3. GSN\_SIM: a HLA simulator

***3.1. Introduction***

This study presents, in an expanded and updated form, the scientific results published in 2017 on the occasion of the scientific conference “Scientific Research and Education in the Air Force” [1]. The GSN\_SIM application was developed for the training and evaluation of military operational personnel at the tactical level in the pre-deployment stage, both for improving skills in operating the fire control system and for improving the decision-making process.

The need for the scientific approach derives from the diversity of requirements: ensuring interoperability between different force structures in a national or allied context; simulators are essential in competency-based training [2]; an educational process adapted to the operational requirements and an efficient training are essential for the accomplishment of the mission.

The planning of the training activity for development/ maintaining operational capacity of the military organizational microstructures of tactical level (e.g. Firing Battery) must take into account: the essential requirements of the mission, the training priorities, the available resources. The list of essential mission requirements is the specific tool that interconnects the requirements, conditions, and performance standards.

The need for effective training, both individually and as a team, is essential for fulfilling the entrusted mission. Simulation-based training has the advantage of evaluating possible scenarios safely and makes it possible to distribute the simulated scenarios to several users/ operators who are put in the situation to make decisions individually.

The concept established for the academic environment is *military simulation-based training*, but it can be easily adapted for the military operational process [2]. In the educational process, gamming has the role of developing and maintaining action and decision-making skills specific to the use of the weapon system in combat [3,4]. In the military decision-making process, war gamming is used for the courses of action analysis, allowing the dynamic visualization of the actions in the combat space.

In the specialized literature [5,6] the simulations are divided as follows:

**1. Live simulation** – real people operate real systems under real conditions. An example would be military exercises with troops in the field: personnel, equipment, logistical support and real medical insurance, real environmental conditions, and subsistence.

**2. Virtual simulation** – real people operate virtual systems under virtual conditions, but virtual conditions are programmed (created) by real people. An example is the training of pilots on simulation systems, in weather and visibility conditions artificially created by instructors (real people); the GSN\_SIM simulator is also one of those serious games that fall within the specifications of a *virtual simulation*.

**3. Constructive simulation** – virtual people operate virtual systems under virtual conditions, but the scenario is built by real people and the behavior of virtual people is set by real people. JCATS is the tool used for training commands, staff, checking and testing standard operating procedures, the analysis of various courses of action etc.

**4. Distributed simulation** – synthetic environment designed to ensure real-time data exchange between different stand-alone simulation applications [7]. In the virtual battlefield a whole “palette” of simulation systems is used. They are interconnected and standardized on a common platform to speak a “common language”.

The best example is the connection to JCAS, through standard simulation systems, of all components/ parts (people, aviation equipment, artillery, tanks, tracks, resources etc.) introduced in the simulation.

The LVC simulation architecture captures the interaction between live assets, virtual assets, and constructive models [8] to facilitate the action and decision-making training necessary to accomplish any type of mission.

For training, both staff and operators should connect constructive, virtual, and real simulators in one physical model – on a multilevel simulation. Now the Romanian army has some individual simulators (aviation, naval, land, air defense), but that model works independently and trains the operator individually, in individual scenarios.

Multilevel simulation is a federation model, a connection between constructive, virtual, and real simulations. That connection is done with High Level Architecture (HLA). HLA is a standard essential part of a multilevel simulation and is the key for models like that. Technically it is a procedure, a common language between different types of simulations [9, 10]. The purpose of the multilevel simulator is to give a feedback to C2 staff representing the results of plans and planning, generating reports, and training the staff in making decisions in the military process. The staff has the opportunity to restart one or more procedures if something is wrong with the execution of the plan.

Another essential part of training is represented by software and models designated to train real people, not only theoretically but also in practice and epistemic learning too. That kind of model is called serious games and is the future key for learning and training skills [11].

The analysis of the market of simulations and serious games for decision makers leads to the appreciation that there are two important trends:

1. focusing on the objective of rendering the simulated reality as accurately as possible.

2. the use of simulations using game theory or other mathematical tools to evaluate scenarios and "disguise" them under a graphical interface.

If in the first case, the graphics takes up a large part of the game's time and budget and the content to be transmitted being extremely simple, in the second situation, the scenario is more complex, but the graphics are often boring, and the interface of the game is complicated.

The purpose of the simulation is to acquire and develop operating/ decision skills with direct effects on the rapid improvement of performance, in the context in which there is a great mobility of staff, and the time required to achieve the cohesion of combat teams is very compressed. In this sense, the GSN\_SIM application has the ability to accurately reproduce the essential characteristics of the workstation from the GunStar\*Night fire control system and to provide them to the decision-maker.

***3.2. The Design of HLA***

The High-Level Architecture (HLA) is a software architecture designed to create computational models or simulations from component models. The HLA was adopted by the US Department of Defense (DoD) to be used in specific modeling and simulation activities.

HLA is implemented in many civilian applications. The name was initially adopted by the DoD, in the context of a variety of military simulation programs. HLA was considered a "high-level" language model, simulation programs that generated information modeled over time [12].

HLA is a standard that allows all types of simulation systems to work together [13]. Simulation systems must work together so that they can achieve a global goal through the exchange of services. According to the literature [14], there are five important concepts regarding HLA:

**1. Runtime** **Infrastructure** (RTI) is a piece of software that provides HLA services. The main service is to send and receive data right to the receiver. RTI is also a component that provides the infrastructure (such as an automatic bandwidth) through which federals, in a federation, can exchange data.

**2. The Federate** is the basic unit of the composition of an HLA-based simulation. A Federate is a system that interconnects a simulator to the RTI. Each federation can model any number of objects in a simulation, can encapsulate an application that participates in an HLA-based simulation; for example, an aircraft simulator can be connected with an air defense simulator and a radar simulator and all can be connected with a constructive simulator.

3. A federated HLA is described by a **Simulation Object Model** (SOM). SOM is an XML-based file that describes the interface of a federation in terms of the data it generates as outputs and the data it requests as inputs.

4. The HLA **Federation** is a set of federations that are used to build a simulation and are all federated together with the RTI. They are described by a **Federation Object Model** (FOM). This is the group of systems that interoperates and describes how federations in a federation are connected to each other, contains a description of the data exchange in the federation, and this can be seen as the language of the federation. The actual operation of an HLA federation is called the HLA **Federation Execution**.

5. **The Federation Execution** is a session in which the federation runs. If you run the federation several times, you will get more executions of the federation, so you will get more sets of results, results that can be interpreted in a statistical form.

The HLA defines two mechanisms to allow federals to coordinate their activities: synchronization points allow sets of federations from the same federation to synchronize their activities; interactions allow federals to perform actions that have an effect on other federations in the same federation.

Like any kind of simulation system, federals are characterized by their interfaces (it is necessary to build a tool that represents the link between the tools - simulator and the main train target - the operator). The structure of these interfaces is in turn defined by the HLA standard.

The HLA standard assumes an input-output architecture in which federations enter (publish) the data they generate as outputs and retrieve (upload) the data they need as input; therefore the GUI (Graphic User Interface) is a "must". A federation can only subscribe to data that is published by other federates in the same federation.

The HLA standard accepts both primitive, simple types (integers, Boolean etc.) and more complex structure types. Some basic data types are defined by the standard, but users can define new user-specific data types.

