Below is a text I proposed for a contribution to the “Common White Paper” (CWP) of the HEP Software Foundation (HSF) about visualisation for the paragraph on “Mobile devices”. After it there is the revisited version of the other contributors that want to enforce a client/server architecture and a coworking new file format for the future of HEP visualisation. (A point of view that, halas, gives a blank cheque to the whole HEP to not doing the needed work to rethink event and detector models and then data access in general. Following this revision, and since the final text is no more mine, I withdraw my name as a contributor for this CWP).

My text :
=======

First, "Mobile devices" wording, than came from 2007/2010 with the arrival of the smartphones and tablets, is perhaps no more adapted ten years later since the technology evolved with devices as the ultrabooks, 2-in-1 and convertible PCs that blur the border between smartphones and more classical laptops, especially concerning programming. Relative to HEP computing activity it is more adequate to speak now of "devices close to people" to put in contrast, or even in opposition, to data centers that are remote to people and highly immovable. Force to state that a lot of interactive and graphics power are now in the hands of people and on their head if we include also the VR devices in this category. Visualisation must be done here, close to people, if we want to build very reactive interactive environments. The time of graphics done with X-terminals connected to the remote mainframe is over. And here HEP has a technological problem related to its data and to its today software. Force to state that right now (2017) the HEP software is targeted for the "batch" to run in data centers where the data had been deposited. Force to state that not only the data are not so easily embarkable on local devices, even if having now local resources to host hundred or thousand of events, but, worst, that the software to read them can't be embarked at all on most local close-to-people devices around. Force to state also that people interested in doing visualisation fall on a strong "batch" sociology very reluctant to consider to do deep changes in their software in order to cope with all these new interactive technologies. HEP has a sociological problem here. Clearly, at the level of the software, if we continue to debate around "frameworks" there is no way out. We must find first the right angle to attack this problem. We propose to concentrate on the lonely thing that must and can be shared between an event display and a batch for an analysis; mainly the event and the detector models. Clearly an event display do not need all what is put in a "framework" for a batch for an analysis, but only the core part that contains the event and detector models along an IO package to read an event in a file. If we have that in head as a guideline, there is a way to ask for modifications in today codes in order to isolate these parts and have them highly portable in order that vis people can be at work by enjoying all the technologies around.

Due to the "batch oriented framework" blocking, some propose that the visualisation effort stays only at the level of "viewer" local programs, mainly programs able to view or render a bunch of primitives deposited in a file, or send through the net, but having been produced by detached programs built onto the batch framework and running on the remote and immovable batch devices. Even if "working", it would be shortsighted to continue in this way and reduce visualisation to that; a "viewer" does not permit to interact back into the data in a highly reactive way (it is the "picking" problem). If we define an event display as being a viewer straight over the event and detector models, then only an event display permits to reach this kind of interactivity. We must then arrange HEP software in order that these core parts be shared between the batch and the interactive. If we define the problem by using this "event and detector models" wording, and then avoid the word "framework" (which is, halas, today de facto tied to the batch and finishes to stress everybody) we may have a way out to organise things in order that everybody be happy; the batchers in producing histos with xillions of events on their remote, big and immovable devices, and the vizzers in navigating in a detector with a single event with their local, light and mobile devices.

To sum up; the "Mobile devices", or devices close to people, put the finger on a today painful problem in HEP computing; the data access, and it would be shortsighted to ignore this problem and this trend of technology. It is interesting to note that other areas of science, for example astronomy, had been much more reactive and that a lot of "apps" of great quality, often done by hobbyists, can be found on the iOS and Android application stores; and this success is clearly related to the fact that their file format is rather universal and data are easily accessible; HEP should draw inspiration from them here.

CWP text (at date of November three 2017) :
==========
Nowadays mobile technology is more and more ubiquitous, people having access to a plethora of mobile devices: from tablets to smartphones to ultra-books. Those devices are used more and more as substitutes of desktop and laptop machines in people's daily life.

Even if the hardware embarked by these devices largely evolves with each generation, mobile devices still do not have the computing power usually needed for HEP data analysis, where huge amount of experimental data are retrieved and processed. In addition to that, they usually run dedicated operating systems, whose self-contained nature makes their integration within the HEP workflow difficult, particularly for the statistical-based visualisation used in data analysis [ref to section].

But portability and simplicity of usage are the strong points of mobile devices. More than as "mobile" devices, smartphones, tablets and ultrabooks should be considered, in fact, as "devices close to people". And because of that, the usage of such devices should be exploited more in the final steps of the visualisation chain, where

heavy batch data processing is not needed. For instance, their usage should be leveraged for the production and visualisation of event displays.

Ideally, a user should be able to easily retrieve interesting events from the experiment and interactively visualise them on all kinds of devices.

Instead, currently, event visualisation on mobile devices is only possible in experiments which developed web-based tools [ref to section]; and only the visualisation of events which have been already extracted and reduced from the experiment's framework is possible, as like as with the standalone visualisation tools [ref to section]; without the possibility of having real interactive visualisations, de facto reducing the visualisation program to a mere "viewer".

That is why we strongly promote the usage of the server-client architecture described and supported in this paper [ref to section] and the new data access patterns presented and supported in the "Data Access and Management" paper [ref to paper]. That would open up new possibilities for interactive visualisation on mobile devices: it would let visualisation clients running on mobile devices connect to server tools running in the experiment's framework to easily and interactively retrieve the desired data.

It is worth noting that in other areas of science --- for instance, in astronomy --- researchers have worked to facilitate data access and to migrate to more standard data formats already. And that was beneficial to the possibility of having data visualisation tools on mobile devices, in addition to desktop and laptop machines. And that not only helped the researchers easily accessing and visualising their data, but it also paid out in making science accessible by the public, having eased the development of programs used in Outreach and Education activities and events. It is true that HEP data are usually much more complex than astronomy data, and so it will be harder to achieve, but we think that an effort in simplifying the access to experimental data would be worth anyway.

Therefore, the leverage of the usage of mobile devices in HEP adds a strong point to the development and the support of common client-server tools and data exchange formats [ref to paper] among HEP experiments in the near future.

There are at least a couple of examples of HEP applications developed for mobile platforms or that can be run on them. LHSee [LHSee] was a mobile application which live streamed Atlantis events to a user’s phone and provided contextual information on ATLAS and the events being displayed. The CMS iSpy WebGL-based application runs on mobile devices in the browser and users can interact with the visualisation with touch events; there is also a Google Cardboard mode to exploit Virtual Reality (more on that in section … [add ref]). The Camelia application [CERNCamelia], developed by the CERN Media Lab using the Unity game engine, can be run on mobile

devices as well. “More than ALICE” is an Augmented Reality application developed in Unity, allowing to superimpose detectors description or event visualisation of the camera image of the ALICE detector or its paper model.