REPORT

Beta Release

- project deliverable 6.3

Authors: Carlos Alberto Martín, Agustín Hernández, Ignacio Domínguez, Juan Carrasco, Aitor Vélez, Francesco D'Andria (ATOS);

Confidentiality: *Public*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825585.

Beta Release	
Project name HELIOS	Grant agreement # 825585
Author(s) Carlos Alberto Martín, Agustín Hernández, Ignacio Domínguez, Juan Carrasco, Aitor Vélez, Francesco D'Andria (ATOS)	Pages 2+17
Reviewers Vasiliki Gkatziaki (CERTH), Jarkko Kuusijärvi, Ville Ollikainen (VTT)	
Keywords Integration strategy and planning; DevOps, Continuous Integration, Software Release	Deliverable identification D6.3
Summary This report summarises the beta release (HELIOS deliverable D6.3). The deliverable, which is a demonstrator, includes also the code generated for the different HELIOS blocks, the AAR or JAR libraries and two HELIOS applications, as result of the building process. In this sense, the beta release is a set of HELIOS components, ready to be used by developers to create HELIOS apps.	
Confidentiality	Public
Madrid, Spain 25.6.2020 Written by Carlos Alberto Martín	
Contact address Ville Ollikainen, ville.ollikainen@vtt.fi, +358 400 841116	
Distribution HELIOS project partners, subcontractors, the Project Officer, public	

Contents

List of Figures

List of Acronyms

Acronym	Description
APK	Android Application Package
CD	Continuous Delivery
CITDM	Continuous Integration, Testing and Deployment Methodology
CI	Continuous Integration
CVS	Concurrent Versions System
DevOps	DEVelopment OPerationS
P2P	Peer-to-Peer
QoS	Quality of Service
WP2	HELIOS activity Work-Package 2 “Concept Design”
WP3	HELIOS activity Work-Package 3 “System development”
WP4	HELIOS activity Work-Package 4 “Social Networks Layers and Analysis”
WP5	HELIOS activity Work-Package 5 “Services and user interfaces”
WP6	HELIOS activity Work-Package 6 “System integration and operation”

1 Introduction

This short report summarises and introduces the HELIOS deliverable D6.3 (beta release), whose type is demonstrator. The beta release is public and offers a complete set of HELIOS modules, enabling key HELIOS functionalities. The release candidate (D6.4) is planned for M28 and will include consolidated versions of all the HELIOS components.

The compressed file of this deliverable includes four different items:

- #1 This short introductory report
- #2 A compressed file that contains the code developed in HELIOS by the moment and integrated in the beta release, following the distribution in repositories of the HELIOS GitLab group (master branches). Section 3 provides more information about these repositories.
- #3 A compressed file that contains the AAR or JAR libraries generated during the building process and using the continuous integration tools deployed in HELIOS.
- #4 Two APK files, created from the mentioned repositories. These applications are test HELIOS apps created in the project: TestClient and MediaStreaming.

Due to its size, items #2 and #4 cannot be submitted by means of the European Commission deliverable interface. For this reason, they are available in a machine used by the consortium for integration purposes under this URL: https://builder.helios-social.eu/ec/ . The authentication information is:
Username: **
Password: **********

After this introduction, section 2 explains how integration tools are operative; section 3 describes the architectural blocks that have focused the effort for this beta release and their links to the HELIOS repositories; section 4 and section 5 are dedicated to conclusions and references, respectively.

According to the work plan, the components of the beta release will be available at this public GitHub repository at the end of M18: https://github.com/orgs/helios-h2020 (this URL was already included in D6.1).

Due to the Continuous Development / Continuous Integration / Continuous Delivery (CD/CI/CD) approach of HELIOS, this beta release must be interpreted as a snapshot of the current versions at the time of finishing this deliverable. The development and integration activities will continue immediately after submitting it.

2 Availability of integration tools and methodology

Deliverable D6.2 [1] already detailed information about integration tools used for the alpha release, as planned in the deliverable D6.1 [2], such as:

- Jira, to register and monitor de implementation activities.
- GitLab, as collaborative implementation tool and issue tracker
- Slack, to enable a quick and efficient communication between partners.