The main advantage of an environment with a friendly interface, close to visual communication, is that it is easy to replace a simulation package with another simulation package or, moreover, it can be updated and replaced with a better one. If the new simulation package (with new or upgraded blocks) has the same interface as the old one, then changing the simulation package has no impact on the rest of the simulation.

GSN\_SIM has been designed and works as an HLA architecture, according to the "data flow": the main console (GSN SERVER) sends a packet to the secondary console (GSN NIGHT) and the packet represents "an input" for the second (for the main purpose to train). After all procedures and adjustments, GSN NIGHT sends the data to a "text file" and the data is "an output".

Runtime Infrastructure is a basic topology consisting of several simulations that have a single connection to a main highway (Figure 1).

JCATS

Runtime Infrastructure - RTI

FOM

Flight

Simulator

ATC

Simulator

Radar

Simulator

GSN SIM

Simulator

**Fig. 1** The HLA Architecture for Air Defense

Data resulting from compliance/ non-compliance are interpreted and saved in external files (this data represents ***federation’s output***).

***3.3. Specific technical requirements***

Oerlikon Contraves 35 mm air defense system is an automated system designated to protect different objectives (e.g. HQs, air bases). The Oerlikon Contraves system is a complex system, consisting of several automated, interconnected subsystems, which can work centrally (basic mode), but also decentralized.

If the radar (SHORAR-TCP), Gun\*Star Night and 35 mm Gun Air Defense System are engaged in a permanent position, according with the Drill Book, a complete reconnaissance needs to be made only once [15].

Usually the position of Search Radar, Fire Control Units (FCUs), Guns and Power Supply Units (PSUs) is fortified or protected in some way. Also, the ammunition dump, first aid post and quarters can be constructed and perhaps coordinated with the installation of the defended critical asset or vital point. The communication systems can be permanently installed. A typical critical asset of this type is an air-base or a land forces-base. In this case the defense system should be coordinated with the activities of the Air Force or Land Force. A lot of infrastructure can be used by air defense, too.

The Oerlikon Contraves 35 mm air defense system (Figure 2) is a complex one. It works in a centralized mode and is mainly composed of [16]:

1. SHORAR TCP - designated for search, detection, identification, and automatic target tracking. The SHORAR has the main role of managing the air space security and optimizing the air targets data to the fire control system Gun\*Star Night.

2. Gun\*Star Night is an automated subsystem designated to control two cannons. It is based on electro-optical tracking system and equipped with a computer to calculate ballistic trajectory and the timing of the fire release. Fire control system Gun\*Star Night (FCU-GSN) provides the following main operations:

a) receives data on air targets at SHORAR TCP or its digital optical sight (DOS);

b) evaluates the potential air attack in the area of responsibility (in decentralized mode);

c) searches, finds and identifies the target;

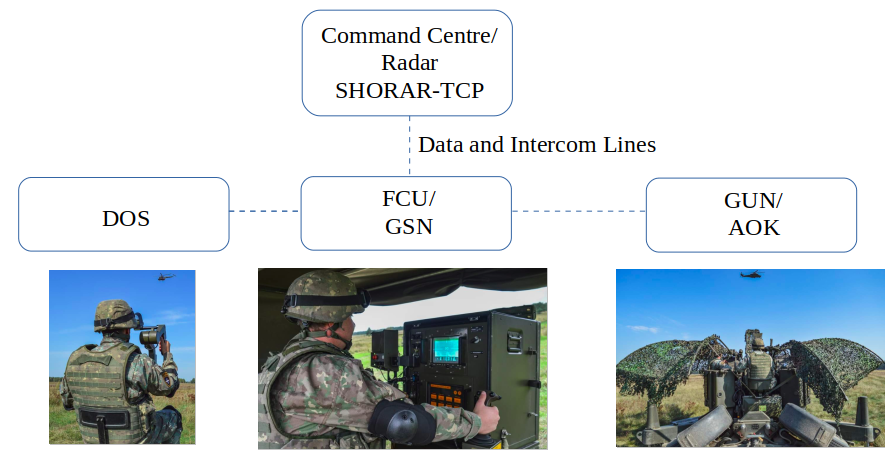
d) tracking three-dimensional target (finders using laser and electron-optical system);

e) calculates angles of sight for guns;

f) forwards angles in gun sights.

3. Digital Optical Sight (DOS) is operated by a single operator. It is designated to search air targets and send data to GSN. DOS is the main Intel sensor when GSN works in decentralized mode and receives data only from this subsystem.

4. Anti-aircraft automatic guns (AOKs) comprises two 35 mm caliber and are designated to fire directly against air targets or field targets, controlled by GSN or independent.



**Fig. 2** The system architecture

A usual scenario looks like this: SHORAR-TCP radar is searching the air space and detects a target at 28 km; it identifies the target as friend/ non-friend and sends data to the GSN which has the best position in AOR or is ready to combat.

Sending data to one or more GSN is made according to the best course of action and according to the value of the target. This is a procedure that is one of the most important part of the military decision-making process and is a quick and short one.

GSN is requiring the target and it follows it with infrared camera. The target’s data are continuously updated by computer and laser telemeter. When the target is in the fire area, AOK can open fire against it conducted by FCU.

The output laser device wavelength is 1.54 μm. In normal conditions of utilization, this laser device is considered eye safe as defined under class 3A in accordance with the EN 60825-1 classification. To prevent permanent injury to the human eye it is strongly recommended never to view at the laser directly or with optical instruments (lenses, binoculars).

The whole system (two sets cannons and fire control system) is powered by electricity, with voltages of 220V AC and 110V DC. To work independently of the national electricity network in any place and time conditions, voltages are provided by two sets, running on leaded petrol, with a consumption of 24 l\h, and a generating set of low power, functional unleaded petrol, with a consumption of 1.5 l\h. Thus, we can realize the costs involved in preparing a one-man operator, since the basic and advanced skills needed require at least three training phases, each phase taking place over a period of 2-3 weeks.

***3.4. GSN\_SIM: a HLA simulator***

GSN\_SIM is a simulation program for Gun\*Star Night fire control system destined for learning and involvement of the operators and staff, for training and improving their skills to shorten the preparation time and training costs.

The model is a multilevel simulator. The FCU simulator (for individual operator) and the console (for staff or instructor) are connected and are working together, sending, and receiving data and generating reports to staff about how the operator works.

Simulation software can be a way to train even when, for some reason, the technique is unable to be efficient. It also represents a necessary learning tool for the main menus and submenus and the presentation of the art verification algorithms. In current conditions, modern combat no longer leads through direct contact with the enemy, but it is based on information received from higher echelons or the discovery and tracking of its equipment.

In accordance with modern conflict laws, the Gun\*Star Night system uses information received from radar sensors regarding the aerial targets it displays in video format. Target data are displayed while the operator has selected the option of designating the target by radar and has been aligning with the radar.

If the operations have not been performed, no data on the flight parameters of the target will be displayed on the screen. To make it easier to understand the simulator's operations, all data can be entered by the commander/ instructor or operator depending on the scenario.

