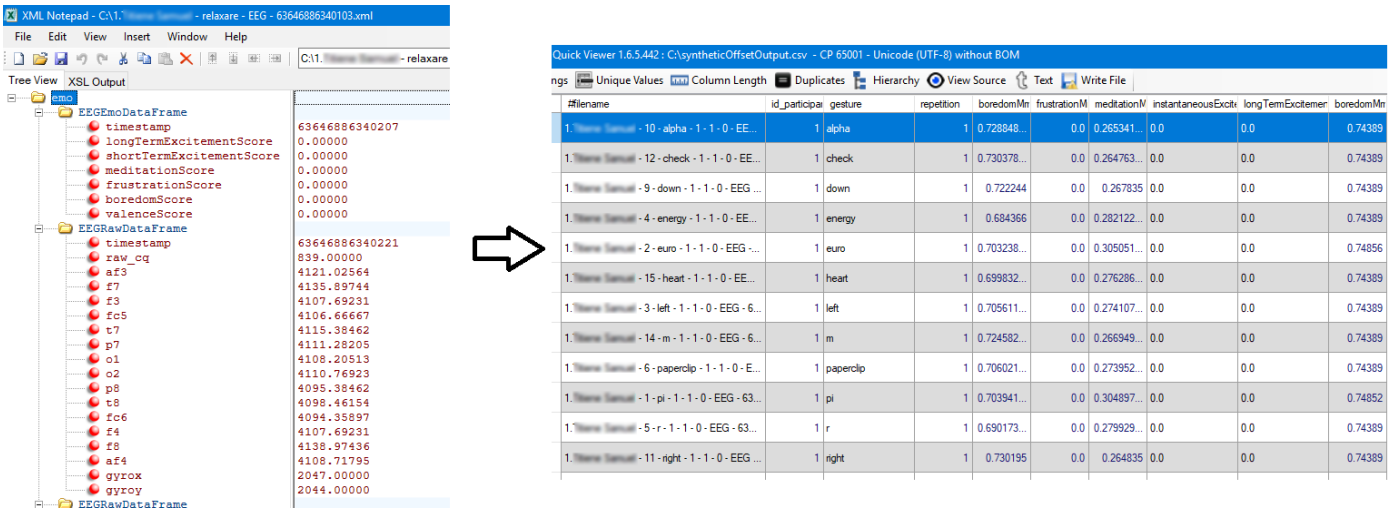

Fig. 1 – Acquired experimental data and the final processed data in VErGE experiment.

