Developing a Tele-Visit system in ACTIVAGE project

Andrea Carboni

Signal and Images Laboratory (ISTI-CNR)

andrea.carboni@isti.cnr.it

24/01/2019

Premise

The work described in this technical note is part of the technological development activities, for the year 2018, related to the H2020 project ACTIVAGE, a European Multi Centric Large Scale Pilot on Smart Living Environments.

1 Introduction

The CNR-SiLAB [1] Laboratory participation in the ACTIVAGE project involved two main fronts: the study of the possible interoperability between services belonging to different project instances (each participating country has its own project instance and its services) and the development of a system for remote audio/video communication, called "Tele-Visit". In this note, this last task will be analyzed in detail.

The purpose of the "Tele-Visit" system is to reduce the distance between healthcare workers and patients, with particular reference to cases in which the latter have motor problems or live in rural areas, offering them the possibility to be visited remotely by a specialist.

This note starts with a brief description of the ACTIVAGE project and of the Italian project instance. In the central part the various steps that led to the development of the final prototype of the “Tele-Visit” software are analyzed in detail, focusing on network issues encountered during development. Finally, the conclusions describe some security task to be implemented over the next few months.

2 The ACTIVAGE project

ACTIVAGE [2] is a European Multi Centric Large Scale Pilot on Smart Living Environments. The main objective is to build the first European Internet of Things (IoT) ecosystem across 9 Deployment Sites (DS) in seven European countries, reusing and scaling up underlying open and proprietary IoT platforms, technologies and standards, and integrating new interfaces needed to provide interoperability across these heterogeneous platforms, that will enable the deployment and operation at large scale of Active & Healthy Ageing (AHA) IoT based solutions and services, supporting and extending the independent living of older adults in their living environments, and responding to real needs of caregivers, service providers and public authorities. The project will deliver the ACTIVAGE IoT Ecosystem Suite (AIOTES), a set of Techniques, Tools and Methodologies for interoperability at different layers between heterogeneous

IoT Platforms and an Open Framework for providing Semantic Interoperability of IoT Platforms for AHA, addressing trustworthiness, privacy, data protection and security.