The success of the operation depends not only on the operator, but also on the commander/ instructor; good planning and good force management will lead to the success of the action. If the procedures are not followed or if the plans contain errors or omissions, then the mission may not be fulfilled, even if the operators have done everything according to the existing procedures. The failure of the mission can also be determined by the fact that the system takes into account all the details (location, landmarks, forbidden areas, shooting and safety sectors etc.).

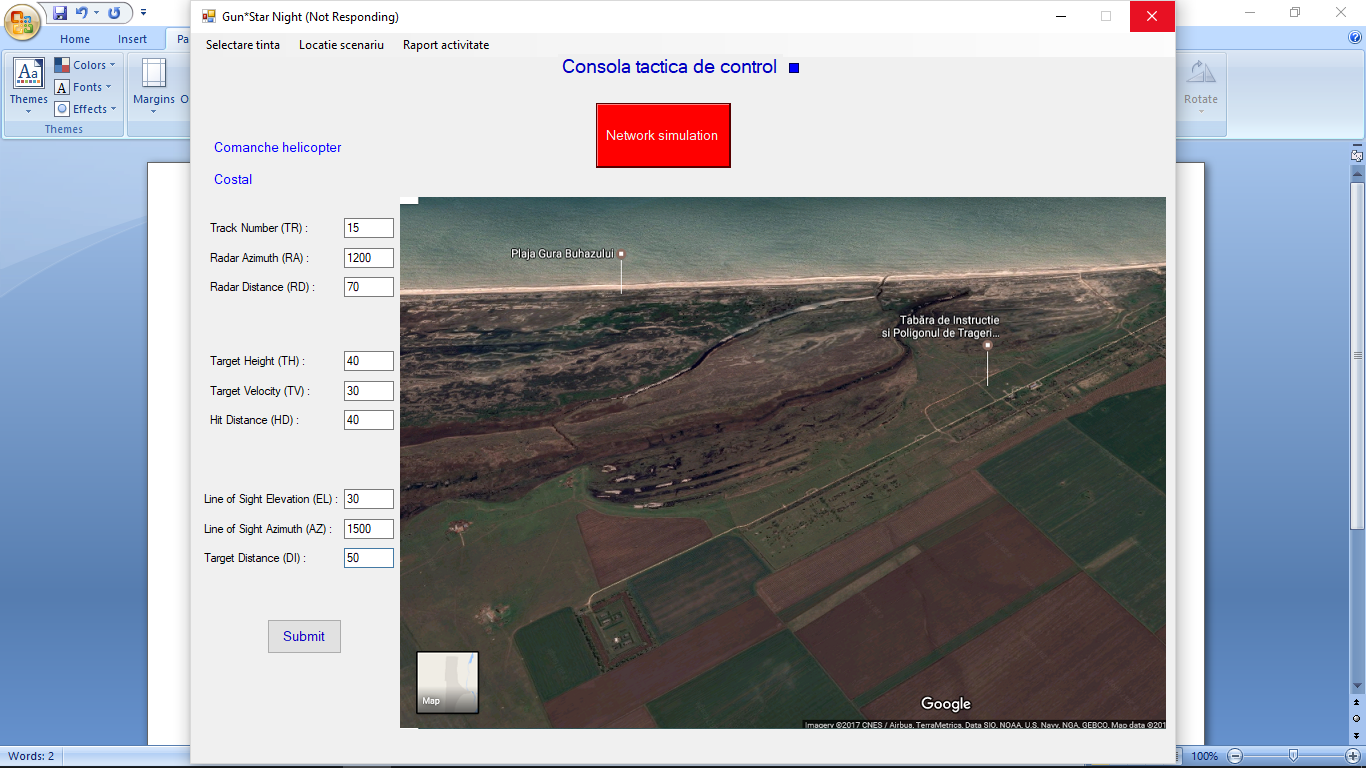
3.4.1. The tactical control console

The tactical control console represents an innovative approach resulting from the operational requirements at the level of each firing channel through which the battery commander is introduced in the command and control chain for the purpose of planning-training-evaluation (C2).

One of the tools he has at hand are work maps. They capture three locations in Romania, locations where training-evaluation activities can be carried out with anti-aircraft artillery system. All three locations are recognized, marked on military and civilian work maps and fully comply with the requirements for carrying out these types of activities in very good conditions.

The locations represent the Training Camp for missiles and anti-aircraft artillery from Capu Midia (coastal area), a training ground near the city of Brasov (mountainous area) and another range near the city of Craiova (plain area).

The locations where the trainings take place are selected from the work console of the subunit commander (GSN SERVER), who has the task of choosing the location (Figure 3). This option was introduced to give the subunit the possibility to train in any weather conditions (clear sky or fog) and customized to the areas of interest, so that the formation of skills is done in a virtual environment as close as possible to the real one.



**Fig. 3** Selecting location

From a technical point of view, the maps are loaded with the help of procedures whose source code for the *Costal* variant includes:

privatevoidtIPSACapuMidiaToolStripMenuItem\_Click(object sender, EventArgs e)

{

HartaLucru.BackgroundImage = Image.FromFile(@"D:\GSN SIM\Tools\Pictures\ADACapuMidia3DAligned.jpg");

GSN.LocatieScenariu = "CapuMidia";

label12.Text = "Costal";

}

private void tIPSACapuMidiaToolStripMenuItem\_Click(object sender, EventArgs e)

{

HartaLucru.BackgroundImage = Image.FromFile(@"D:\GSN SERVER\Tools\Pictures\CapuMidia.png");

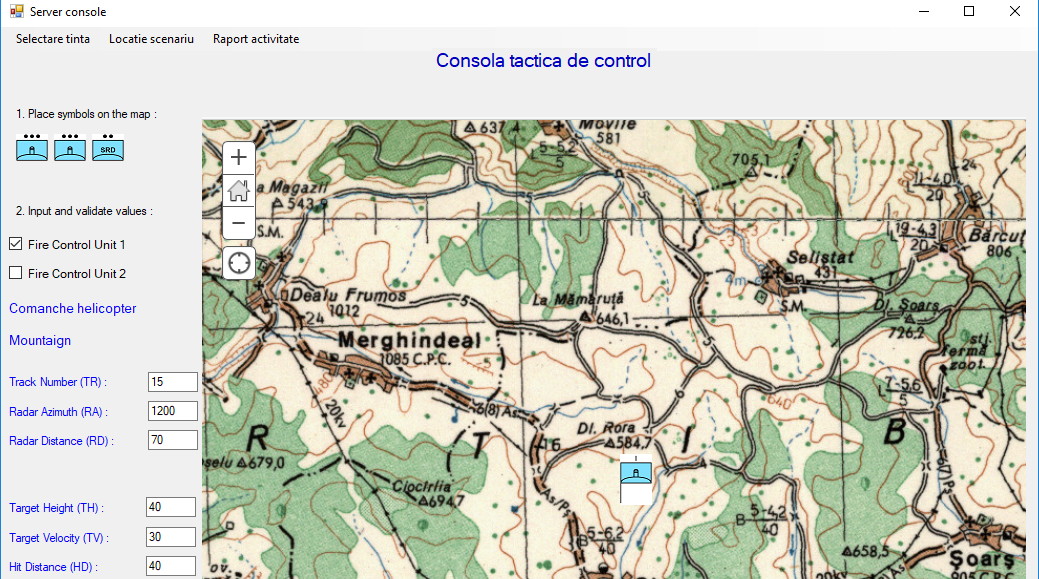
Location.Text = "Costal";

}

The maps themselves were uploaded from secure, unclassified sources in .jpg electronic format. The working format is close to that of military maps, so that the observance of operating procedures by operators is as close as possible to the real one.

