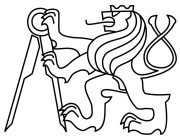
CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF INFORMATION TECHNOLOGY



ASSIGNMENT OF MASTER'S THESIS

Title:	Interactive Network Simulator for Analysis and Visualization of Protocols
Student:	Bc. Pavel Goncharov
Supervisor:	Ing. Alexandru Moucha, Ph.D.
Study Programme:	Informatics
Study Branch:	Computer Systems and Networks
Department:	Department of Computer Systems
Validity:	Until the end of winter semester 2018/19

Instructions

The tools for network topology representation and analysis belong to one of the following groups:
Network Simulators: allow the drawing of the network topology and the simulation of its behaviour, are difficult to use and require learning.
Static Diagram Drawing Tools: represent the topology as a non-interactive diagram, are simple to use but lack any practical application as they cannot simulate anything.
Network Topology Mappers: allow to scan a physical network and represent it as a static diagram.

Design and implement a Windows/Linux desktop application that can:
- -construct a network topology in the form of an interactive diagram,
- -modify and reconfigure the topology at run time,
- -save/load state of the network,
- -simulate a selected set of the most important network protocols according to their formal specifications,
- -simulate network events and observe the responsive behavior of the network,
- -be easily used and quickly understood by new inexperienced users.

References

Will be provided by the supervisor.

prof. Ing. Róbert Lórencz, CSc.
Head of Department

prof. Ing. Pavel Tvrdík, CSc.
Dean

Prague May 24, 2017

Acknowledgements

I would like to thank Ing. Alexandru Moucha, Ph.D., for his support and guidance starting from the early planning stage. His advice and experience have helped to shape this Thesis. I would also like to thank my colleagues at work for their patience and encouragement throughout this research.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on 29th December 2017 .....

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Goncharov, Pavel. *Interactive Network Simulator for Analysis and Visualization of Protocols.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2017.

Abstrakt

Vývoj síťových technologií je hnací silou moderní inovace a pokroku v mnoha klíčových oblastech. To má významný dopad na množství průmyslových odvětví a globální ekonomiky jako celku. Mezinárodní korporací, vláda a akademická obec stále nacházejí nové praktické aplikace pro tuto rychle se rozvíjející technologii. Požadavek na nástrojů pro návrh a plánování sítí, specifické pro danou problematiku, stále roste, protože rozmanitost praktických aplikací se zvyšuje. Simulace sítě zůstává jedním z nejdůležitějších přístupů k návrhu sítě, ale musí se neustále přizpůsobovat novým požadavkům. Tento výzkum se zaměřil na analýzu hardwarových a softwarových síťových komponent s cílem navrhnout a implementovat zcela nový flexibilní uživatelsky přívětivý síťový simulátor, který lze snadno přizpůsobit a rozšířit pro různé účely. Tato diplomová práce nakonec vyústila ve vytvoření komplexního rámce pro autentickou simulaci hardwarových a softwarových komponent s několika síťovými protokoly, které jsou již zabudovány: Ethernet, ARP, IPv4, TCP, UDP, RIP, EIGRP. Schopnost rychle a efektivně implementovat a vyhodnocovat jakékoliv síťové protokoly a hardware podle požadovaných specifikací jsou klíčovými silnými stránkami nově vyvinutého simulátoru, který mu jistě umožní předat test času. Plánuje se začlenit tento síťový simulátor do vyučovacího procesu, stejně jako jeho případné uvolnění jako projekt spolupráce s otevřeným zdrojem pro další vývoj.

Klíčová slova návrh sítě, síťové plánování, návrh topologie, návrh síťového protokolu, vyhodnocení síťových protokolů, síťová simulace, Agilní vývoj, Unreal Engine 4, TCP\IP model, Ethernet, IPv4, IPv6, ARP, UDP, TCP, RIP, EIGRP, BGP.

Abstract

The evolution of networking technology has been the driving force behind modern innovation and progress in many key areas. It has made a significant impact on the number of industries and global economy as a whole. International corporations, government and academia keep finding new practical applications for this rapidly advancing technology. The demand for problem-specific set of network design and planning tools keeps growing as the variety of practical applications increases. Network simulation remains to be one of the most relevant approaches to network design, but it has to constantly adapt to the new requirements. This research focused on analysis of hardware and software network components with a goal to design and implement a brand-new flexible user-friendly network simulator which can be easily adapted and extended for various purposes. This Thesis has ultimately resulted in development of a comprehensive framework for authentic simulation of hardware and software components with several network protocols already built into it: Ethernet, ARP, IPv4, TCP, UDP, RIP, EIGRP. The ability to quickly and efficiently implement and evaluate any network protocols and hardware according to the required specifications are the key strong points of the newly developed simulator which will most certainly allow it to pass the test of time. There are plans to incorporate this network simulator into the teaching process as well as to potentially release it as an open-source collaboration project for further development.

Keywords network design, network planning, topology design, network protocol design, network protocol evaluation, network simulation, Agile development, Unreal Engine 4, TCP\IP model, Ethernet, IPv4, IPv6, ARP, UDP, TCP, RIP, EIGRP, BGP.

Contents

List of Figures

Introduction

Modern Internet is constructed from innumerable number of individual interconnected networks scattered across the world. They range from small private ones all the way up to highly sophisticated corporate and government networks. It is now hard to imagine ordinary day-to-day activities without network connectivity. Global economy becomes more dependent on on-line client-facing technology with each passing day. International corporations together with small private companies invest significant resources in order to adapt their business model to the new paradigm. While interest in these emerging technologies is undeniable, we can't ignore the fact that tools for network design and planning are not capable of meeting growing demand.

The expansion of world wide web requires constant innovation, experimentation and adaptation. The methodology and tools which proved to be capable in the past are often unable to keep up with evolving requirements. The growing number and variety of devices communicating over network creates a demand for specialized design and planning software, capable of performing a specific set of tasks, while been more flexible and free of overwhelming complexity.