For the beta release, the integration effort has been focused on the deployment and configuration of additional tools, as described in this section. Particularly, WP6 has deployed:

- Jenkins
- Nexus

Jenkins is an open source tool for continuous integration. Jenkins enables the access to the GitLab repositories and the automatic building, based on the configuration files included in the respective repositories. Since GitLab is the tool used in HELIOS for content repository and collaborative development, Jenkins obtains the source code there. Once the partner responsible for a HELIOS module has pushed or merged code in the master branch, the Jenkins web interface enables the generation of the respective library or apk. The configuration of this process for each HELIOS module (or repository) is made by means of a Jenkins job. Figure 1 shows the list of jobs configured in Jenkins. It was taken during the execution of the integration activities for the beta release and it shows different levels of stability for the different components or apps (weather icon on the left side).

As partner responsible for integration, ATOS is providing the GitLab repository. The solution selected in the project has been the creation of a HELIOS GitLab group, in order to have different projects / repositories inside, according to the modular approach of the project. In this way, it exists one GitLab repository for each architectural block. Later, in section 3, we identify the correspondence between architecture blocks and repositories.

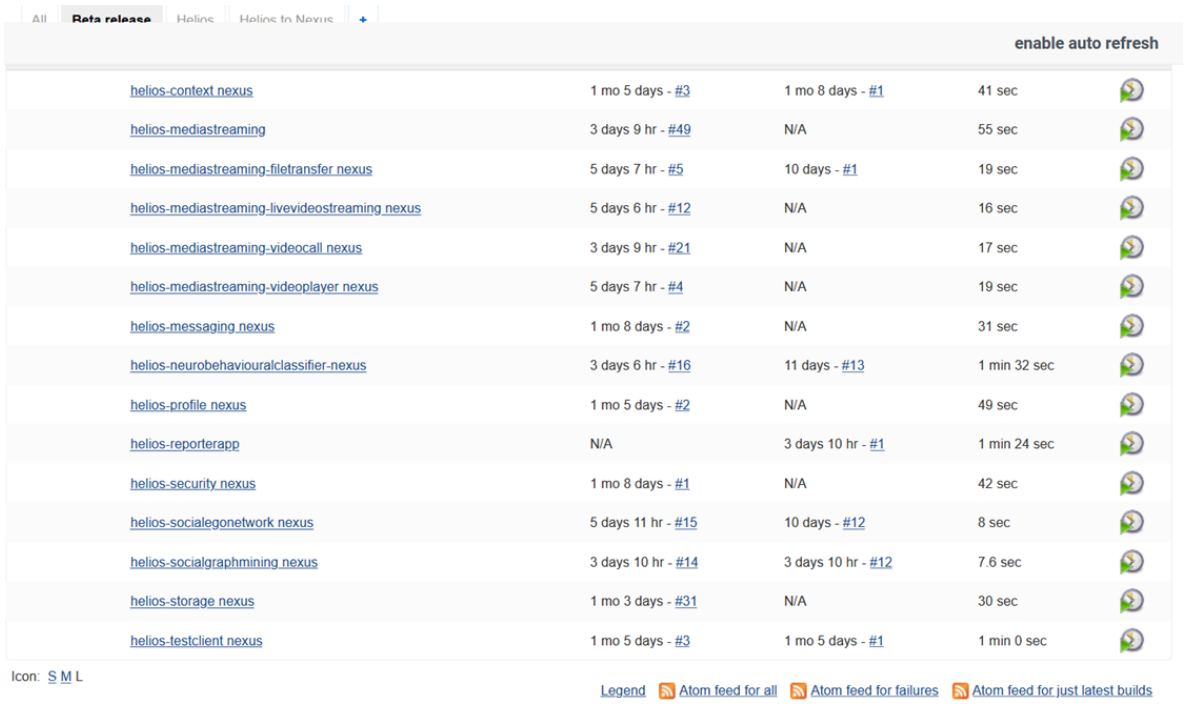

Figure 1. View of Jenkins jobs configured to build HELIOS components and apps

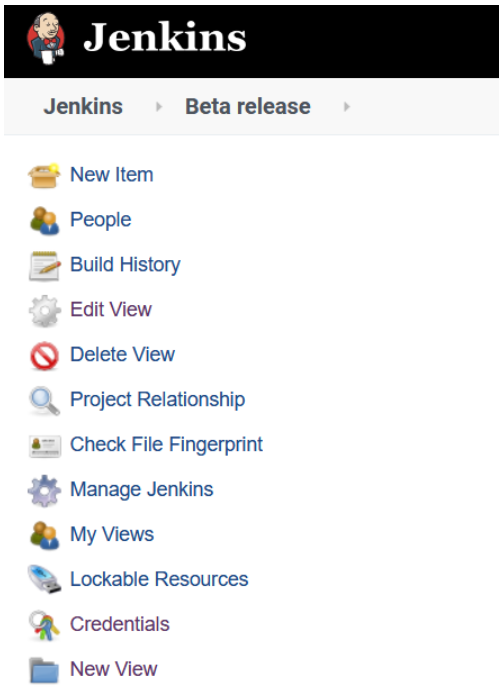

Figure 2. Jenkins configuration options

Figure 2 shows the general configuration menu offered by the Jenkins web interface, including new jobs configuration, view edition or credentials (e.g., to access the GitLab repositories or the Nexus explained later). Figure 3 depicts the information once a job is selected. In this case, the videocall feature, included in the media