The subunit commander has the possibility to train both fire control devices, in an integrated or independent way, in turn, with the aim of acquiring and training the skills regarding the preparation for combat of the system.

Training in an integrated or separate way is done by choosing the fire management system, no 1, no 2 or both from the ‘Input and validate values - Fire Control Unit’ submenu. Another element through which we introduced the subunit commander in the centralized training process is the location, on the tactical map, of the main elements of the anti-aircraft defense subunits (Figure 4).



**Fig. 4** Selecting the location, FCU and setting the target data

By placing, on the work maps, the elements of the subunit, the training, from a tactical point of view, of the subunit commanders is achieved. He has the freedom to place the elements of the subunit, respecting the technical and tactical norms and it is assumed that he knows both the tactical norms and the technical possibilities of the system.

private void FU1Symbol\_MouseDown(object sender, MouseEventArgs e)

{

moving = true;

startLocation = e.Location;}

private void FU2Symbol\_MouseDown(object sender, MouseEventArgs e)

{

moving = true;

startLocation = e.Location;}

private void BtSymbol\_MouseDown(object sender, MouseEventArgs e)

{

moving = true;

startLocation = e.Location;}

private void FU1Symbol\_MouseMove(object sender, MouseEventArgs e)

{

if (moving)

{

FU1Symbol.Left += e.Location.X - startLocation.X;

FU1Symbol.Top += e.Location.Y - startLocation.Y;}

}

private void FU2Symbol\_MouseMove(object sender, MouseEventArgs e)

{

if (moving)

{

FU2Symbol.Left += e.Location.X - startLocation.X;

FU2Symbol.Top += e.Location.Y - startLocation.Y;}

}

private void BtSymbol\_MouseMove(object sender, MouseEventArgs e)

{

if (moving)

{

BtSymbol.Left += e.Location.X - startLocation.X;

BtSymbol.Top += e.Location.Y - startLocation.Y;}

}

private void FU1Symbol\_MouseUp(object sender, MouseEventArgs e)

{

moving = false;}

private void FU2Symbol\_MouseUp(object sender, MouseEventArgs e)

{

moving = false;}

private void BtSymbol\_MouseUp(object sender, MouseEventArgs e)

{

moving = false;}

private void GrCcSymbol\_MouseDown(object sender, MouseEventArgs e)

{

moving = true;

startLocation = e.Location;}

private void GrCcSymbol\_MouseMove(object sender, MouseEventArgs e)

{

if (moving)

{

GrCcSymbol.Left += e.Location.X - startLocation.X;

GrCcSymbol.Top += e.Location.Y - startLocation.Y;}

}

private void GrCcSymbol\_MouseUp(object sender, MouseEventArgs e)

{

moving = false;}

}

In this way, the chances of fulfilling the mission are maximized. The skills of preparing the technique and conducting the fight with an air opponent are formed and perfected, errors or unfavorable courses of action can be identified and eliminated, the cohesion of the fighting team is developed.

3.4.2. Network data transmission-reception procedure

There are several processes running on each computer, and each process can have multiple open channels of communication. Therefore, there must be several active links (connections) on a computer at a time.

Communication is mediated by the operating systems on the computers running the two processes. Opening a connection, closing it, transmitting, or receiving data on a connection and configuring the parameters of a connection are done by the operating system at the request of the process. Process requests are made by calling system functions from the socket family.

In the case of the communication between the user process and the local operating system (via socket family calls), the local ends of open connections are called sockets and are identified by integers, unique in a process at any given time.

Each entity that communicates within the network is identified by a unique address. An address is actually associated with a socket. The address is formed according to the rules of the used network protocol.

The socket interface contains functions for communication according to both the connection model and the datagram model. The system functions provided allow communication to be established through different protocols (e.g. IPv4, IPv6, IPX), but have the same call syntax independent from the desired protocol.

According to the literature, the server process provides services obtained through the network; accepts (iteratively or concurrently) requests from a client process; performs a certain service and returns the result. The client process initializes communication with the server and requests a service and then waits for the server to respond.

The definition of the sockets for both consoles of GSN (client) was done as follows:

namespace GunStarNightServer

{

public partial class Client : Form

{

public byte[] data = new byte[1024];

public IPEndPoint ipep1 = new IPEndPoint(IPAddress.Parse("172.16.68.21"), 9050);//pt FCU1

public Socket server1 = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

public IPEndPoint ipep2 = new IPEndPoint(IPAddress.Parse("172.16.68.22"), 9050);//pt FCU2

public Socket server2 = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

The connection attempt, through the sockets, is made by dealing with the exceptions, on each client, as follows:

private void Target\_Click(object sender, EventArgs e)

{

if (FireUnit1Values.Checked)

{

try

{ server1.Connect(ipep1); }

catch (SocketException exceptie1)

{ return; }

server1.Send(Encoding.ASCII.GetBytes(Target.Text.ToString()));//transmit Target

}

if (FireUnit2Values.Checked)

{

try

{ server2.Connect(ipep2); }

catch (SocketException exceptie2)

{ return; }

server2.Send(Encoding.ASCII.GetBytes(Target.Text.ToString()));//transmit Target

}

Target.ForeColor = System.Drawing.Color.Blue;

}

The choice and transmission of the target type was made according to the following lines of code:

public Client()

{

InitializeComponent();}

private void attackHelicopterToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Attack helicopter";}

private void comancheHelicopterToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Comanche helicopter";}

private void transportHelicopterToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Transport helicopter";}

private void navyHelicopterToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Navy helicopter";}

private void maneuvreHelicopterToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Maneuvre helicopter";}

private void standHelicopterToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Standing helicopter";}

private void motorAirplaneToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Motor airplane";}

private void reactionAirplaneToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Reaction airplane";}

private void f16JRoToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "F16J Ro";}

private void tacticalBombardierToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Tactical bombardier";}

private void parachuteToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Parachute shoting"; }

private void renegateAirplaneToolStripMenuItem\_Click(object sender, EventArgs e) {Target.Text = "Renegate airplane"; }

As a convention, it has been established that, after the successful connection and full data transmission, the font color will change from red to blue. Otherwise, the red color implies the appearance of a network-connection interruption.

3.4.3. Data conversion and display procedure

After completing the data transmission procedure, it is necessary to close the client-server connection and deallocate the memory. From a technical point of view, the procedure consists of:

1. closing the connection;

2. "graceful" termination of the connection;

3. deallocate the memory allocated to the socket.

For processes that share the same socket, the number of references to that socket is decremented, so that when it reaches 0, the socket is deallocated. Problems that can occur during data transmission-reception are:

- the server cannot end the connection, it does not know if and when the client will not send other requests;

- the client cannot know if the data has reached the server.