State of the Art

Tools for network design and planning can be divided into the following categories:

- Static Diagram Drawing Tools: able to represent network topology as a static, non-interactive diagram, which could be useful, but is insufficient
- Network Topology Mappers: capable of scanning existing networks and creating static diagrams. We can use topology mappers for analysis of existing networks, but it won't help us with construction of the new ones

- Network Simulators: capable of constructing topology and simulating network events. They are typically complex and require special training from the user. Notable network simulation software includes: NS3, OPNET, OMNeT++, REAL, QualNet, J-Sim, NetSim

As we can see, network simulation is the most suitable approach to topology design and protocol analysis. Even though several tools exist in this category, they can all be characterized by having at least one of the following drawbacks:

- Limited selection of tools
- High level of complexity
- Difficult to use
- Basic graphical user interface
- Lack of visual feedback

If we take a look at existing network simulators, both commercial and open-source, we can see that they are not able to meet demand of industry and academia in several areas. It can be explained by high cost and complexity associated with development of such applications. In order to overcome these obstacles and deliver required tools, a different approach should be taken. The development of network simulation software with a smaller scope, but highly focused and optimized for performing specific tasks is an optimal solution to this issue. An analysis of existing tools, use-cases and software requirements should enable us to create efficient, state-of-the-art network simulator capable of solving the problems outlined in this work.

Structure of the Thesis

First chapter of this Thesis provides detailed analysis of existing methods and tools for design and implementation of complex software projects. It describes Agile methodology and compares it with alternative software development approaches. This chapter covers history, capabilities and advantages of Unreal Engine over alternative development tools.

Second chapter of this work is focused on definition, specification and implementation of TCP/IP model. It covers Link, Internet, Transport and Application layers of TCP/IP model and corresponding components. It includes specification of network devices, protocols and their respective implementation

in the scope of New Network Simulator. This chapter also covers development approach and tools used for implementation of graphical user interface.

Third chapter of this research project describes features and capabilities of developed New Network Simulator. It includes implementation details of available components and functionality. This chapter describes practical use-cases and provides user with instructions on how to use New Network Simulator to construct and analyze flexible efficient networks.

Chapter 1

Development Tools and Methodology

This chapter describes tools and methodology used during development of New Network Simulator. It covers advantages and disadvantages of existing development tools and methods. It also explain why these particular tools and methodology have been chose for development over others.

1.1 Development Methodology

This section defines Agile development methodology and analyzes its advantages and drawback in comparison with other approaches. It explains why Agile has been identified as the most suitable and practical software development methodology for this project.

1.1.1 Agile Development

Agile development can be defined as a set of fundamental principles and guidelines for the development of software projects. The main concept behind Agile lies in small independent flexible teams, closely cooperating with each other [1]. This approach promotes adaptive software planning, iterative development, delivery of functional prototypes and efficient adaptation to rapidly changing requirements [2]. This development methodology first introduced in Agile Manifesto [3] and later on became extremely popular among developers. There are several different Agile framework, among which Scrum and Kanban are especially popular.

We can track the roots of iterative software development approach all the way back to 1957. It started to gain popularity in 1990s, when cumbersome,

highly regulated and fixated methods were heavily criticized. This has resulted in a high demand for alternative lightweight development methods [4]. The Agile manifesto itself has been formulated by a group of 17 developers in 2001. The meeting took place in USA, Utah during which initial principles and guidelines for Agile have been established. In 2011 best practices of Agile have been extended by Agile Alliance through publishing of Agile Glossary. The Glossary itself is an ope-source knowledge base of constantly adapting agile definitions, values, elements and practices supported by agile enthusiasts all over the world.

Original Agile Manifesto incorporates the following principles:

- Focus on collaboration and independence instead of strict hierarchy and control structure
- Delivery of practical, deliverable prototypes after each iteration over creation of comprehensive documentation
- Close and continuous collaboration with a customer instead of trying to define all requirements during initial project planning stage. This in turn enables adaptive response to change.
- Flexible development methods, capable of quick adaptation to new requirements [5]

Agile Alliance considers Agile movement as a way to refresh and redefine software development methodology, rather than remove it as some critics claim. The main goal of Agile is in fact to bring a balance to the development process and optimize the cost. The removal of cumbersome documentation while keeping only relevant information and enabling development process to cope with changing requirements are among the main advantages of Agile.

Agile Manifesto has originally been based on the following principles:

- Ensuring customer satisfaction by iterative delivery of deployable software builds
- Adaptation to evolving requirements in all development stages
- Frequent software delivery
- Active cooperation between development team and clients
- Trust in developers to work without strict supervision
- Keeping project participant in the close proximity to ensure face-to-face communication

- Measuring project progress by looking at latest build
- Ensuring stable development environment and constant progress
- Paying attention to design standards and continuous optimization
- Keeping it simple
- Letting individual teams to find the best methods, design and practices suiting their needs
- Continuous self-reflection of teams on the achieved results
- Keeping development processes flexible, iterative and innovative [6]

Majority of Agile methods tend to divide project development into smaller parts in order to decentralize design and planning among small teams. Development is carried out in iterations also called sprints, which are typically 1 to 4 weeks long. Iterations include all phases of software development such as design, planning, implementation and testing. After each iteration a working deployable prototype is delivered. This package is then used to demonstrate overall development progress and is shown for all stakeholders. Scheduled demos of newly implemented features are used to gather feedback from stakeholders. This process enables flexibility and room for requirements modifications at the early stages [7]. The significance of features added at the end of iteration cycle may not necessary justify production release, but it allows to make more room for adaptation.

In all cases, the teams are expected to include product owner, who act as representative of customer. Product owner is expected to keep very close contact with all stakeholders and be able to answer business-related questions which developers might have. The most important role of product owner is to insure that customer's needs are met and to act as a negotiator between developers and customer.