streaming extension module. Although media streaming is a block in the HELIOS architecture, during the T3.3 execution we preferred to extract the functionalities to produce a more modular platform. From the interface shown in the figure 3, a Jenkins user can configure the job (e.g., GitLab repository where the code is located, credentials to access GitLab, Nexus repository, etc.) The shown interface also offers information about the building process and the last buildings and it enables the download of the generated files. There are two different cases:

- The GitLab repository is a HELIOS component, as in this example of the videocall. Then, the compilation and building process creates a library, in AAR format (Android library format, suitable for components that includes activity and layout) or JAR format (Java format for libraries). Then the resulting library is copied in the Nexus repository. In this way, another module or HELIOS App (HApp) using the compiled component only needs to reference the library in Nexus, instead of replicating locally the code. In other words, the Nexus repository has been introduced in the continuous integration workflow to manage the dependencies and to optimise the workflow, as HELIOS is a collaborative and modular project, combining developments of different partners.

- The GitLab repository is a HApp (TestClient, MediaStreaming, Reporter App). In that case, the result of the process is an APK file, ready to be installed on a mobile phone. The APK files are made available to download via a Nginx web server running on the same machine. To notify the end of the process, Jenkins automatically sends a message to a specified email address and registers the result in the # jenkins channel of the Slack HELIOS room (figure 4).

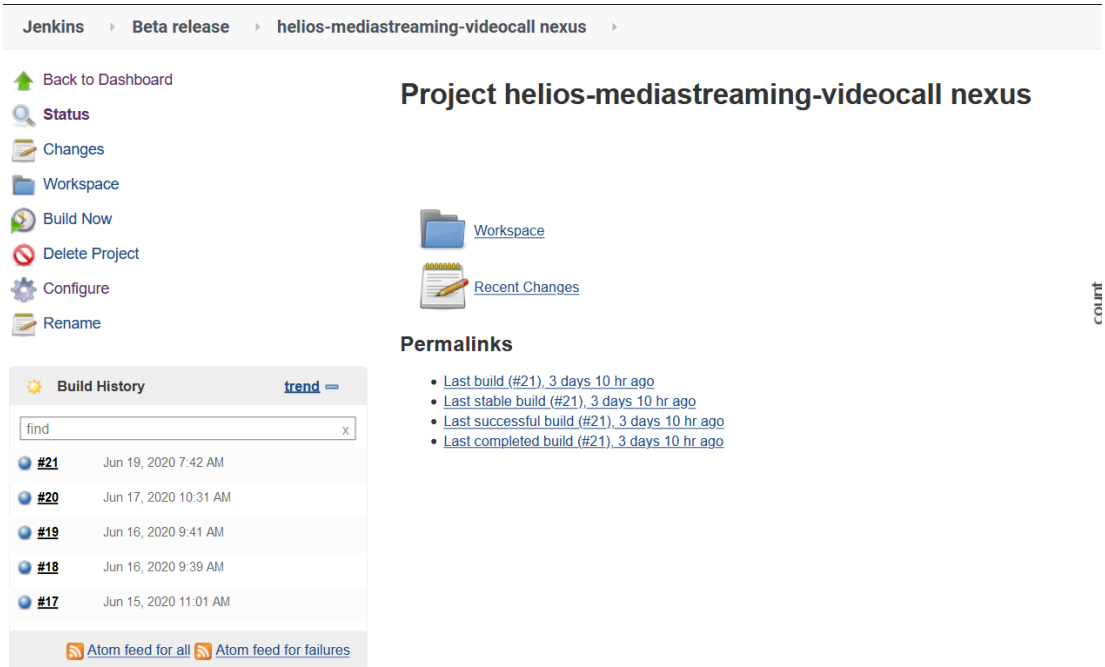

Figure 3. Jenkins interface for the administration of a job

Figure 4 shows the example of the Slack notification for the Media Streaming app, after being built integrating the videocall. The videocall is an AAR library available in the Nexus repository.