Next it is presented the program sequence through which the information received from the server is transformed into data and displayed on the client console, after which the connection is closed. The entire procedure is active as long as data is received from the Server; in other words, the connection is not closed until the default value takes the value 0, thus having the certainty that absolutely all data has been sent to the console.

while (true)

{

recv = client.Receive(data);

if (recv == 0) break;

if (label10.Text == "Target :") label10.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();//TARGET primita de la server

elseif (label12.Text == "Location :") label12.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (TRBox.Text == "") TRBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (RABox.Text == "") RABox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (RDBox.Text == "") RDBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (THBox.Text == "") THBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (TVBox.Text == "") TVBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (HDBox.Text == "") HDBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (ELBox.Text == "") ELBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (AZBox.Text == "") AZBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();

elseif (DIBox.Text == "") DIBox.Text = Encoding.ASCII.GetString(data, 0, recv).ToString();}

client.Close();

newsock.Close();

}

Also, it is presented a sequence from the source code through which, after checking the availability of the 2 fire control systems (client enable), the information about the target were transmitted (target number, direct distance, height, speed, etc.) and their transformation into data (necessary for the interpretation of the flight path, the interpretation of tactics and procedures etc.):

private void TR\_Click(object sender, EventArgs e)

{

TR.ForeColor = System.Drawing.Color.Blue;

if (FireUnit1Values.Checked)

server1.Send(Encoding.ASCII.GetBytes(TRBox.Text.ToString()));//transmit TR

if (FireUnit2Values.Checked)

server2.Send(Encoding.ASCII.GetBytes(TRBox.Text.ToString()));//transmit TR

}

if (FireUnit1Values.Checked)

{

server1.Send(Encoding.ASCII.GetBytes(DIBox.Text.ToString()));//transmit DI

server1.Shutdown(SocketShutdown.Both);//inchid FCU1

server1.Close();}

if (FireUnit2Values.Checked)

{

server2.Send(Encoding.ASCII.GetBytes(DIBox.Text.ToString()));//transmit DI

server2.Shutdown(SocketShutdown.Both);//inchid FCU2

server2.Close();}

}

The final analysis report of the combat actions (preparation of the technique) is constituted in the form of check type lists, with simple and efficient interpretation.

public Raport()

{

InitializeComponent()

private void button1\_Click(object sender, EventArgs e)

{

Client mynewform = new Client();

this.Hide();

mynewform.Show();}

private void Raport\_Load(object sender, EventArgs e)

{

textBox1.Text = Client.raportul1;

if (Client.raportul1[1].ToString() == "1") NorthDirectionCheckFU1.Checked = true;

if (Client.raportul2[1].ToString() == "1") NorthDirectionCheckFU2.Checked = true;

if (Client.raportul1[2].ToString() == "1") DOSCheckFU1.Checked = true;

if (Client.raportul2[2].ToString() == "1") DOSCheckFU2.Checked = true;

if (Client.raportul1[3].ToString() == "1") AOK1CheckFU1.Checked = true;

if (Client.raportul2[3].ToString() == "1") AOK1CheckFU2.Checked = true;}

}

If a procedure was completed successfully, then it appears in the report as a check. Any unfinished or even unsuccessful procedure appears as an uncheck on the activity report and leads to the nonfulfillment of the combat mission.

3.4.4. Use based on scenarios

A classic scenario of combating an air target is designed so that all elements of the anti-aircraft system Oerlikon Contraves 2x35mm work in a centralized regime: the SHORAR-TCP radar performs surveillance, search and research in airspace management, discovers and identifies the target (friend/ foe), and depending on the evolutionary parameters, the radar designates the target to the fire control unit responsible for area surveillance (AOR) which is ready for firing.

Within the GSN\_SIM simulation program, the execution of combat firings in 3 different locations was foreseen, each location having its own terrain characteristics and visibility. The instructor can also set the fire limits and safety limits (left/ right) depending on the location where the training takes place.

Another important element of the scenario is the type of air target and its direction of flight. Fixed targets and moving targets, left-to-right or right-to-left moving targets and fixed wing or rotary wing targets have been provided in the simulator database. Also, the target parameters (set by the instructor - the battery commander or by the operator) influence the effectiveness of anti-aircraft firing.

Problem scenario parameters includes:

- *Helicopter* target;

*- Costal* location;

- target number *15;*

- radar azimuth *1200*;

- radar distance *70*;

- target height *40*;

- target speed *30*;

- impact distance *40;*

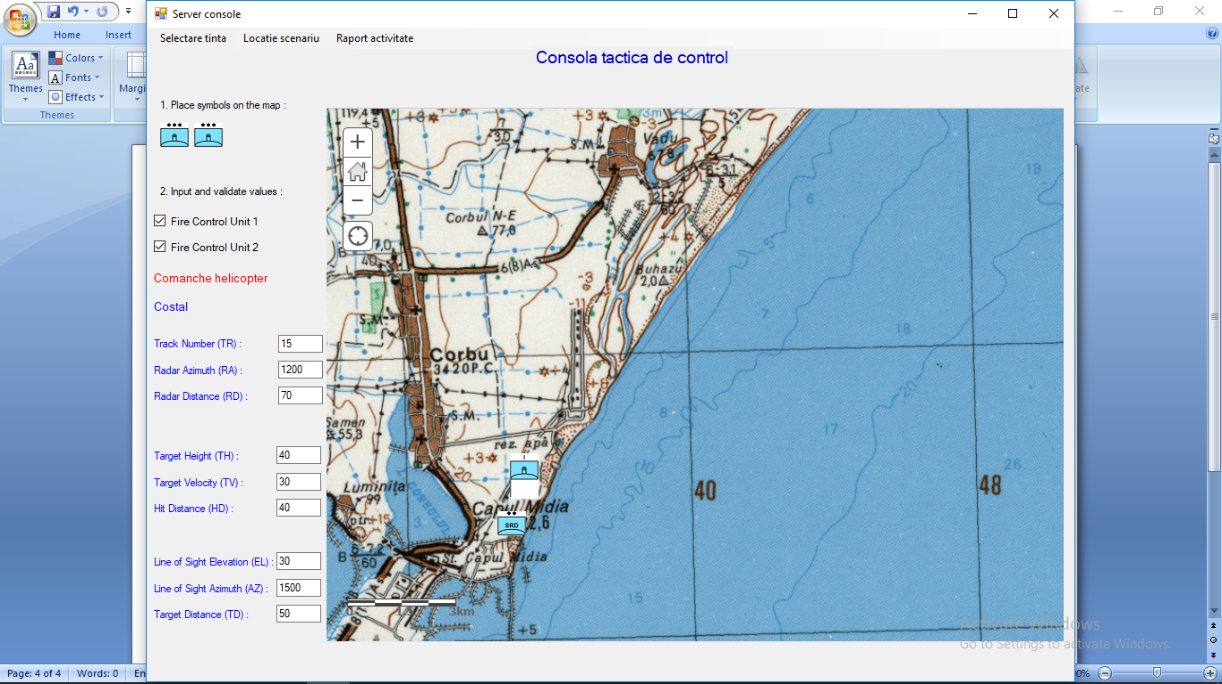
- elevation angle of sight line *30*;

- azimuth of the aiming line *1500*;

- distance to the target *50*.

The transmission of the target parameters to one or more fire control units is done within the decision-making process, in accordance with the best and most effective course of action, but also with the list of priority targets (value of the target).