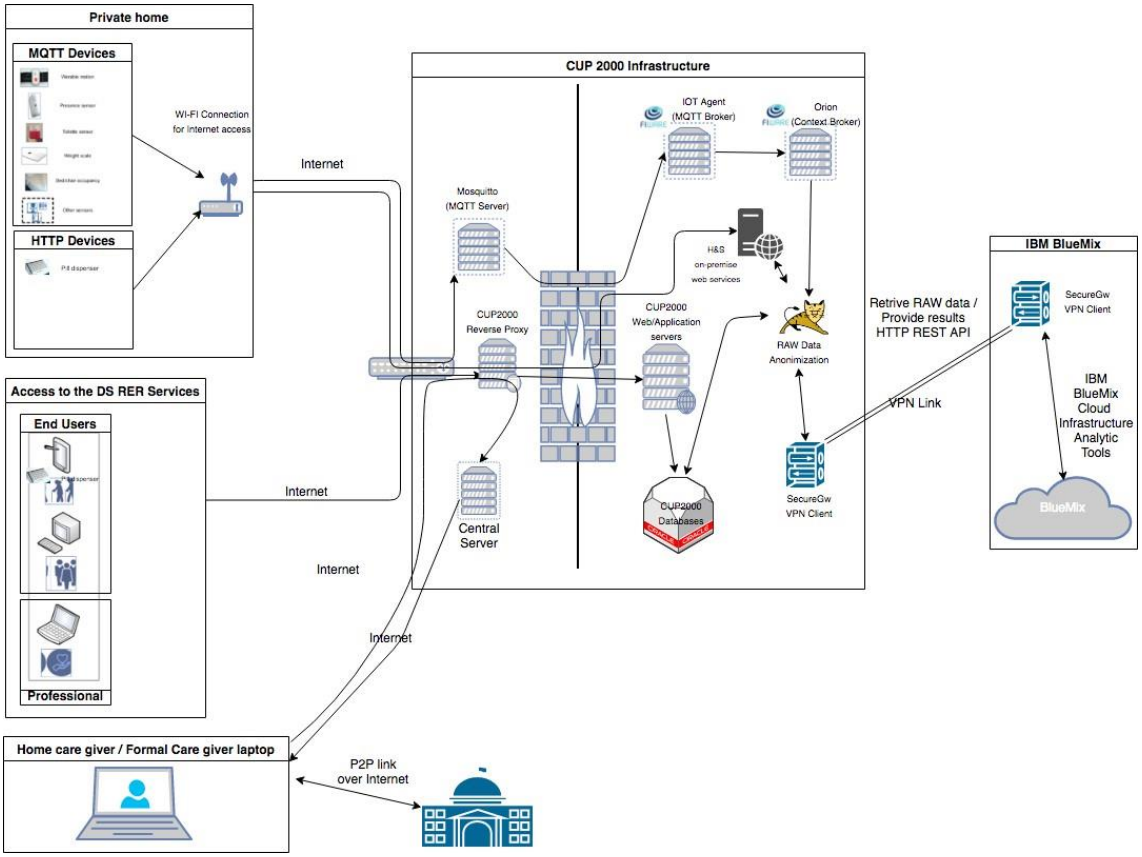

Figure 1: The Italian DS structure

IoT-enabled Active & Healthy Ageing solutions will be deployed on top of the AIOTES in every DS, enhancing and scaling up existing services, for the promotion of independent living, the mitigation of frailty, and preservation of quality of life and autonomy. ACTIVAGE will assess the socio-economic impact, the benefits of IoT-based smart living environments in the quality of life and autonomy, and in the sustainability of the health and social care systems, demonstrating the seamless capacity of integration and interoperability of the IoT ecosystem. ACTIVAGE will validate new business, financial and organizational models for care delivery, ensuring the sustainability after the project end and disseminating these results to a worldwide audience. The consortium comprises industries, research centres, SMEs, service providers, public authorities encompassing the whole value chain in every Deployment Site.

2.1 The Italian deployment site

The goal of the Italian DS (Fig. 1) is to provide IoT solutions and services to improve quality of life of older people in prevention of their physical, functional and cognitive decline, keeping them out of hospital, with focus in cardio-vascular disease patients. The pilot is aligned with a new organization model who is taking place in many regions in Italy: this new organization is called “Home of Health” and the new model is focused on the “case management” to provide high level services to citizens, increasing prevention activities. The Province of Parma was selected as the most promising for starting the deployment of ACTIVAGE IoT solutions. The Local Health Authority (LHA) of Parma has already established the territorial “home of health” as provided in the Emilia Romagna regional resolution since 2011. Demonstrating the evidence-based values of ACTIVAGE will allow to scale up firstly to the other 7 LHAs of the Region that will trust the provision of ACTIVAGE services potentially to about 1 million older persons who are over 65 and their relatives who will demand AHA services to support their independence and better quality of life. The expected reduction of costs produced by the pilot and the alignment with the new organization model, will contribute to the sustainability of the solution after the project’s end. CUP 2000 is the in-house providing company of Emilia Romagna region for ICT services so the replicability and the scalability of the solution in other areas will be managed by this partner. Social cooperatives, like AURORA, which provide services for adult older people through agreements with municipalities and Local Health Authorities will benefit in terms of additional market capabilities and improved revenue opportunities. CNR involvement in the Italian DS concerns data analysis activities, the study of interoperability problems among different DS instances and the development of the “Tele-Visit” software described in this note.

3 Tele-Visit system

The heart of the system is a client-server application for Windows operating systems, where the personal computers involved, respectively in use to patients and health professionals, are able to exchange audio/video data through a single server that acts as a bridge for communication. The choice to develop proprietary software was necessary to ensure the best level of security for data transfer, but also to avoid possible network congestion, which is why most of the existing commercial remote video-calling software cannot be used at all inside internal networks of most research centers, public administrations, hospitals, etc.

In the following two paragraphs, the starting and final status of the developed prototype are briefly shown, while the various intermediate development steps are discussed in the next chapter.