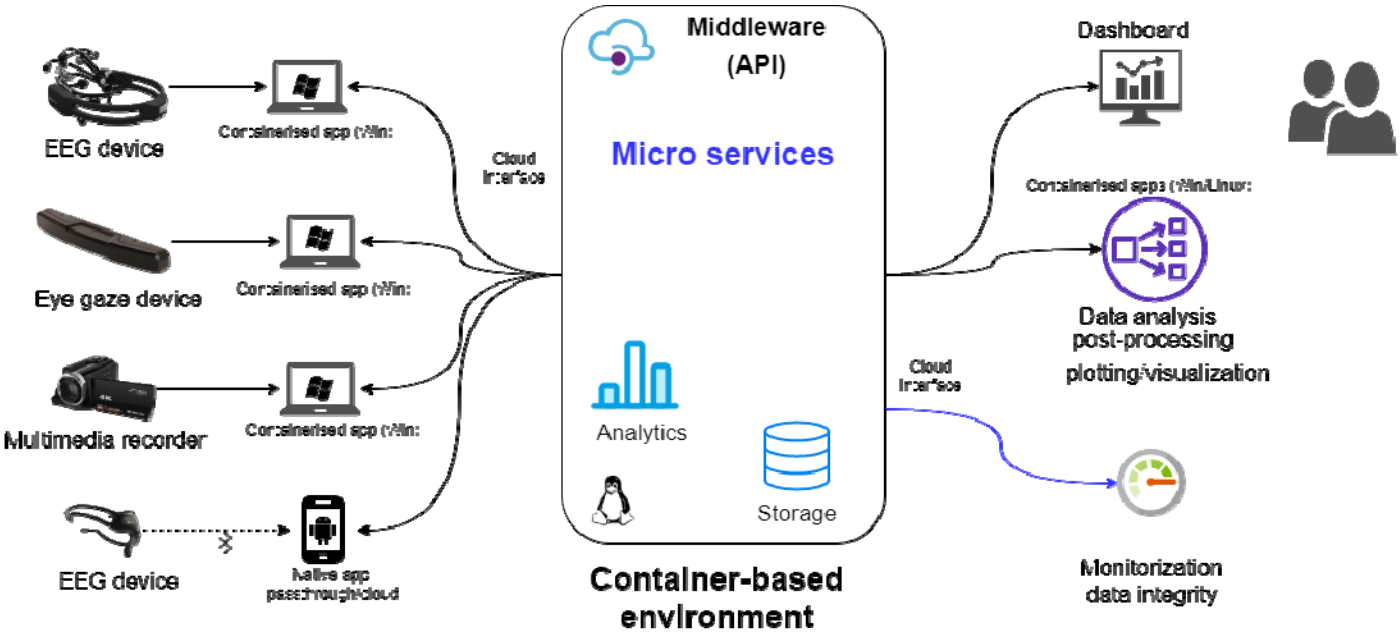

Fig. 2 – The proposed architecture for the management of multimodal data (presented: acquisition and control).

C. *Implementation Details*

The technology proposed for the platform design is based around software containers. This technology allows deploying bundles of software, configuration files and libraries with minimal expertise from the field operator, via automatic processes. The acquired data can be extracted and placed in a cloud system (also based on containers), from where can be archived and "served" via various micro services towards various applications for further analysis.

In this case, the main container technology proposed for implementation are Docker containers [7], [8]. Their ubiquity in industry, reliability and flexibility allows for easy integration in cross-platform architectures and a reduced cost of maintenance for the user. Also, Docker containers are very modular by design, allowing the development of various new modules (to respond to new data acquisition devices appearing on the market), fast bug fixing, fast deploy of various types of modules and other services. A big advantage is that this technology works on Windows and Linux (allowing deploying of Windows containers in the field devices – given that usually the acquisition equipment is made to run under Windows – but also deploying of Linux components for the cloud-based servers that will become the backend of the system).

For mobile devices of smartphone type (if necessary) a native component must be used – the container technology cannot run reliably on such devices. But given that an smartphone is a data acquisition device packed with useful sensors, certain amendments can be done to deploy custom-written packages to allow capture of relevant data and the transfer in the cloud component of the proposed platform. The best approach is to use Android-based devices, given their

popularity, relative low cost, and easiness of software deployment on them.

In the experimental approach to validate the basic premises of the proposed architecture was examined the feasibility of first-stage data acquisition. Usually this is the most problematic aspect of the design considering that interface with hardware is traditionally a difficult proposition for container technology [9], it being oriented more towards app hosting. In this regard it was examined the capability of a containerized application to ingest EEG data generated by an acquisition device (Epoc EMOTIV headset). This was implemented via binding a volume between the host and the containerized app. The result can be seen in Fig. 3.

D. Advantages

The implementation of the proposed platform will allow easy administration of the entire process - from developing auxiliary modules to improve the platform, to the deployment of containers from central repositories on the field devices. In this regard, the automatization of the entire process will require less technical expertise from the users, allowing a larger population of experiments to be conducted by various experimenters. Moreover, the development of new modules can be delegated towards interested – and capable – researchers/developers (for example in an open-source model), new configured containers being reusable for a given equipment. Fig. 4 presents the proposed deployment workflow for the platform.

E. Limitations

There are as series of limitations identified in the proposed architecture. The most important ones are discussed in this section.

The usage of specialty equipment: if the experiment requires the use of an smartphone (given the value of its sensor pack this is a viable option for a large spectrum of experiments), special accommodation will be required for the relevant software. Given the nimble nature of the hardware, the