Once the data is confirmed (parameters are colored blue), they are transmitted to the FCU console (it is possible to transmit to one FCU or both simultaneously) (Figure 5).

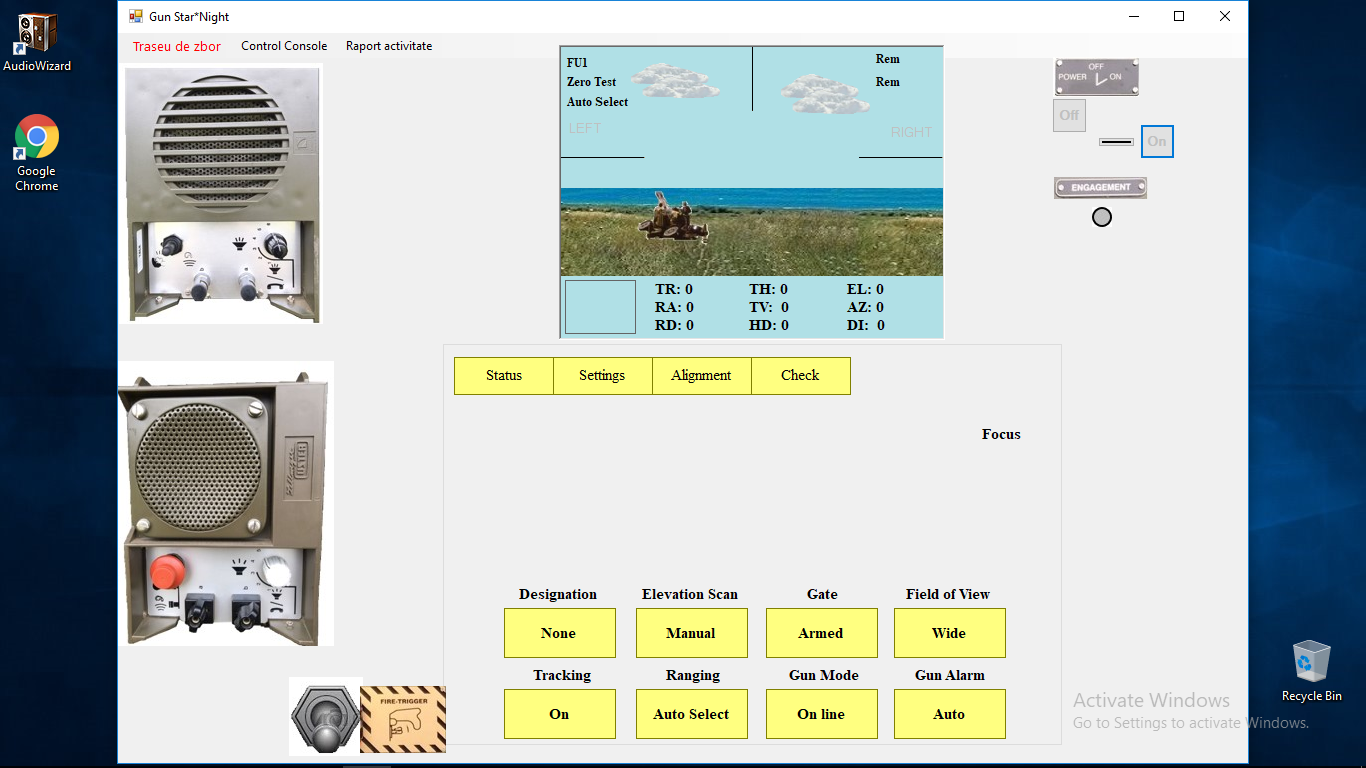


**Fig. 5** Graphical operator interface (after receiving data on the network)

The fire control device (GSN), after the process of designating the target by the SHORAR TCP or DOS, continuously accompanies and follows the aerial target with the help of the thermal chamber. The target parameters are constantly updated with the help of the laser rangefinder, they are analyzed and calculated by the central processor (DPU), and when the target enters the combat possibilities of the system, anti-aircraft guns (AOK's) can open fire, action led by to the fire control unit (FCU).

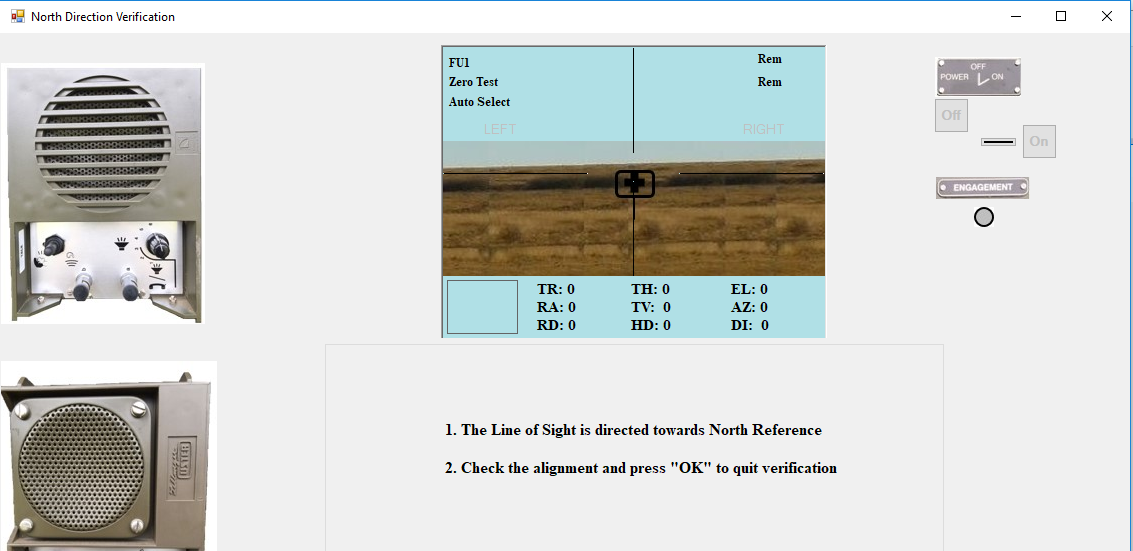
The entire training process of the operators is divided into three distinct phases and the transition to a later stage involves completing the entire training stage, the basic condition being the completion of at least 85% of tasks (tasks).

Phase I: identification, understanding and knowledge of the elements on the main frame (video screen, tach-panel menus and submenus, on/ off tumblers and FIRE) (Figure 6).