Agile team typically doesn't have a strict hierarchy between developers. Every team members is supposed to cooperate within the team an make independent responsible implementation decisions required to achieve established goals. Scrum Master is often appointed to help coordinate the team and improve communication between developers. Scrum master is mostly responsible for organizing daily stand-ups and sprint reviews but acts as a any other developer in the meantime.

Daily stand-up, also known as scrum meeting is crucial to make Agile development possible. Scrum meetings take place every day at the same time, even if some of the team members are not present. It is usually fifteen to thirty

minutes long and is used to exchange quick progress updates between developers. Daily stand-ups help to keep all team members up to date and provides everybody a chance to express their concerns and problems which hinder the development. Scrum master is responsible to take notes of individual reports in order to address them alter on.

Agile software development heavily relates on information board available to all developers and product owner. Development boards contains all tasks, bugs and defects which have to be taken care of. The main purpose of the board is to allow task tracking and assigning. It serves as summary of development progress and is the most important tool for project management. The board could be a physical object, divided into swim-lines and containing paper-notes representing tasks. In practice specialized software for Agile development is used instead [8].

Agile software development makes use of pair programming, refactoring, automated testing and continuous integration in order to increase efficiency and productiveness of development process [9]. Agile approach is one of the most suitable solutions for development of projects involving non-deterministic processes and evolving requirements. It is often difficult to provide good estimates of time and resources required for development.

Agile approach uses iterative process to make estimate more accurate with each cycle. As the project requirements evolve, team members have to adapt as well. In most cases Agile developers avoid overestimating their capabilities in order to minimize the risk and corresponding financial loses. These development principles have been established over the years in response to criticism of heavy development methods often failing to estimate project complexity [10].

Agile software development methods belong to adaptive category, which can be characterized by flexible planning and establishing of dynamic milestones [10]. Overly distant goals can lead to poor estimation accuracy, while reduced time ranges make it easier to measure complexity of tasks.

There is a broad range of Agile development frameworks, among which some of the most popular are:

- Agile modeling
- Kanban
- Scrum
- Scrumban

Agile methodology has been adopted in order to develop New Network Simulator. Long term project planning and design was identified as been infeasible in the early stage. Agile approach has made it possible to adjust implementation requirements as research moved forward. The development was split into one week long iterations, each producing functional prototypes. Agile has proved to be the right choice and has successfully delivered new innovative network simulator.

1.2 Development Tools

This section describes obstacles associated with development of New Network Simulator. It provides a comprehensive analysis of Unreal Engine 4 capabilities and explains how & why it was adopted for development.

1.2.1 Unreal Engine 4

Unreal Engine 4 is a complex comprehensive set of tools for game development. The very first version of the engine, Unreal Engine 1, was introduced in the late 1998. It was used originally designed for development of first-person shooters followed by a variety of game genres. The code-base of Unreal Engine is written in C++, providing high efficiency and portability for developers [11]. Unreal Engine has been a very successful project, wining numerous awards, such as Guinness World Record for the most successful game engine in 2014 [12].

From its inception, Unreal Engine supported collision detection, rendering, lighting effect and basic texture filtering. The lightning featured support of light sources and real-time illumination. The very first version already included the central tool behind the engine - Unreal Editor. The early version of Editor allowed on-the-fly layout mapping and geometry construction [13][14]. Starting from 2000, Epic corporations has added support for skeletal mesh animations and high-scale terrain editing [6].

The modular architecture of the engine and a specialized language - UnrealScript have made nodding accessible to many people. The support for modding has greatly boosted engine's popularity and attracted many developers. Unreal Engine has been designed as a set of separate independent modules. Such modular structure made engine very flexible and extensible, making it future proof.

According to Epic Games founder Tim Sweeney, Unreal Engine was developed to be as flexible as possible. The code-based is being extended by new

features over many iterations, which keeps it relevant and up to date. This in turn requires Epic developers to ensure that base-code is not obfuscated and modular. This approach have been part of the foundational design of the engine and allowed higher flexibility for licensing contracts.

Modern iteration of Unreal Engine, version 4 supports software development for almost every major platform, including:

- MS Windows
- Unix/Linux
- HTML5
- iOS and Android mobile platforms
- Game consoles such as PS4, Xbox One and Nintendo Switch
- Virtual reality headsets: Oculus Rift, Google Daydream, Samsung Gear VR and HTC Vive

Unreal Engine 4 was introduced during Conference of Game Developers in February 2012. An impressive demo of Engine's capabilities have been prepared and showcased on a personal computer with three Nvidia SLI-enabled GTX 580 graphics cards [15]. Among major features, it demonstrated high quality global illumination implemented with the help of cone tracing and voxel, therefore replacing static lightning effects.

Unreal Engine 4 has introduced several features intended to reduce development time. Among them hot-reload feature has allowed to recompile and reload C++ modules without the need to restart the engine. One of the significant included additions to the engine was Blueprint visual programming language, which allows compilation of C++ classes and functions into blueprint diagram-like components.

This compiled blueprints can the be used to design a very high level, self documenting code. Unreal Engine 4 has also enabled support for live-debugging, which allows to trace program executions using blueprints logic while it runs [16]. Some of the most significant outcomes of these additions was high level of integration and smaller development iteration cycles. The flexibility of Unreal Engine 4 has made it easier for programmers, designers and animators to collaborate on the project [17]. In comparison to other engines of that time, Unreal Engine 4 has helped to eliminate long waiting time between compilation and testing, making development much more efficient.

Engine was originally available only for large-scale corporations for a very high cost of several millions dollars. The emergence of indie development studios and ever growing community of developers has lead Epic to change its business model. Unreal Engine 4 has officially been available for the general public in March 2014. This included usage of all its features, modules and C++ code-base under subscription [18]. This new business model was a response from Epic corporation to the evolving business model and rapid growth of the industry.