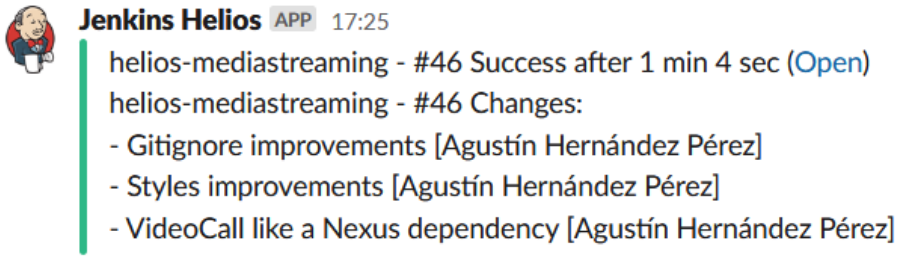

Figure 4. Jenkins notification on a Slack channel

For both cases (libraries or apps), the result of the building process is available at the Jenkins interface, including the libraries or APKs generated during the last compilation processes, as shown in the figure 3. In this sense, one of the advantages of Jenkins is the capability to track the building process.

The AAR or JAR libraries are available to be integrated in other Android projects via Nexus. Moreover, Nexus offers a web interface. Figure 5 shows the tree of HELIOS components available from Nexus and the details for the example of the videocall, including the AAR library and metadata. Nexus also enables the version tracking (e.g., versions 1.0.20 and 1.0.21 are available) and the dependencies on a concrete version.

The set of libraries included in this deliverable (item #3 in the list of the introduction section) are distributed in a replica of the directory tree shown in Figure 5. We have included only the latest versions.