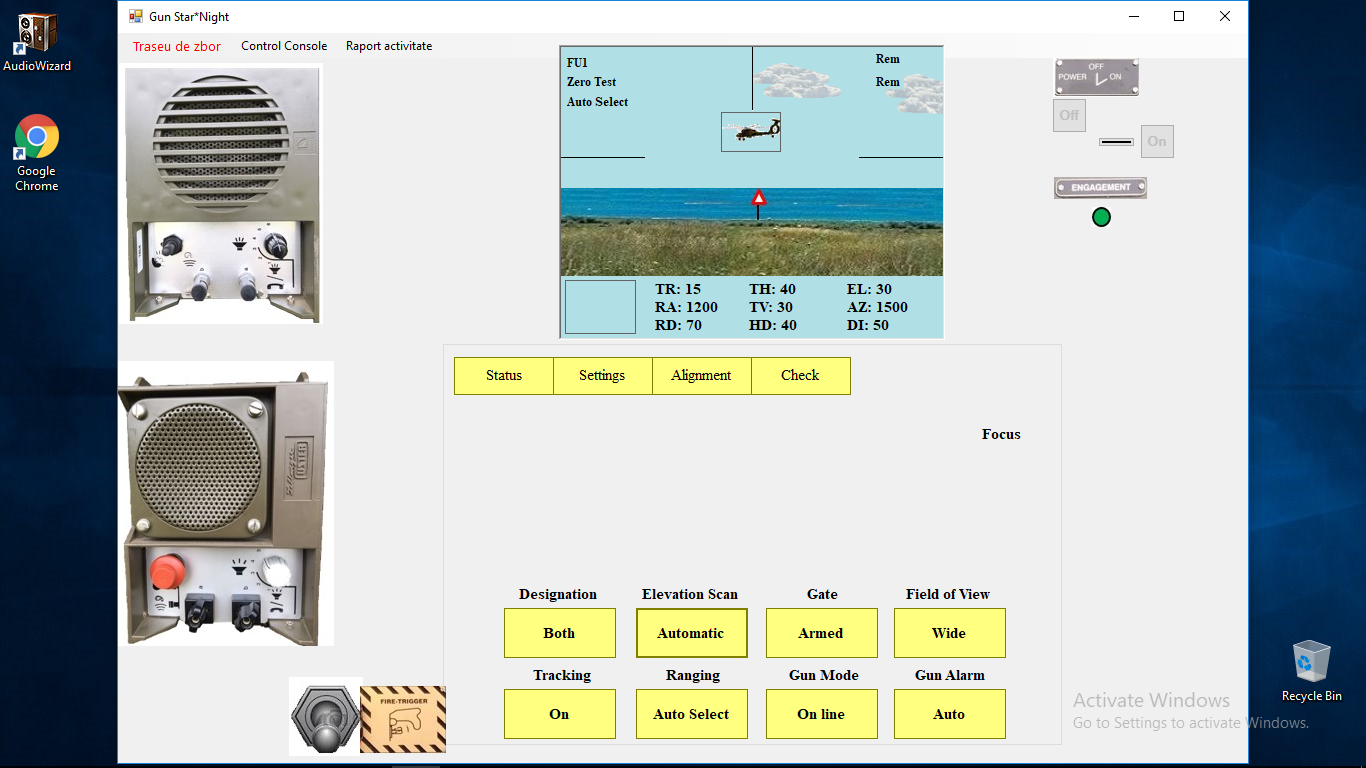
**Fig. 6** Main Menu

Phase II: knowledge and application of procedures for preparing the technique for combat - making alignments and checking them, correctly entering all parameters, entering values in accordance with the requirements of safety and security of shooting (Figure 7).



**Fig. 7** Alignments

Phase III: combating or destroying the distributed air targets, within the safety limits and within the predetermined time interval (Figure 8).



**Fig. 8** The flight path

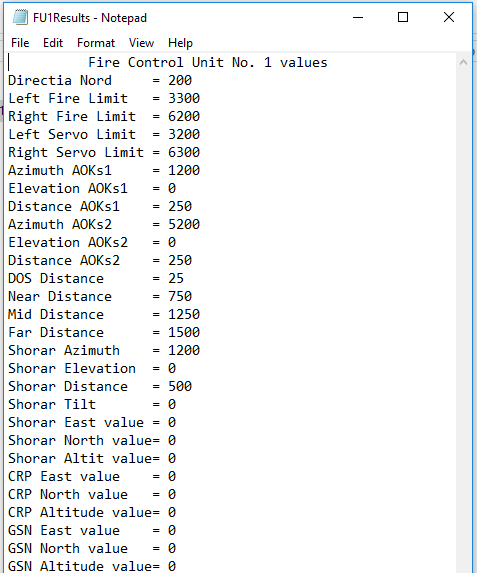
Another important function of the fire control system simulation is to generate, if all stages of the preparation of the combat technique have been completed, a probability of fighting/ hitting the air targets. The main key that determines the probability of hitting is the time of evolution of air targets in the firing area. This time is determined by the speed of the target, the height at which the target evolves and the time interval in which the operator opens fire on the target, directly influencing the number of projectiles that can be fired at the target.

Of course, the probability of hitting a supersonic plane that is moving at medium altitude and high speed will not be the same as the probability of hitting a helicopter that is flying at low altitude and low speed. The probability of hitting decreases even more if the operator fails to fire more than 1 to 2 salvos on the target.

A last variable in the calculation of the probability of hitting the target is a random variable: in addition to the time spent in the fire zone and the number of salvos fired by the operator, this variable was introduced in the system, a variable that influences the small measure the result of the shooting, but enough to lead to the fulfillment or nonfulfillment of the mission. This random variable makes a randomization of probability, an adjustment plus or minus by a few percent and its role is to capture the change in shooting conditions (e.g. weather parameters).

In conclusion, the simulation system takes into account the values of the probability of hitting different types of targets, depending on the number of projectiles fired, the evolution of the target and the reaction speed of the operator. But because in the real environment nothing can be strictly reduced to an exact mathematical formula, because the values of probabilities are not the same, in an attempt to solve the problem of "meeting the projectile with the target", we introduced the random factor, the coefficient of organized chaos, an eternal constant companion of anti-aircraft gunners and missiles.

To perform the post-action analysis, the simulator offers the possibility to the commander/ instructor to view an activity report for each FCU, in which the actions performed by the operator are marked. Also, you can view/save/print at the level of each operator a report that includes the recorded values of the parameters.



**Fig. 9** Activity report

The control and verification of the observance of the procedures for preparing the combat technique and the alignments with the vital subsystems can be done by accessing the *Activity Report* menu (Figure 9) and the data entered in the system are analyzed using the file *FU1Results.txt* or *FU2Results.txt*, from the folder *Reports*.

In summary, GSN\_SIM allows the use in two modes:

A. **Independent** - for learning, training basic skills and training the FCU operator;

B. **Centralized** (networked) - to train and strengthen the cohesion of the combat team in the decision-making process in combating air targets.

For the **independent** work regime, the operator carries out the following preparatory activities:

- accessing GSN SIM FCU1;

- selecting the type of aerial target from the *Target selection* menu;

- selecting the scenario location from the *Scenario location* menu;

- entering the target parameters in the text boxes, according to the scenario of the exercise

After entering all the parameters, their validation is done by pressing the *Submit* button, the button below the parameter fields. After transmitting the parameters to the FCU interface, the command and control (C2) window automatically closes and the FCU start window opens.

For the **centralized** operating mode, the instructor/ battery commander and the FCU operators carry out the following preparatory activities:

- accessing GSN SERVER;

- accessing GSN SIM FCU1 and GSN SIM FCU1;

- FCU operators press the *Network* button to receive the target parameters;

- the instructor selects the type of aerial target from the Target selection menu and the scenario location from the *Scenario location* menu;

- selecting the scenario location from the *Scenario location* menu;

- enter the target parameters in the text boxes, according to the exercise scenario;

- after entering the last parameter, they will be automatically sent to the FCU, and the parameter names will be colored in blue.