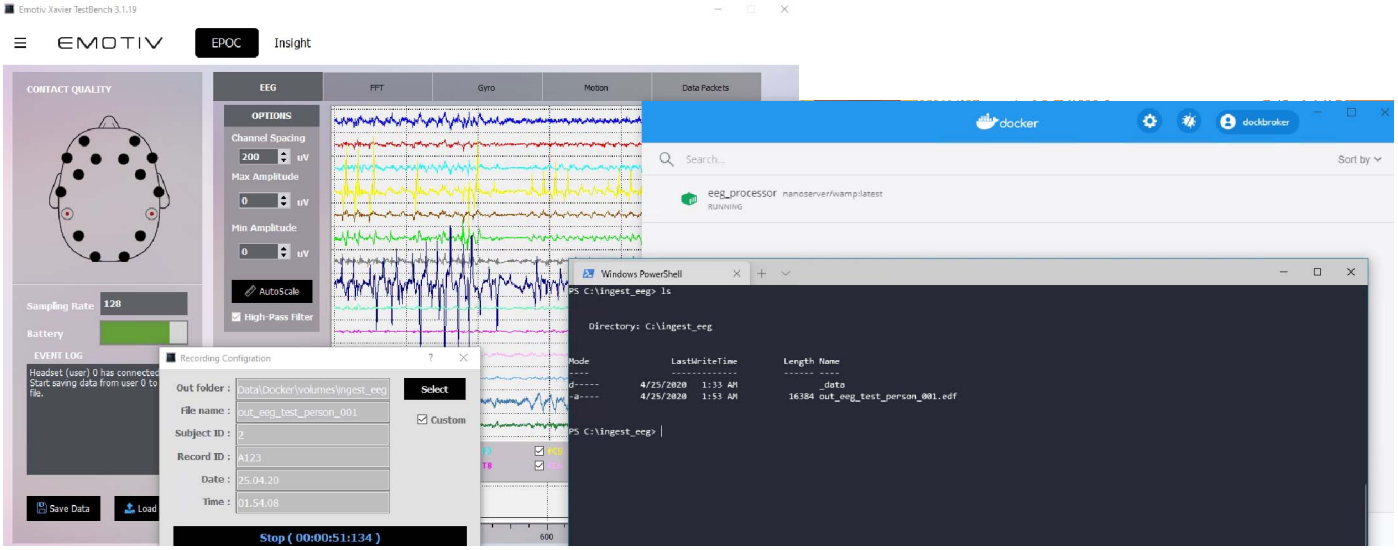

Fig. 3 - EEG data ingestion in the container via a bounded Docker volume.

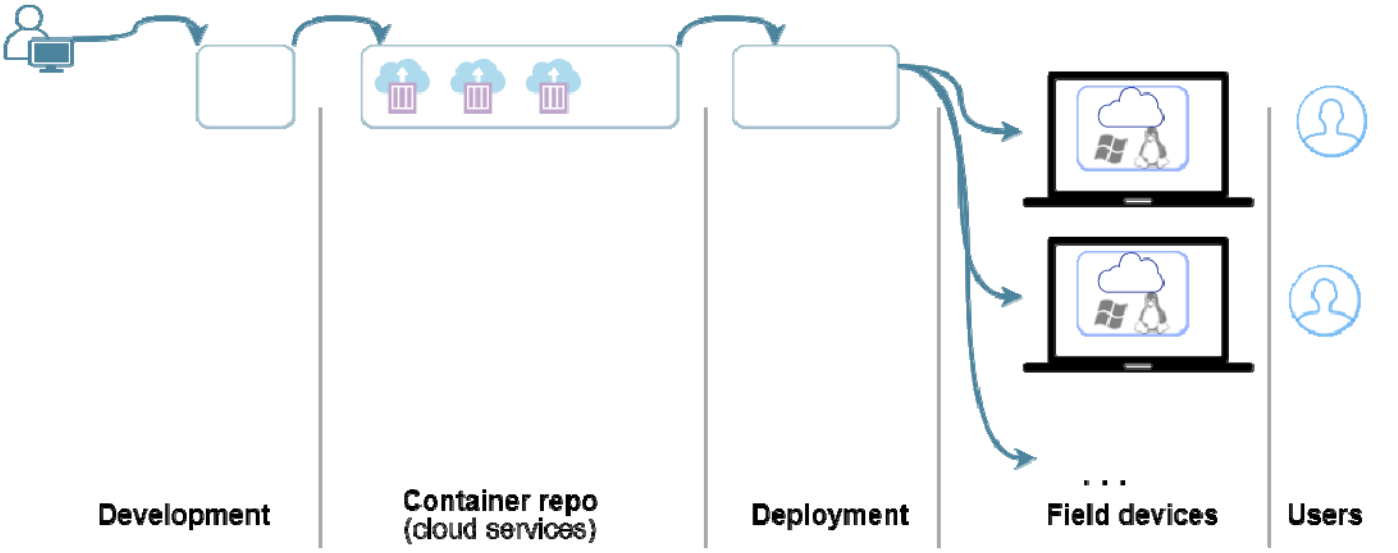

Fig. 4 - Management workflow for the proposed platform.

customized hardware architecture and the closed software ecosystem, running Docker containers (or any kind of containers) on mobile phone is not possible yet. Even with the (relative) openness of the Android platform, running customized software bundles is not generally possible (although there are methods to try this via custom ROMs). In this case, native application must be developed. The management and orchestration of these should not be a problem though (various deployment mechanisms can be used, including using official app store).