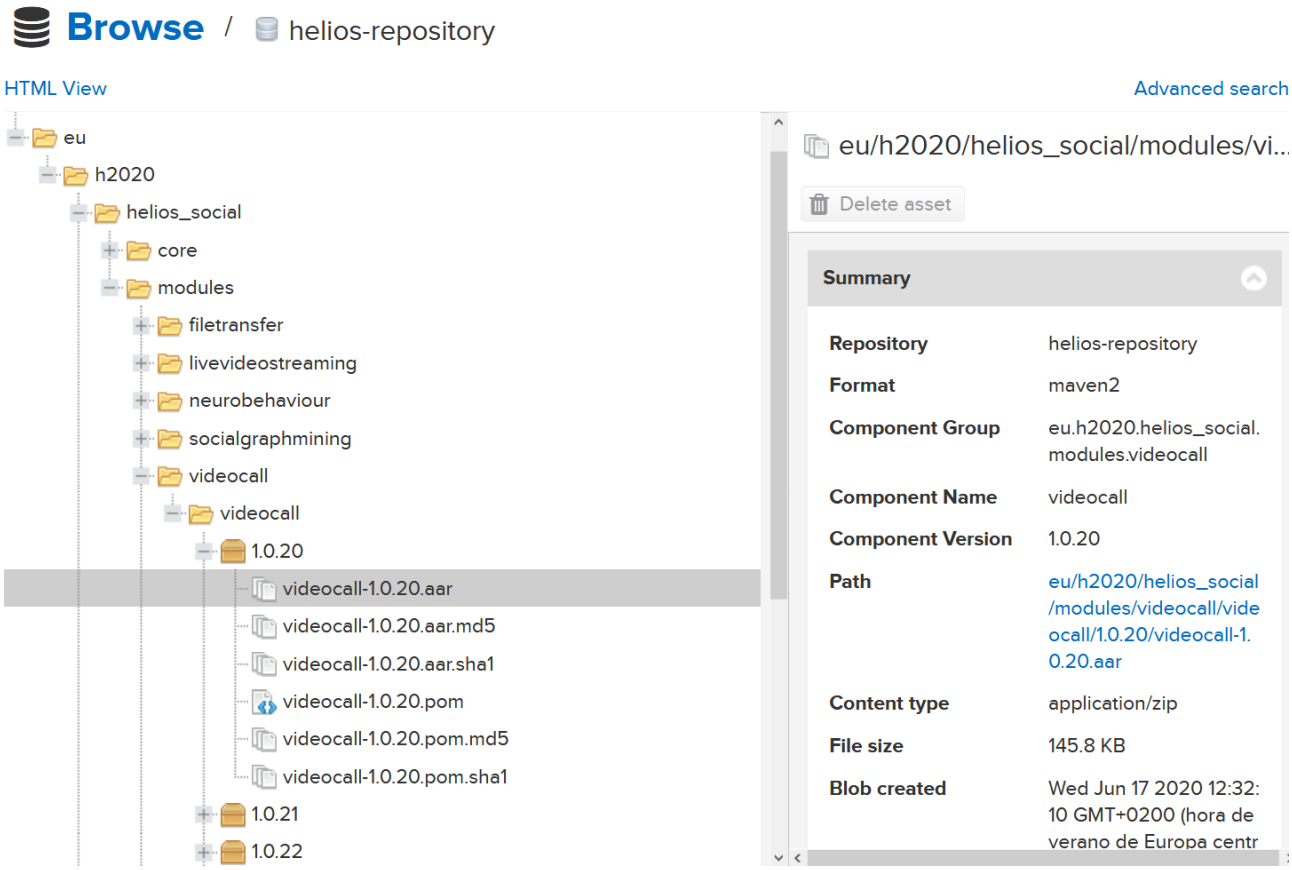

Figure 5. Web view of the HELIOS libraries available via Nexus

Figure 6 shows the repositories created inside the HELIOS GitLab project. Due to ATOS internal reasons, the GitLab server containing the HELIOS group has changed since the D6.2. The new server is found at this URL: https://scm.atosresearch.eu/ari/helios_group/

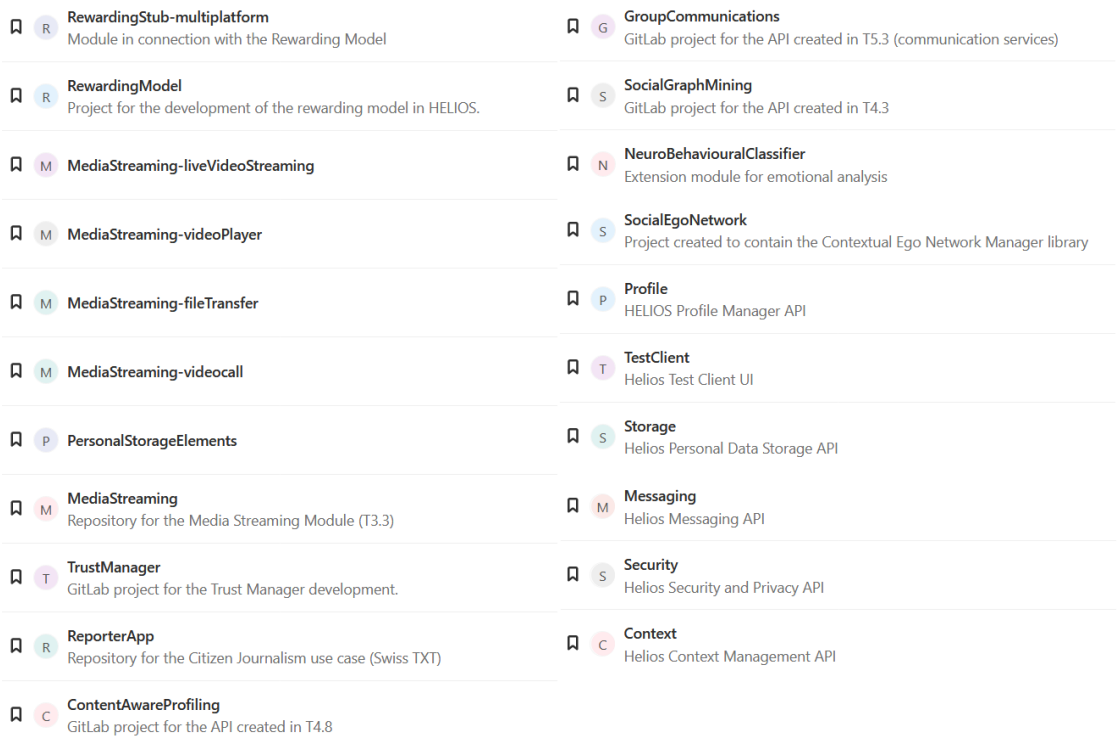

Figure 6. Existing projects/repositories in the HELIOS GitLab group

Since each HELIOS component is usually maintained by a certain consortium partner according to the work plan (e.g., UH is in charge of Security) but other partners may need to contribute (e.g., for interoperability and joint functionality purposes), some developing guidelines have been applied before the beta release. Particularly:

- The master branch of each repository is protected and only maintainers can push code or merge.
- The partners responsible for each repository (HELIOS component) has the role of maintainers. The rest of partners are developers.
- If a partner needs to introduce modification, she/ he creates a development branch and sends a merge request to include the changes on the master branch.
- The partner in charge of the module can accept the request. In this way, we ensure control and consistency in the GitLab projects.