3.1 First step

The first software prototype of the "Tele-Visit" system was developed in 2016 in the context of the SiDoREMI [3] project, part of a home exer-games system for children affected by Autism Spectrum Disorder (ASD) [4], to be used in collaboration with family members. The basic idea was to be able to connect a family with health professionals directly from the software that ran the exercises, without having to forcefully switch to third-party applications. In this first version (Fig. 2) the IP address of the computer used by the healthcare professional was public and data for both audio and video

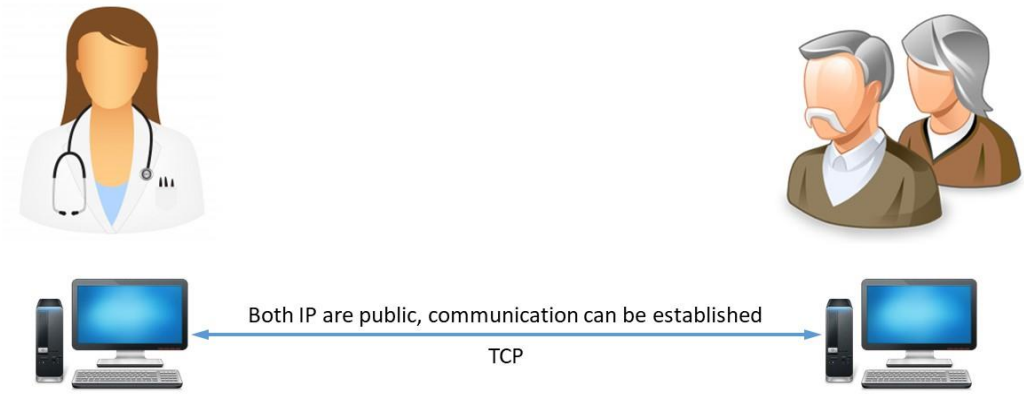

Figure 2: Tele-Visit system first prototype, connection example

communications traveled over TCP channels. The scenario thus saw the specialist's computer operating as a server waiting for TCP connections requests from patient-side services on two specific *IP/port* pairs (one carrying audio data and the other video data).

The main issues of this approach lay in the need to have at least one static IP address between the two endpoints and in a video streaming performance limited by the use of the TCP protocol, not an ideal option for real-time applications.

3.2 Final prototype

Through a series of steps that we will see in the following chapter, the final prototype of the system has evolved from point-to-point communication to centralized communication via public server.

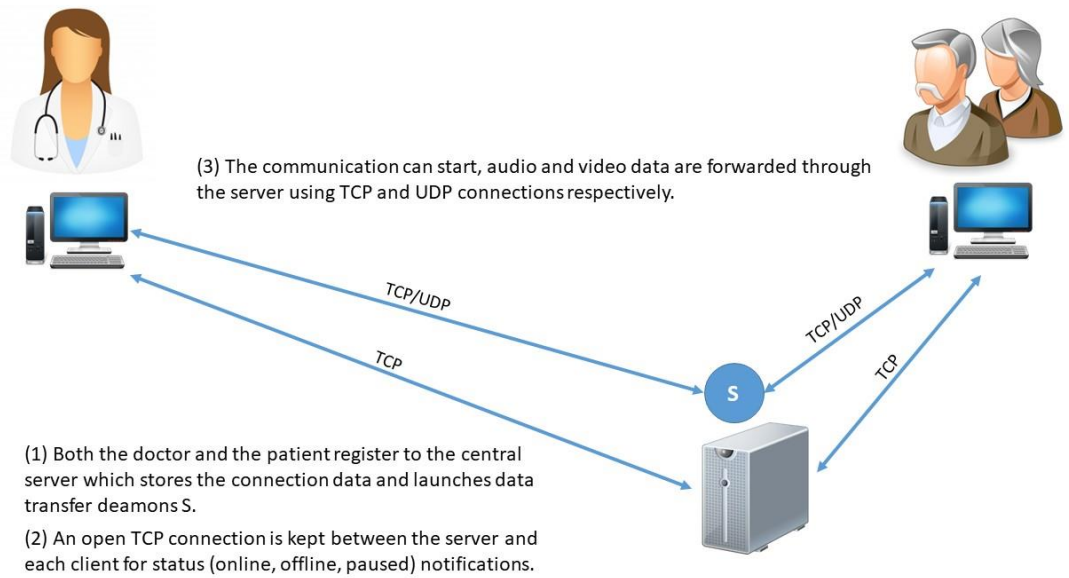

Figure 3: Tele-Visit system final prototype, connection example

The service daemons running on the server manages three main functions:

- Register public and private *IP/port* duplets of computers who want to take part in the communication
- Manage status changes (online/offline/active/inactive/paused)
- Launch and manage daemons **S** (Fig. 3), responsible for routing all audio/video data traffic

All audio data, service registration messages and status changes are transmitted via TCP protocol, while the transmission of video frames relies on UDP protocol, more suitable for real-time applications.

In order to optimize performance and reduce the possibility of packet loss, a parameterizable buffering algorithm has been implemented on both sides of the communication together with the possibility to set a packet size tailored to the specifics of the underlying network, through a custom algorithm responsible for the fragmentation and subsequent reconstruction of the video frames.

4 Development

The changes that led to the final prototype of the software mainly concern the part related to video streaming. The main steps are summarized in the following points:

- Introduction of a public central server service, responsible for endpoints registration, status handling and traffic routing
- Experiments with UDP Hole Punching techniques for point to point transmission
- Implementation phase for the correct routing of audio and video traffic between the two endpoints and management of the services state in real time
- Studies on the correct sizing of packages, use of buffering techniques and packet segmentation
- Communications security study

In the following paragraphs will be highlighted the most important aspects that led to the creation of the definitive prototype of the system, also giving a brief description of how a commercially established software such as Skype works.

4.1 Hole punching techniques

UDP Hole Punching refers to a technique that allows the establishment of bidirectional UDP connections between computers belonging to private networks connected via the Internet using Network Address Translation (NAT). Each host behind a NAT device contacts a third known server, accessible through a public address, then, once the NAT device has established a UDP session with that host, it can switch to direct communication, supposing that the NAT device will retain the information status, despite the fact that the packets come from different hosts.

With Symmetric NAT, since the IP address of the known server is different from that of the endpoint, the NAT mapping that the known server sees is different from the mapping that the endpoint would have used to send packets through the client. For this reason, UDP Hole Punching doesn’t work with Symmetric NAT (or bi-directional NAT), which, as in our case study, tend to be used within large corporate networks.