***3.5. Conclusions***

Education and training, two of the pillars of the professional completion of any military, go through a stage characterized by systemic changes: innovative digitalization, the requirements of skills and abilities, technological changes. The demands of the operational environment seem to outline a possible dilemma: we will witness a competition with a winner or a coordinated and integrated functioning of people and technology.

In this context, a first solution could come by applying an emerging leadership model, which means first of all creating the synergy of people - processes, understanding the interdependencies between personal development and organizational development. In this sense, changing organizational strategies and training/ retraining the staff in accordance with the new requirements of competencies and skills are imperative for decision makers.

The need to redesign the training and development process to keep up with new technologies is already felt in the military. A possible approach could follow two main directions:

1. preparation and training in line with the challenges posed by technological progress and the requirements of the mission; an education that includes both STEM (Science, Technology, Engineering and Mathematics) and CORE (Creativity, Originality, Reciprocity, Empathy) elements;

2. professionalization of personnel through different forms of training starting from the opportunities that automation will offer (transformation of military units into true learning organizations).

In the context in which the competition for obtaining advantages in the technological field is awfully close, the human factor is the one that could make the difference on the battlefield, and the level of training is the result of the military educational process and specific training.

GSN\_SIM has been designed to respond to changes in the operating environment or to improve technical performance through modular adjustments. GSN\_SIM is an HLA not only from the perspective of data flow, but also from the perspective of interoperable open architecture:

1. **Runtime Infrastructure (RTI)** is represented by the technical part of the program, by the programming platform (Visual C#) where the lines of code, the functions, the procedures, the definition of the input and output variables were written, where the connection and transmission/ reception of data techniques were implemented and their transformation into information took place.

2. **The Federate** is the socket interface - the technical part at a network level - which connects the simulator itself (GUI graphical interface, frames etc.) to the programming platform and has the main role in transmitting data in the network.

3. **Simulation Object Model (SOM)** is represented by the dedicated frames, those frames that actually contribute to the retrieval of data from the input peripherals (keyboard or virtual camera), transforms them into information and sends them to GSN CLIENT information related to the target speed, height, azimuth etc.) or to .txt files, output files, for interpretation (interpretation of the way in which the standard operating procedures were followed).

4. **HLA Federation** represents all the frames in a simulation system (GSN SERVER or GSN CLIENT), regardless of their role (graphical interface, data retrieval, data processing, interpretation, commissioning of various tools etc.).

All processes running on the two federations are interconnected at this level, although in the first stage these processes run independently, without connection to each other, and the interconnection occurs when the connection between the two federations opens.

With the opening of the connection, the exchange of data and their transformation into information takes place, so that, at the moment when the connection (from a technical point of view – network) has closed, the network protocols have ceased their activity 100% and it returns to the local type operation. In other words, the whole process related to the networking area takes place at Federation level.

5. **The Federation Execution** represents the final session, at the highest level and is represented by the operation of Federations, by the operation, as a whole, of the GSN\_SIM simulator by running/ putting into operation both simulators/ consoles:

- console/ simulator of the subunit commander, with the part of preparation and organization of the tactical situation, introduction of data about the target, organization of the tactical position and the elements of the subunit etc.;

- the subunit commander's console/ simulator (primary target), with the preparation of the fighting technique, in relation to the data received from the battery commander's console, their running and interpretation and the actual conduct of the fight.

The advantages offered by GSN\_SIM are multiple: training of operators in modern conditions; significant saving of resources; use in any weather conditions; effective assessment tool for the acceptance to firing of combat teams.

Future research directions include connecting GSN\_SIM with other simulators, such as Microsoft Flight Simulator or JCATS (Joint Conflict and Tactical Simulation), in a standard HLA configuration.

**References**

[1] Alexa, M., and Cioaca, C. (2017). Study of High-Level Architecture Applicability in Air Defense, Scientific Research and Education in the Air Force – AFASES 2017, Brasov, 15-22.

[2] Sennersten, C. (2010). *Model-based Simulation Training Supporting Military Operational Processes,* School of Computing, Blekinge Institute of Technology, Sweden.

[3] Coman, M.M., and Cioaca, C. (2019). Provocari manageriale tehnice privind implementarea modelarii si simularii in cadrul procesului educational militar, in *Provocari manageriale in domeniul securitatii si apararii,* Coord. Badea and Iancu, ‘Nicolae Balcescu’ Land Forces Acade[9] my Publishing House, Sibiu.

[4] Popescu, V.F. (2020). C5ISR-D Revolutionary Systems and Processes, *Romanian Military Thinking,* no. 1/ 2020, 166-176.

[5] Hodson, D. D., and R. R. Hill. (2013). The Art and Science of Live, Virtual, and Constructive Simulation for Test and Evaluation, *Journal of Defense Modeling and Simulation* 41 (1): 6–19.

[6] van der Pal, J., Keuning, M., Lemmers, A. (2011). *A comprehensive perspective on training: live, virtual and constructive*. NATO Modeling and Simulation Group (NMSG), Symposium ‘Enhance of Replace: Finding the Right Live vs. Synthetic Balance’. Retrieved from: https://www.researchgate.net/publication/264924595\_A\_Comprehensive\_Perspective\_on\_Training\_Live\_Virtual\_and\_Constructive/citations

[7] US Department of Defence (2011). *Modeling and Simulation (M&S) Glossary.* Retrieved from http://www.acqnotes.com/Attachments/DoD%20M&S%20Glossary%201%20Oct%2011.pdf

[8] Hill, R., and Miller, J.O. (2017). A History of United States Military Simulation, Proceedings of the 2017 Winter Simulation Conference W. K. V. Chan, A. D’Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer and E. Page. Retrieved from: http://simulation.su/uploads/files/default/2017-hill-miller.pdf

[9] Fujimoto, R. M. (1998). *The Management in The High-Level Architecture*, Computational Science and Engineering Division, College of Computing, Georgia Institute of Technology, Atlanta.

[10] IEEE New Standards Committee (NESCOM) (1997). Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Object Model Template (OMT) Specification, IEEE Project Authorization Request (PAR) 1516.2.

[11] Zeigler, B.P., Praehofer, H., Kim, T.G. (2000). *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*, 2nd Edition, Academic Press.

[12] [Dahmann](https://www.semanticscholar.org/author/J.-Dahmann/32653941), J., and [Morse](https://www.semanticscholar.org/author/K.-Morse/2854758), K. (1998). *High Level Architecture for simulation: an update*, Proceedings 2nd International Workshop on Distributed Interactive Simulation and Real-Time Applications (Cat. No.98EX191), 32-40.

[13] IEEE Standard Association (2010). IEEE 1516.1-2010 - IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification. Retrieved from: <https://standards.ieee.org/standard/1516_1-2010.html>.

[14] Pitch Technologies AB (2012). *The HLA tutorial a practical guide for developing distributed simulations*. Retrieved from: <https://vdocuments.mx/the-hla-t-pitch-hla-tutorial-a-practical-guide-for-developing-distributed-simulations.html>.

[15] Oerlikon Contraves (2002). *GDF – 103 AA Drill book Oerlikon*, Oerlikon Contraves Publishing House.

[16] Romanian Ministry of Defence (2008). *Air defence artillery operator’s training manual – Land Forces*, 35 mm Oerlikon System, Romanian MoD Publishing House.