3 HELIOS blocks integrated in the Beta Release

The HELIOS architecture is described in deliverable D3.2 ("Final system architecture and API specification"). Deliverable D6.1 ("Integration strategy and planning") [2] contains a summary of a

previous version of the architecture. The architecture fulfils a modular approach. The three logical layers of the architecture are:

1. HELIOS core
2. HELIOS extension modules
3. HELIOS applications or HELIOS apps.

Deliverable D3.1 [3] provides more information about these layers.

The logical modular architecture is depicted in the Figure 7, extracted from a draft version of D3.2 (D3.2 will be submitted at the same time of the present deliverable). D3.2 also considers the HELIOS architectural options, distinguishing if the HApps includes the HELIOS core and the HELIOS extension modules or if the HELIOS core and/or the HELIOS extension modules are installed separately.

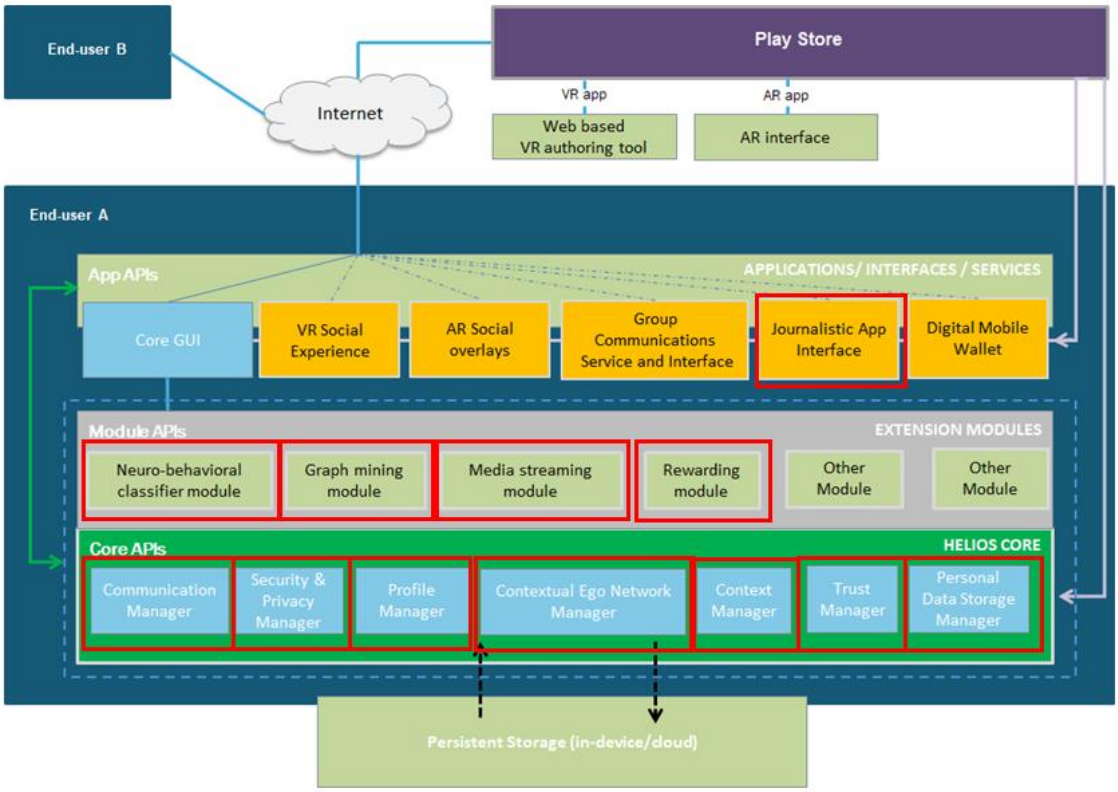

Figure 7. HELIOS logical architecture and modules implemented before the beta release

Figure 7 represents also the blocks that have focused implementation effort during the last months, before the beta release and that are using the integration tools provided in the project (red rectangular contours). Thus, the implementation effort has been concentrated on these HELIOS blocks:

HELIOS core modules

- Communication Manager (the respective repository is Messaging in the list of GitLab projects of section 2 - Figure 6). HELIOS Messaging API is used to communicate with other Helios nodes. The current implementation provides methods to facilitate the process of connectivity and messaging between peers. Additionally, the consortium has created another Messaging repository (named Messaging-nodejslibp2p) that contains a new implementation based on libp2p.