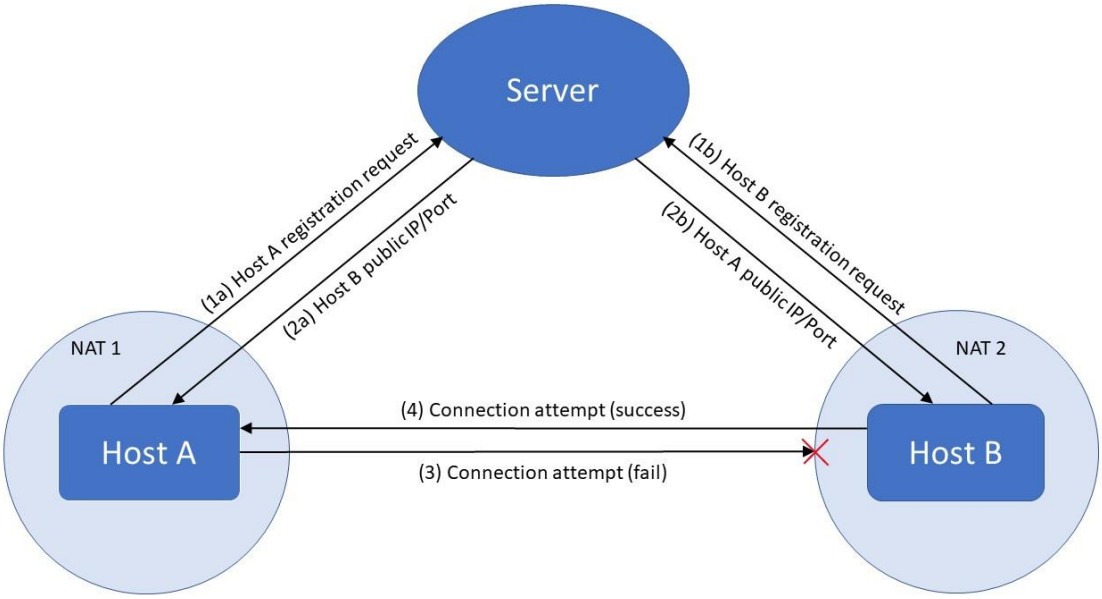

Figure 4: UDP Hole Punching

Taking a more detailed analysis (Fig. 4), we consider the situation in which both hosts start sending packets to each other, making several attempts. On a "Restricted Cone NAT", the first packet coming from the other hosts will be blocked. After the NAT device has stored a record resulting in the sending of a packet destined for the other machines, it will allow every packet coming from these IP addresses and with the same port number to pass through. The technique is widely used in P2P software and VoIP telephony, this is one of the methods used by Skype to bypass firewalls and NAT devices and it can also be used to establish VPN connections. The same technique is sometimes used with TCP connections, albeit with much less success.

4.1.1 Algorithm

Let **A** and **B** be the two hosts, each in their own private network; **Nat1** and **Nat2** are the two NAT devices; **S** is a server with a public and reachable IP address.

1. **A** and **B** begin a UDP conversation with server **S**; **Nat1** and **Nat2** devices create UDP translations and assign temporary port numbers for external communication. **S** records the respective global *IP/port* pairs for each host.
2. **S** sends to **A** the *IP/port* pair relative to host B and does the same with **B** and **A** data.

3. **A** tries to contact **B** using the data received from **S**, the connection attempt fails but, in doing so, it creates a hole in the firewall of **A** to allow the reception of packets from **B**. Same thing happens at the first attempt to connect **B** to **A**.
4. Once the hole in the firewall has been created, **B** can successfully send messages to **A** and vice versa.

4.2 Public server introduction

The first requirement that emerged once the development of the new prototype was started, was dictated by the fact that in this new scenario the personal computers involved in communication could be indifferently on public or private networks. The first step was therefore to introduce a server (Fig. 5) with a known and accessible address to be used to facilitate communication. This server has undergone a natural evolution in the course of development and, from a simple database of hosts connected and interested in communication, has become the real heart of the system. The three main steps of its evolution are described below.

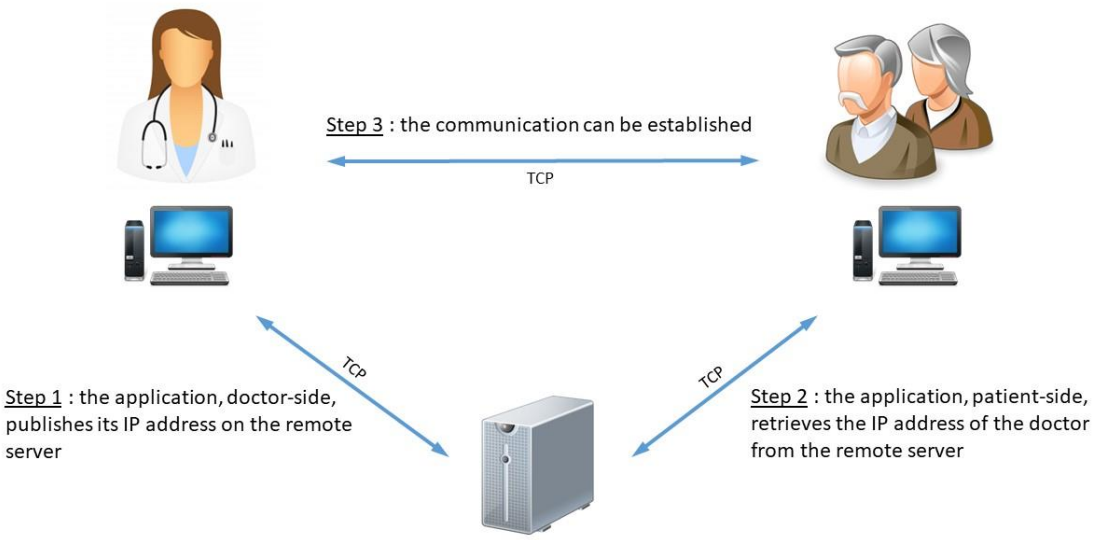

Figure 5: Server introduction

4.2.1 Host database & UDP Hole Punching

The first step in the server introduction was to set up a service able to keep track of all the hosts interested in communicating. All data transfers between this service and the