Also, the programming effort for maintaining the platform can be significant if a massive number of acquisition devices is intended to be supported by the main application. For each new device, the driver, the main library and supporting software must be installed in a container, configured, and tested for optimal usage. Once done, the container can be deployed seamless on every target device.

Furthermore, data communication of data between the customer facing machines and cloud infrastructure can hold biological data acquired from volunteers. As such, special measures must be employed to ensure the privacy of the data, given that there can be legal consequences if the data is intercepted by unauthorized parties. In this regard, using a well-established cryptography library can solve the problem.

III. CONCLUSIONS

The proposed architecture was designed to allow efficient collection of experimental data from various field equipment, allowing for easy deployment of relevant software (as to ease the operator's job) using containers as the main way of deployment. Moreover, the architecture allows for efficient storage of data and offers various micro services towards the end users (the experimenters) via dedicated interfaces for data extraction, analysis and other forms of post-processing.

The final goal is to fully implement the architecture using the experimental VErGE platform as to allow the acquisition of multimodal biological data for experiments. This testbed will be used to validate the proposed model and improve it – if necessary. The final product will allow easy configuration of diverse types of equipment used to acquire data from different fields of work.

Further work will be concentrated on the implementation of the cloud backend for management of containers and of the acquired data. To evaluate the validity of the architecture and to develop the platform, a current experimental system (VErGE) will be used, it's acquisition modules being employed for the customer-facing component. In this way, the complete porting of the functional modules from VErGE will allow for a direct comparation of the old platform with the new one, regarding speed, ease of use and the experience of the terminal operator. Moreover, test feedback can be used to develop new features for the platform or to adjust components that are not functioning as intended or with reduced performance (compared with the old platform).

ACKNOWLEDGMENT

This work was supported by Machine Intelligence and Information Visualization Lab (MintViz) / MANSiD Research Center.

REFERENCES

[1] P. Jermann, D. Gergle, R. Bednarik, and S. Brennan, "Duet 2012: Dual eye tracking in CSCW"", in Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work Companion (CSCW '12), ACM, 2012, pp. 23-24.

[2] C.A. Chin, A. Barreto, G. Cremades, and M. Adjouadi, "Performance analysis of an integrated eye gaze tracking / electromyogram cursor control system", in Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '07), ACM, 2007, pp. 233-234.

[3] D. W. Hansen and Q. Ji, "In the eye of the beholder: A survey of models for eyes and gaze", in IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, iss. 3, 2010, pp. 478-500.

[4] B. Charyyev, A. Alhussen, H. Sapkota, E. Pouyoul, M. H. Gunes, and E. Arslan, "Towards Securing Data Transfers Against Silent Data Corruption", 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Larnaca, Cyprus, 2019, pp. 262-271.

[5] O. Gherman, O. Schipor, and B. Gheran, "VErGE: A system for collecting voice, eye gaze, gesture, and EEG data for experimental studies", 2018 International Conference on Development and Application Systems (DAS), Suceava, 2018, pp. 150-155.

[6] K. Holewa and A. Nawrocka, "Emotiv EPOC neuroheadset in brain-computer interface", Proceedings of the 2014 15th International Carpathian Control Conference (ICCC), Velke Karlovice, 2014, pp. 149-152.

[7] M. Abdelbaky, J. Diaz-Montes, M. Parashar, M. Unuvar, and M. Steinder, "Docker containers across multiple clouds and data centers", 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC), Limassol, 2015, pp. 368-371.

[8] T. Combe, A. Martin, and R. Di Pietro, "To Docker or not to Docker: A security perspective", in IEEE Cloud Computing, vol. 3, no. 5, pp. 54-62, Sept.-Oct. 2016.

[9] "Devices in containers on windows", Microsoft documentation, Aug. 2019. [Online]. Available: https://docs.microsoft.com/en-us/virtualization/windowscontainers/deploy-containers/hardware-devices-in-containers.