- Security & Privacy Manager (the respective repository is Security in the list of GitLab projects of section 2 - Figure 6). Security and Privacy API provides cryptographic services to HELIOS components.

- Profile Manager (the respective repository is Profile in the list of GitLab projects of section 2 - Figure 6). The Profile Manager handles user account creation and creation of associated keys. At least one key pair is needed for signing and verification of electronic signatures and another public key - private key pair is needed for encryption and decryption. Other keys are needed for encrypting contents the user is producing and/or sending to other users. Uses Identity manager and personal data storage manager to handle profile information control, as well. This manager also interfaces with the functionality offered by, e.g., Security & Privacy Manager.

- Contextual Ego Network Manager (the respective repository is SocialEgoNetwork in the list of GitLab projects of section 2 - Figure 6). This manager consists of a library to create and store the structure to keep the contextual network information.

- Context Manager (the respective repository is Context in the list of GitLab projects of section 2 - Figure 6). The Context manager module handles context-related monitoring and reasoning for the user. It provides a framework for implementing context types and context detection with core implementations. The context manager can access sensors with the help of a separate Sensor Manager as well as other context sources. The Context Management API provides methods to accessing and managing context information.

- Trust Manager (the respective repository is TrustManager in the list of GitLab projects of section 2 - Figure 6). This manager is responsible for assessing the trust in the relationship between two users. It takes advantage of scores provided by the Neuro Behavioural Classifier (see below). The implementation will be released in M22 (deliverable D4.5). For this reason, a repository for the Trust Manager has been already created but the implementation is not included in the beta.

- Personal Data Storage Manager (the respective repository is Storage in the list of GitLab project of section 2 - Figure 6). Personal Data Storage is used to provide certain unified file

storage abstraction for HELIOS applications without binding these APIs to specific storage location.

HELIOS extension modules

- Neuro-behavioural classifier module (the respective repository is NeuroBehaviouralClasifier in the list of GitLab project of section 2 - Figure 6). This module analyses the material exchanged between users to assess an emotion value. This value is used at the trust computation.

- Graph Mining Module (the respective repository is SocialGraphMining in the list of GitLab project of section 2 - Figure 6). This module aims to provide dynamic machine learning capabilities to HELIOS users to facilitate recommendation tasks.

- Media Streaming Module. The respective repository is MediaStreaming in the list of GitLab project of section 2. This module includes media transmission and playback capabilities. To improve the modular design, we have extracted the four features of this module, creating the respective four repositories (see figure 6): MediaStreaming-liveVideoStreaming, MediaStreaming-videoPlayer. MediaStreaming-fileTransfer and MediaStreaming-videocall. These four repositories are compiled as AAR libraries and MediaStreaming is a HApp that integrates them. Moreover, some media functionalities require the personal storage, as described in the DoA. The respective software uses the repository named PersonalStorageElements in the figure 6. The Personal Storage Elements are given as a set of Docker containers. For the beta, a docker-compose file is provided (which contains all the information to automatically create the containers and their volumes) along with the necessary documentation for their deployment.

- Rewarding Module (the respective repository is RewardingModel in the list of GitLab project of section 2 - Figure 6). This module manages the generation and distribution of tokens to create the HELIOS rewarding module. At the moment of the beta release, the rewarding module runs on an external machine. In the future, a stub will work as a gateway in the mobile phone to enable the connection with the external machine, preserving the same interoperability approach that any HELIOS component. Considering the implementation calendar in WP4, this module is not included in the beta release.

- Content Aware Profiling (the respective repository is ContentAwareProfiling in the list of GitLab project of section 2 - Figure 6). The repository contains a draft implementation, which is not yet ready to run on phones. The code is included in this deliverable, but it is not a part of the set of functionalities ready to deploy (in a consistent manner with the HELIOS work plan).

- Group Communications (the respective repository is GroupCommunications in the list of GitLab project of section 2 - Figure 6). The repository contains a draft implementation. The Group Communication module offers a set of functionalities for handling one-on-one and group communications over the p2p HELIOS network. The draft code implementation is included in this deliverable, although it is not a part of the set of functionalities of the beta (this component is included in a future deliverable, according to the work plan)

Beyond these architectural blocks, key elements for the integration of the beta release have been the apps, which are located at the top level of the architecture. The beta release integration has involved several apps:

- TestClient (which has a specific GitLab repository). This application includes a graphical interface and involves the use of the APIs provided by the core managers, such as Profiling, Security, Context, Contextual Ego Network and Messaging. TestClient has been a key piece to validate the beta approach, i.e., a HApp (or APK) that imports AAR libraries (HELIOS components) from the Nexus repository.