Tele-Visit application running on endpoints were made through TCP channels and consisted only in the exchange of the *IP/port* pairs. The server also took into account the logout of the users once the application was closed on the client side, becoming no longer available for the communication. In this phase the UDP hole punching techniques described in the previous paragraph were tested, changing various network conditions, proving to work exactly as expected. However, in our use case, it was not possible to rely completely in hole punching techniques due to an incompatible NAT configuration, as some of the users were under the national health service network. Finally, it was decided to use the server as a bridge to all the audio and video data traffic.

4.2.2 The server as audio and video data bridge

In this phase, the server has undergone the most profound transformations since in addition to the services related to the simple registration of connected hosts, the daemons responsible for the correct routing of all traffic from one host to another have been implemented.

The main phases of development are as follows:

- Registration of all the hosts interested in the communication
- Setup all services for the correct data traffic routing among endpoints
- Creation of a simple buffering algorithm for the management and correct routing of incoming and outgoing messages

4.2.3 Status notifications handling

The five possible states of a client are: online, offline, active, inactive and paused. Since all traffic passes through the server, it has to manage the status of the connected clients and communicate to all of them the status changes to be made. For example, supposing that during communication a network problem causes the interruption of communication with **Client1**, the server must manage the correct closing of communication channels to **Client1** and inform **Client2** to change status from active to inactive, thus interrupting the acquisition of audio/video from microphone/webcam and all the network traffic to the server, alerting the user with appropriate messages.

4.3 **How Skype work**

Skype uses the UDP hole punching technique to allow communication between users who are behind NAT. However, Skype does not use a separate server to act as a third party host. Rather it uses users' computers to act as a third party host. Any client, which has a publicly reachable IP, can become the third party host. Hence, this may increase the load on Skype's users as they are responsible for initiating the connection between the users who are behind NAT. For this reason, use of Skype is usually forbidden inside internal

networks of most research centers, public administrations, hospitals, etc. Sometimes UDP hole punching may not be possible due to various reasons like NAT port randomization. In the cases where UDP hole punching is not possible, the third party host (i.e., a Skype user's system having a globally reachable IP address) is used to relay the whole communication between the users who are behind NAT.

4.4 UDP video transmission optimization

As already seen above, in the Tele-Visit software implementation, the channels for the transmission of audio and video data are based respectively on the TCP and UDP transport protocols of the second level of the TPC/IP stack. Since UDP, unlike TCP, does not guarantee the actual arrival of data segments and their correct ordering, it is important to establish correctly the size of each UDP datagram sent over the internet in order to reduce packet loss. Both TCP and UDP rely on the underlying IP protocol (network layer), where the maximum size of a packet is 65535 bytes, therefore, focusing on UDP, considering the IP packet header size of 20 bytes and the UDP header size of 8 bytes, the theoretical maximum size of the data field would be 65507 bytes. It is also necessary to consider the MTU (maximum transmission unit), that is defined as the maximum size in bytes that a data packet sent through a communication protocol can have. Since [5] the minimum MTU supported by the IP protocol is 576 bytes, the maximum data field size in a UDP packet that aims to minimize the possibility of going lost, considering the already mentioned 28 bytes of header, is set at 548 bytes.