A mew Marketplace has been introduced in September 2014, allowing users to share or sell their assets to other developers. This has made a big impact on the development community, creating an incentive to contribute to the Unreal Engine 4 as well as to start development of smaller independent projects. Epic has also released a large number of completely free assets, including entire packs of models, textures, sound effects, animations and level layouts. Unreal Engine 4 as also allowed developers to gain access to content used for all impressive Epic demos [19]. Unreal Engine 4 is supported by a rapidly growing community of people, providing help to both inexperienced and professional developers. There is a large number of tutorials available for people willing to join Epic community.

Looking at the success of Epic's new business model, they have a made another step towards brining of the engine to more people. In the same month when they introduced asset Marketplace, Unreal Engine 4 have been made available to academia for free. Free copies of the engine have been provided to universities and several schools to engage students working on simulation software, game development, architecture and computer science [20]. Several grants have been granted to developers using Unreal Engine 4, allowing everybody to have a chance in sponsoring their projects [21].

Finally after more than a decade since its inception, Unreal Engine has become available for free to everybody, including all future releases [22]. Epic has based its business model around into royalty fees, paid by the developers who use Unreal Engine 4 to make profit. The royalty fee only applies to developers making a considerable profit and differs based on project scale. In addition Oculus Store has announced that it will pay royalty fees on behalf of the developers whose products gross revenue doesn't exceed five million US dollars. This decision has become one of the factors behind very rapid adoption and support for virtual reality headset in Unreal Engine 4.

Epic corporation is also working on several titles using Unreal Engine 4. New tools and features developed in order to deliver these products are been included in future release of the engine, and therefore improving its capabilities.

In order to avoid potential misconceptions, especially taking into account that Unreal Engine 4 has served as a prime development tool for the New Network Simulator, it should be mentioned that game industry is not the only area where Unreal Engine is actively used. The areas in which Unreal Engine 4 is commonly used include, but not limited to:

- Simulation software tools
- Architecture design
- Production of special visual effects
- Creation of trailers and movies
- Motion capture and green-screen technology integration
- Heavily used by various government agencies as a part of Unreal Government Network project

Some of the few examples of creative ways in which Unreal Engine has been used over the years include simulation of crime scenes by the FBI Academy [23]; simulation of transportation system by Michigan Department of Transportation [24]; simulator for training of techniques intended to counter improvised explosive devices by IPKeys Technologies [[25]]; development of virtual reality plug-in to be used by United States Air Force; development of training platform for Enhanced Dynamic Geo-Social Environment by United State Department of Homeland Security [26] and many others.

It can be said, that Unreal Engine 4 is among the most advanced technology for development of complex software projects for multiple platforms. It provides tools for graphical design and support for visual programming, powered by C++.

New Network Simulator is a very ambitions and complex project, which would not have been realized without state of the art development tools. The use of Unreal Engine 4 libraries and tools has made a tremendous impact on development time and efficiency especially in terms of GUI and user interactions with simulated devices. In addition Blueprint system of the engine has made it possible to design and plan New Network Simulator in a very unique way.

Visual programming made it possible to design application logic as a self-documenting schema, representing the relation and dependency between components, which could be implemented afterwards. The hot-reload of engine modules, has made it possible to implement several custom components required to deliver fundamental features of New Network Simulator. These

components have been compiled and integrated into the code-base of Unreal Engine 4 itself, making development more efficient and flexible. In addition, availability of Unreal Engine 4's source code has made it possible to build the engine for Windows and Linux platforms, which in turn made it possible to deliver native optimized versions of New Network Simulator for both platforms.

Chapter 2

Design And Implementation

This chapter provides in-depth details of design and implementation of New Network Simulator. The chapter is divided into several sections, each covering a specific TCP\IP layer. Each section includes definitions of hardware and software components as well as their implementation details in the scope of New Network Simulator. It describes tools and design choices used for implementation of graphical user interface

2.1 Link Layer

This section provides specifications of networking hardware components and link layer protocols. It provides implementation details and design choices associated with development of underlying components.

Link layer of TCP/IP model covers network components of local area network which a capable of communication without intermediate routers. Link layer protocols are mainly used to interconnect network topology and support network interfaces. Internet layer is using datagrams to send data between neighboring devices. The scope of this layer is limited to local area network directly connected to the host. This type of connection is typically refereed to as link which is also reflected in the name of this layer.

Link layer is the lowest layer of TCP/IP model and serves as a foundation for the whole network. Higher layers of the model are required to operate independently of the link layer components and therefore support all networking hardware types.

The main purpose of TCP/IP link layer is transport of data packets, refereed to as datagrams. The transmissions mechanism for sending and receiving packets over the direct connection is managed with the help of network card

drivers and firmware. Modern networking devices often make use of specialized hardware components for transmission control which greatly speeds up this process.

A typical scenario taking place on the link layer include such activities as preparation and appending of packet headers to prepare for the start of data transmission; the transfer of packets over link connection to the next neighbor; as well as processing of received datagrams on the other side of physical medium.

TCP/IP model defines Media Access Control address used to identify network interfaces on the link layer and includes methods for network address translation. Link layer doesn't explicitly specify additional requirements beyond that.

The control mechanisms of virtual private networking take place on the link layer. This type of connection can be established with a help of transport or application layer protocols. The main purpose of these protocols is to establish a tunnel for operation of virtual private network. TCP/IP model in general doesn't enforce a strict hierarchy of data encapsulation. In the ISO/OSI model link layer is represented as a combination of physical and data link layers.

2.1.1 Ethernet Interface

Ethernet is a set of standards and specifications defining network communication technology which is mostly used in local, metropolitan and wide area networks. Ethernet enabled network devices re able to communicate via exchange of data packets. Segmentation of complex data into smaller individual packets makes it possible to transmit any type of data. Ethernet operates on the network-access layer of TCP/IP model, which corresponds to data-link and physical layers of ISO/OSI.