- MediaStreaming. As explained previously, it is a HELIOS app that integrates the four media functionalities developed in the project. It utilises the same mechanism described for TestClient: an APK that integrated the four features, compiled as AAR libraries and available via Nexus. A screenshot of the main activity (enabling the access to the four features is shown in Figure 8)

- Reporter App. This is a HApp linked to the Citizen Journalism use case (Journalistic App Interface in the architecture). This app also counts on a GitLab repository, named ReporterApp in the figure 6. This HELIOS app has been used to test an inter-modules communication approach based on Android intents. These tests have involved Reported App and the MediaPlayer included in MediaStreaming. This deliverable includes the Reporter App source code at the current state, although this development is due for future deliverables.

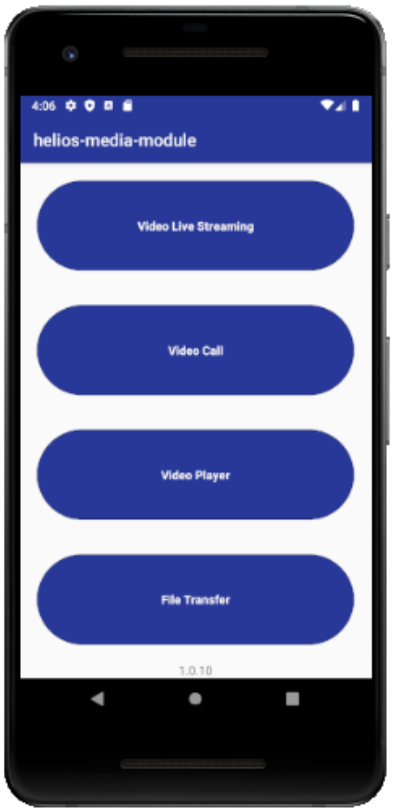

Figure 8. Main activity of the MediaStreaming app

Beyond these applications, during the preparation of the beta release, HELIOS partners are working on the integration of the Video Call (media functionality included in MediaStreamig) on HeliosTALK, which is a HApp developed in WP5 for use case A and group communication services. Except from Video Call core module, HeliosTALK also integrates, the Profile core module, the Contextual Ego Network Management core module, the Context Manager core module, and the Social Graph Mining extension module.

Considering that the beta release is public and the set of HELIOS components will be available for external developers, a key part of the integration strategy has been the creation of quality documentation by partners, to make easier the use of the respective components. Documentation consists of:

- Detailed API description, using the Javadoc format. This information is included and compressed in the tree of each repository

- A description with all the relevant information in the respective Readme.md of the GitLab repository.

4 Conclusions and next steps

According to the HELIOS workplan, this beta release is delivered in M18. This release integrates new functionalities beyond the alpha release, delivered in M12, and includes new HELIOS components.

The integration focus has also changed. Whereas the alpha release used TestClient as a key HApp to put together the functionalities of the implemented HELIOS components, the beta release is offering a set of HELIOS components, ready to be imported in HApps. In this sense, applications like TestClient and HeliosTALK are now HApps examples, that show the potential of the HELIOS modular approach to integrate features, from the developer point of view. The result of the compilation process for a HApp is an APK, including all the required modules inside. Deliverable D3.2 describes other architectural options, enabling a modular design also for the execution on the mobile phone. Future releases will consider this approach.

As described in this report, two new key integration tools have been added to HELIOS: Jenkins and Nexus. These tools have enables automation in the project, as planned in D6.2. Moreover, the consortium has continued the use of the previously deployed tools, like GitLab and Slack. The admin of the GitLab repositories has been a key part of the integration effort to prepare the beta release.

The beta release is public, according to the DoA. This deliverable is also public. The release of the set of HELIOS components is made by WP8, as part of the HELIOS milestone MS7 (Project Exploitation kit, initial version)

This beta release must be interpreted as a snapshot of the HELIOS components at the moment of submitting the deliverable. WP6 will resume the continuous integration / continuous delivery (CI/CD) HELIOS strategy, aiming at the Release Candidate (due date: M28), which will include consolidated versions of all the HELIOS components.

5 References

[1] HELIOS deliverable D6.2 "Alpha release" (December 2019)

[2] HELIOS deliverable D6.1 "Integration strategy and planning" (June 2019)

[3] HELIOS deliverable D3.1 " Draft system architecture and basic API specification" (June 2019)