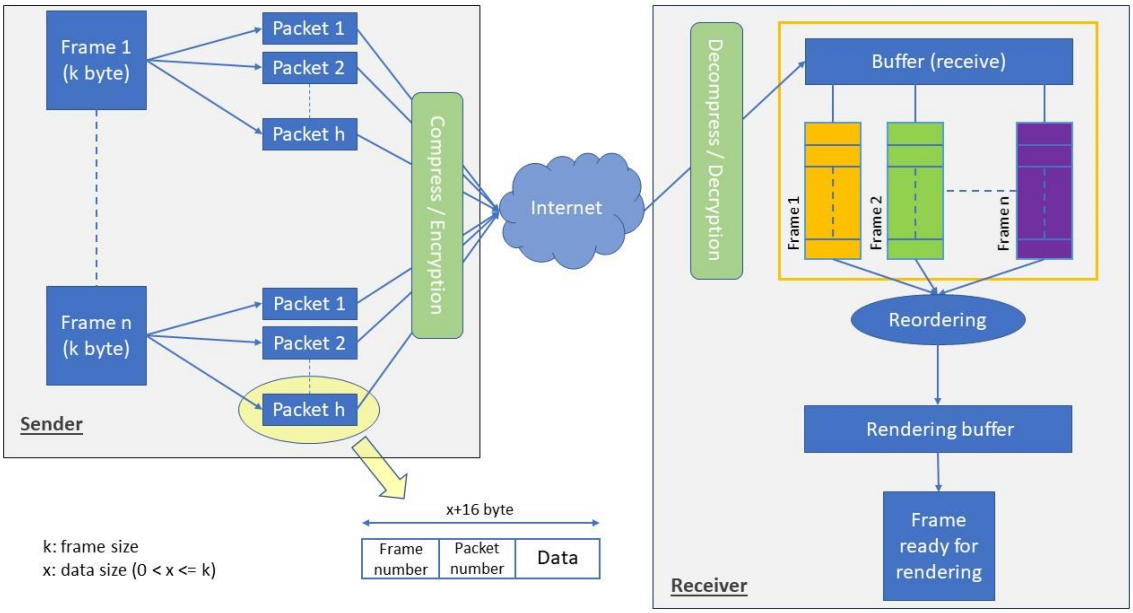

Figure 6: Frame fragmentation and reconstruction algorithm

This number represents a minimum estimate and is therefore susceptible to variations as each network segment may have different MTUs, but based on the tests performed it has proved to be reliable.

Returning to the software object of this report, it was therefore necessary to implement an algorithm (Fig. 6) capable of fragmenting each video frame into *n* packages, with a maximum size of 548 bytes, and subsequently, once all packets of a frame are received by the recipient host, proceed with the recomposition of the image.

The main problem, since it is UDP, in addition to the possibility of loss of packets (that being video streaming in real time are simply ignored) is not guaranteed the order of arrival, so it was necessary to change the UDP data field introducing the two fields Frame Number (order number of the video frame) and Packet Number (order number of the data packet relative to the current frame) of size 8 bytes each. In doing so, the maximum size of the data part drops to 532 bytes.

Once dimensioned correctly, the algorithm operation is very simple: each frame is broken up into *n* sub-packages that are sent by the Sender on the network. On the receiver side, each received packet is analyzed and, depending on the Frame number and Packet number values, its data field is inserted into a list of byte arrays in the correct position. Once all the packages relating to an image are received, it is rebuilt and then passed to the rendering algorithm that will take care of the on-screen display.

5 Conclusions

The software described here represents a working and approved prototype for experimentation on some subjects identified within the ACTIVAGE project. The next steps will focus on issues related to the security of transmissions, with the aim of ensuring a level of service that is appropriate to different network conditions. Currently, a public-key style TLS algorithm is being implemented for the security of UDP communications where the key exchange phase takes place via the SSL/TLS layer on the TCP channel.

Bibliography

[1] http://si.isti.cnr.it/
[2] http://www.activageproject.eu/
[3] http://sidoremi.isti.cnr.it/
[4] Magrini M., Salvetti O., Carboni A., Curzio O. (2016) An Interactive Multimedia System for Treating Autism Spectrum Disorder. In: Hua G., Jégou H. (eds) Computer Vision – ECCV 2016 Workshops. ECCV 2016. Lecture Notes in Computer Science, vol 9914. Springer, Cham
[5] Mogul, Jeffrey C., and Steven E. Deering. Path MTU discovery. No. RFC 1191. 1990.