Ethernet devices can identify each other with a help of unique 48-bit hardware addresses, also called media access control addresses. Each Ethernet device is identified by a single MAC address, which plays a key role in data transfer. In order to transmit packets Ethernet establishes a link connection by including source and destination media access control addresses into packet header. Ethernet header contains EtherType field which indicates protocol used for data transfer such as IPv6, IPv4 or other. Ethernet data frames allow devices to use multiple different protocols on the same network [27]. Ethernet technology is a de-facto standard for modern day Internet network.

New Network Simulator doesn't have centralized control mechanism over simulated devices, network interfaces or protocols. In order to accomplish

authentic simulation of all components New Network Simulator treats every object as independent and capable of running in parallel. This design choice allows to implement hardware and software components according to their corresponding formal specifications instead of imitating them. All simulated components communicate by generating and responding to events. The strategy has made it possible to implement efficient and fully-parallel simulation software.

Physical Network Interface has been implemented as Unreal Engine 4 actor class. UE4 actor class is a type of construct which supports movement, animations, artificial intelligence possession and many other capabilities. Majority of Unreal Engine 4 classes have visual representation, hoverer not all of them can be repositioned at the run time. Real-time modification of visual look, size, scale, rotation angles as well as object position are supported only be non-static UE4 classes.

Actor class is a well balanced structure which allows developers to dynamically modify it at runtime. There are several other Unreal Engine 4 classes which have these properties, however they consume larger amount of machine resources. Actors provide good balance between functionality and performance. Due to this factor, actors have been chosen as the base class for implementation of all simulated components in New Network Simulator.

General physical interface class has been implemented upon Unreal Engine4 actor class. Base class of physical network interface has been extended by including new features and specific properties in order to implement different types of network interfaces, including Ethernet. Network Interface is one of the most complex implemented components of New Network Simulator. It serves as a foundation for interconnecting all simulated devices, which makes it a crucial part of New Network Simulator.

Each Ethernet interface includes a reference to device which accommodates it. This allows simulated device and physical interface to directly communicate together with a high efficiency. This reference is established when Ethernet object is instantiated. The device owner of the corresponding interface is provided to it as a parameter and is never modified afterwards.

Physical Network Interface includes data structure containing identification information for it. This data structure is unique to each simulated network interfaces and is used to distinguish devices. It contains the following values:

- IPv4 Address.
- IPv4 Network Mask.

- Hardware/MAC Address.
- Autonomous System Number.
- Unique Interface Name.
- Type of Ethernet.

User is able to set interface addresses, leaving remaining values to be managed by other components.

Ethernet interface class contains a reference to an object of the same class, which is referred to as paired interface. This referenced is not assigned at device creation and is not a mandatory parameter for interface object initialization. The main purpose of paired interface reference is to establish a link between any two simulated network interfaces.

New Network Simulator makes use of paired interfaces cross-references in order to establish communication link between them and allow bi-directional generation of software events. The reference to corresponding paired interface is stored directly in interface class as a part of interface identification data structure. Cross-references are assigned when user attempts to connect any two physical interfaces. This process requires user to first create new network interface or select already exiting one and click on "connect to" button located next to it. This functionality is available as a part of device configuration menu and can be accessed by right-clicking with mouse or touch-pad on any simulated network device.

New Network Simulator is using a global data structure in order to store references of network interfaces which user intends to connect together. When user triggers "connect to" button, the selected network interface will be stored by reference in the global space. If user repeats the same operation for another physical network interface then component responsible for interconnection will be triggered. If interface references stored as a result of user actions correspond to the same object or belong to the same device then operation will be aborted. If this validation succeeds, then interface connector object will assign cross-references of interfaces to "paired interface" field of interface identification structure. This allows both references to access each-other with the help fo stored interface reference. It enables them to generate and exchange events with each other and therefore established bi-directional communication channel.

2.1.2 Network Devices

New Network Simulator was designed and developed with flexibility and reusability in mind. In order to implement different types of network devices a base class, called generic device, was developed. This class is one of the fundamental classes of New Network Simulator. It incorporates basic functionality which makes it possible to select, create, delete, save, configure, and move devices.

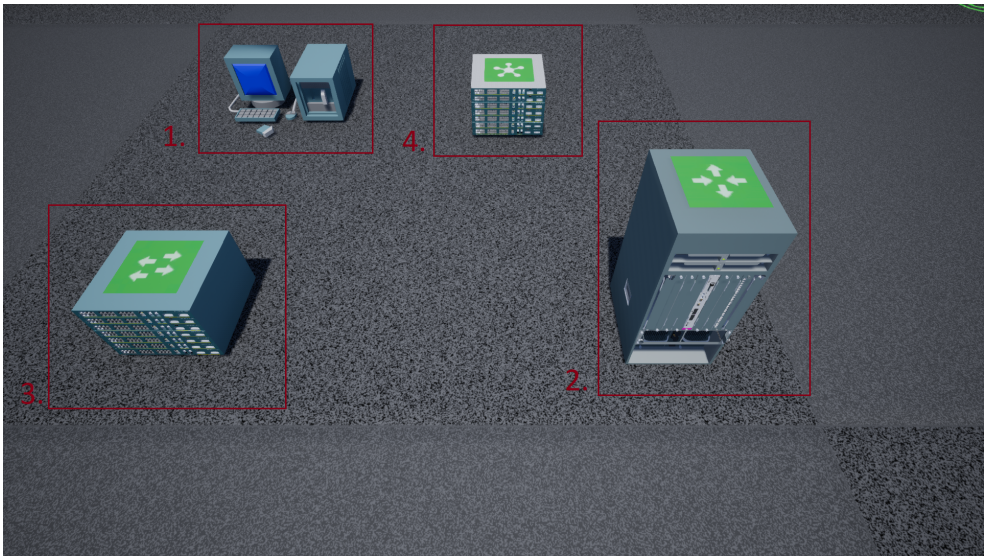


Figure 2.1: Different Types of Simulated Devices

1. Workstation/PC [2.1.2.1]
2. Router [2.1.2.3]
3. Switch [2.1.2.4]
4. Repeater [2.1.2.2]

2.1.2.1 Workstation

Workstation has originally been defined as a computational machine for operating high resource consuming software. In this Thesis the term workstation refers to any computational device capable of network communication and able to support any type of networking technology. In the scope of this research, workstation is considered to include functionality of a network router and is assumed to be capable of operating in non-standard ways. As an example of workstation we can think of a powerful Unix-based personal computer which

can be extended by custom hardware network components and configured on software level to operate as various standard network devices.

Workstation is one of the devices created by extending generic device class. Arbitrary number of workstations can be created by a user and placed at required positions. It is possible to create any number of network interfaces attached to workstation and configure each of them individually. Workstation allows users to enable and configure any combination of various network protocols. Each protocol can be inspected in order to understand its inner-workings and current state. Workstation support messenger application which can be used to communicate between all interconnected workstations and routers. It is possible to configure workstation so that it operates as a network router similar to how modern machines can be used as routers with the help of software tools and operating system.

2.1.2.2 Repeater

Ethernet repeater is a networking device for interconnection of multiple devices into one network. Repeater is often referred to as network hub due to the way it operates. Ethernet hubs are using standard Ethernet network interfaces called RJ45. Standard network hub has multiple physical network interfaces which are capable to receive and send data packets. When a packet arrives at any interface, it is then duplicated and send out from every port except the one it was originated from. Network hub belongs to the physical layer of TCP/IP model, which corresponds to layer 1 of ISO/OSI. This device is typically capable to detect network collisions are inform all ports connected to it.

Ethernet hub is able to recognize preamble of packet, which indicates the start of data. It is capable to distinguish between active and passive state of shared medium. In addition Ethernet hub is able to detect network congestion and generate jam signals. Standard repeaters do not have buffers for storing packets in transit. It requires the data to be send and received simultaneously to avoid packet loss. Due to these factor network hub is able to operate only in half-duplex, that is communication can only take place in one direction at a time. Network hubs are highly susceptible to packet collision due to the amount of traffic forwarded from output ports as well as lack of memory to store packets in transit.

Repeater is one of the supported devices in New Network Simulator which has been implemented by extending generic device class. Network repeater operates in a similar way to switch and has minimal similarities with workstation or router. Hub allows user to create any number of network interfaces but it provides limited configuration capabilities. Each port attached to repeater

has a unique auto assigned hardware address which can't be modified. Network hub is able to copy packets arriving at any port and forward them from every other existing port. Repeaters do not modify incoming packets in any way and can be used to interconnect arbitrary number of simulated devices.

2.1.2.3 Router

Router is a networking device responsible for forwarding data packets towards their destination network. Network routers are the key devices which interconnect different networks together and form Internet. There can be an arbitrary number of routers between source and destination devices. Each router on the path forwards packets toward their destination specified in packet header [2]. A typical router has at least two physical interfaces connected to different networks via shared medium. Router analyzes each packet arriving at any of its ports by parsing the packet header. Routing protocols are crucial to enable communication between routers. Each routing protocol discovers network routes to the required destination and is typically capable of prioritizing routes, based on metrics specific to that protocol. Network routers are able to use multiple routing protocols at the same time. Each standard routing protocol is assigned administrative distance value, which indicates the overall quality of routes provided by the protocol. Router is able to prioritize network routes discovered by protocols which better administrative distance. In order to make a routing decision router retrieves source and destination addresses from received packet and performs look-up of its routing table, while taking into account routing policies. As a result, router will then drop the packet or forward it according to retrieved route.

New Network Simulator allows the user to create and configure simulated network routers. Each device can have an arbitrary number of simulated network interfaces which can be freely configured. It is possible to enable any combination of network protocols on any router per simulated interface. Network router allows the user to set any property of simulated network interface without limitation. It is possible to configure individual network protocols, monitor their behavior and access their internal values.

2.1.2.4 Switch

Network switch is a device used to interconnect several devices. We can think of switch as evolutionary iteration of network hub. Physical interfaces of switch can operate as input and output ports. Switch is able to receive packets on its input ports and forward them from one or more output ports. Network

switch is using hardware addresses specified in packet header to make forwarding decisions. One of the main differences between network hub and switch is the ability to recognize hardware addresses. Network switch is able to build hardware address table, which maps addresses specified in incoming packets to destination port. This process is typically accomplished by storing hardware addresses of received packet and corresponding input port. This allows switch to reduce traffic congestion and avoid forwarding of packets to invalid destinations. When mapping between hardware address and network interface is not present in the table, switch will then perform flooding operation. Network switch operates on physical layer of TCP/IP model, which corresponds to layer2 of ISO/OSI. Some of the modern network switches, referred to as layer3 switches are also capable of manging traffic between multiple local area networks. Network switches support concept of virtual local area networks. The main principle behind VLAN is an isolation of traffic. We can think of VLANs as a set of interconnected individual physical switches. Each port of network switch can be assigned a VLAN number, therefore limiting traffic propagation between ports.

New Network Simulator is capable of simulating network switches. Their implementation has been based on generic device class in the same way as workstation, router and repeater. User is able to attach arbitrary number of network interfaces to any switch, but their configuration is limited in comparison to router or workstation. New Network Simulator makes it possible to define complex switching mechanism. Each interface of network switch can be assigned an arbitrary number of individual VLANs. It is also possible to specify sets of VLAN ranges per network interface.

2.1.3 ARP Protocol

Internet protocol datagrams are encapsulated into Ethernet data frames, which in turn requires us to find corresponding hardware addresses. Address Resolution Protocol is used to discover hardware address of target network port, corresponding to the specific Internet Protocol address. ARP is using a table for mapping between addresses which is constructed by broadcasting ARP request packet containing required Internet Protocol address. When ARP-enabled network interface receives request packet, it compares its own address with the one specified in request packet and responds directly to initiator if the they match. At the same time, device which have received ARP request will also update its mapping between hardware and IP addresses, by storing sender's addresses to ARP table. It then becomes possible for Ethernet enabled network interfaces to communicate by performing ARP table look-up.

Address Resolution Protocol has been implemented as an independent entity running in parallel with other components. In order to enable ARP user can select it in protocol panel of network interface configuration menu. ARP instance is created and initialized the moment user confirms selected settings and is discarded analogously. ARP is able to operate autonomously by reacting to events generated by other components. Each ARP instance has a reference to interface which it belongs to as well as to device to which interface is attached. This allows smooth direct communication between components with the help of application events. ARP object is able to perform required operations in order to provide Internet protocol address for a given hardware address. It is able to independently construct and sent request messages and process received ones.

2.2 Internet Layer

This section provides specifications of Internet layer protocols. It describes implementation details of Internet layer components and explains design choices.

Internet layer of TCP/IP model is responsible for transmission of datagrams beyond the borders of local area network. It serves as a virtual interface concealing underling network topology. Internet layer is used to establish internetworking, serving as a backbone of world wide web. Internet layer provides specification of network addressing in addition to routing mechanisms.

The main protocol operating on this layer of TCP/IP model is Internet Protocol. It defines network addressing and provides functionality for transfer of datagrams between IP-enabled routers on its path to the destination. Internet layer is responsible for transmitting packets beyond a single network, this process is called routing. In order to route packets from source to destination routers forward them to the most suitable neighbor according to the underling routing protocol. In addition, Internet protocol makes it possible to identify individual devices on the network with the help of unique addresses.

Internet layer is ignorant of the payload which is been transfered and is capable of transferring packets for different protocols operating on the upper layer. Each such protocol is identified by a unique number, allowing IP to distinguish between them. Internet protocol is also used transport Internet Group Management Protocol which is used to established multicast group membership as well as Internet Control Message Protocol, designed for sending error messaging and diagnostic information. The equivalent of TCP/IP Internet layer in ISO/OSI model is network layer.

The original system used for addressing on Internet layer was ARPANET. It was then replaced by the Internet, powered by Internet Protocol version4. The limited length of IPv4 addresses was a major obstacle which led to the introduction of its successor IPv6. Internet Protocol version6 uses 128 bit addressing system which provides approximately 4.3 billions assignable addresses.

2.2.1 IPv4 Protocol

Internet Protocol version 4 is a fundamental Internet layer protocol used fo communication in the scope of TCP/IP model. IPv4 specification was first published as Internet Engineering Task Force Request for Comments 791 publication in Septmeber 1981. It has been originally deployed as a part of Advanced Research Projects Agency Network in 1983 and is still the most commonly used Internet layer protocol today. IPv4 is defined for packet-switched networks and operates on best-effort delivery model. Internet Protocol doesn't ensure correct order of transfered packets, has no mechanism to detect duplicate packets and doesn't guarantee successful packet delivery. It is assumed that upper layer protocols of TCP/IP model will take care of these functions.

IPv4 uses 32-bit integer number to represent addresses and doesn't explicitly require to use any particular notation. The most common way to describe Internet Protocol version 4 address is to use four octets, each represented as decimal number and separated by a dot. Classless Inter-Domain Routing is a standard used for presenting IPv4 subnet masks. In CIDR notation IPv4 address is separated from prefix, representing network mask - the number of bits in the address set to 1.

Internet Protocol version 4 has been implemented as autonomous object operating in parallel with other components. Each simulated network interface has exclusive instance of IPv4 which is created and discarded by selecting corresponding option in protocol panel of interface configuration menu. IPv4 is used to encapsulate data into packets as well as to extract it from received packets. Internet Protocol version 4 has a reference to shared properties of device to which network interface is attached. This allows bi-directional communication with components on higher layers. When any data frame is received on device network interface, it is analyzed, decapsulated and transfered to higher level protocols based on the received data. New Network Simulator will attempt to pass the data to IPv4 protocol which is responsible for further processing. This provides low coupling of software components and makes simulator flexible.

2.3 Transport Layer

This section specifies fundamental transport layer protocols - UDP and TCP in addition to EIGRP. It describes design choices and implementation details associated with them.

Transport layer is responsible for enabling communication between hosts on local area network and beyond its borders. This layer establishes communication channels used by applications to transfer data. This layer provides process to process connection independent from the format of data being transfered and exchange mechanism. Transport layer provides the following functionality:

- Flow control
- Port numbers to distinguish between applications
- Error management
- Segmentation of data
- Congestion management

Specification of application port is one of the key features of TCP/IP transport layer. The main purpose of ports is to distinguish communication channels used by different applications on the same host. Several ports have been reserved for standard protocols and services, which in turn provides hosts ability to use some services without the need to now exact ports used by them.

Two most commonly used protocols for data transmission are UDP and TCP. UDP is a fundamental transport layer protocol. It is capable of transferring datagrams at a higher speed than TCP, but is unreliable. UDP is typically used by applications where the amount of data transfered is relatively large and packet loss is allowed. Some of the common applications for UDP include transmission of Voice over IP, video and audio data. TCP protocol is very similar in its design to UDP, but it also provides reliable data transfer. TCP is able to detect packet loss and retransmit the data when needed, while UDP does not. TCP has the following responsibilities on transport layer:

- Ensure that packets follow correct order
- Ensure integrity of data
- De-duplicate received packets
- Monitor and retransmit lost packets

- Manage network traffic congestion

UDP and TCP are not the only existing transport layer protocols. It is also possible to use Internet protocol over High Level Data Link Control Protocol or Stream Control Transmission Protocol, which are both reliable and capable of transferring data over the local network or beyond. The equivalent of TCP/IP transport layer in ISO/OSI model is a fourth layer sharing the same name.

UDP and TCP protocols have been implemented as a separate software classes which can be plugged to simulated network device as required. They are initialized when user enables corresponding protocol and is discarded when user disables it. UDP and TCP are independent from device itself or network interfaces connected to it. This design concept makes it possible to run a large number of operations in parallel and makes operation processes of New Network Simulator authentic. UDP and TCP are capable to encapsulate data an pass it to lower layer protocols as well as to deconstruct received packets in order to extract the data.

2.3.1 EIGRP Protocol

Enhanced Interior Gateway Routing Protocol is a dynamic protocol capable of exchanging routing information. EIGRP shares all network routes and form neighbor adjacency when two devices running EIGRP are connected for the first time. After initial route exchange EIGRP will only perform partial updates, therefore reducing the traffic and improving device performance. EIGRP is sending Hello packets in order to identify neighbors on the network. After two devices running EIGRP exchange Hello packets they are considered to be neighbors. EIGRP devices then proceed by exchanging their entire routing tables. It is able to recognize changes on the network and update routing information. EIGRP informs its neighbors of any detected network changes by sending partial updates, which in turn enables fast convergence. This behavior is typical to link-state routing protocols and for this reason EIGRP is usually called a hybrid protocol.

EIGRP is using two tables to make routing decisions:

- Neighbor Table. EIGRP stores information about established neighboring relations on the network in its neighbor table. Enhanced Interior Gateway Routing Protocol define neighbor as a directly connected device. This relation is not transitive.

- Topology Table. EIGRP uses topology table to keep routes learned from neighboring devices. The most important fields of topology table records are route metrics and state. The state of route indicates whether or not

best path for a specific destination has already been processed. Route metrics are used to identify feasible successors, which are then inserted into routing table of a device.

Enhanced Interior Gateway Routing Protocol is one of the more complex components of New Network Simulator when it comes to implementation. EIGRP operates as an independent object which communicates with device and network interface it is assigned to. In order to enable this bi-directional communication EIGRP keeps a reference to shared properties of simulated device as well as a reference to network interface it is assigned to. EIGRP object can be created/discarded when user enables/disables related option in protocol panel of interface configuration menu. Each EIGRP instance keeps neighbor table in the object itself while sharing topology table among all EIGRP instances running on the same device. EIGRP object also contains timers used to manage routing process. All packets received at interface referenced by EIGRP instance are processed at lower layers and delegated to EIGRP if it passes the verification and contains EIGRP related data. EIGRP uses multicast on address 224.0.0.10. Enhanced Interior Gateway Routing Protocol is able to retrieve decapsulated data and process it based on operation code. EIGRP is capable of constructing packets following specification and delegating them to lower layers. Protocols on lower layers are able to encapsulate EIGRP data into corresponding packets all the way down to physical layer, which is responsible for transferring data frames over shared medium. EIGRP

2.4 Application Layer

This section describes Application layer components of TCP/IP model. It covers the processes taking place on Application layer as well as network protocols. It then provides implementation details of individual components and related features.

Application layer of TCP/IP model is used to transfer data between different applications over the network. This is accomplished with the help of various application layer protocols, communicating on top of lower TCP/IP layers. Application layer include some of the routing protocol as well as File Transfer Protocol, Hypertext Transfer Protocol, Dynamic Host Configuration Protocol, Simple Mail Transfer Protocol and others [8]. Data generated on application layer is encapsulated and sent over transport layer protocols such as Stream Control Transmission Protocol, Transmission Control Protocol, High Level Data Link Control protocol or User Datagram Protocol.

The format and presentation of exchanged data is not explicitly specified the the scope of TCP/IP model. In contrast to ISO/OSI model, TCP/IP

doesn't have any intermediates between application and transport layers. Instead of it, TCP/IP model assumes that data formation and presentation is managed by software. Application layer protocols expect that lower layer protocols will establish reliable connection over the network for their use. The inner-workings of lower layers are not part of specification are only aware of basic information, such as port numbers and Internet Protocol addresses.

Application layer protocols are in very close relation with communication software running on sender and receiver machines. Several fundamental protocols are using port numbers reserved for them by Internet Assigned Numbers Authority organization. This in turn allows applications to communicate without the need to know all port numbers used by machine on the other end. Another approach is to use established ranges of usable port numbers which can be assigned at random for a limited duration, or for end-to-end applications agree on a specific port number.

Application layer protocols consider transport layer as been oblivious of the data type being transfered or it's meaning. Lower layer protocols are expected to encapsulate the data, split it into several pieces, if necessary, and then send it over physical medium. TCP/IP distinguish between support and user application layer protocols [9]. User protocols are responsible for exchange of practical data, while support protocols are used to make that transfer possible. According to this grouping, File Transfer Protocol is considered to be a user protocol, while Dynamic Host Configuration Protocol belongs to support protocols.

In the scope of ISO/OSI model we can consider application layer of TCP/IP as being equivalent to combination of Session, Presentation and Application layers.

2.4.1 RIP Protocol

Routing Information Protocol is a distance-vector protocol which uses hop count as route metric. It is one of the first distance-vector protocols and was originally defined in RFC 1058 publication. RIP has been updated several times which resulted in RIPv2 protocol, supporting Classless Inter-Domain Routing and RIPng (next generation) designed for IPv6. Routing Information Protocol assumes 15 to be maximum number of allowed hops, while routes exceeding that are considered to be unreachable.

RIP uses two different types of messages to exchange routing information - request and response. Routing Information Protocol sends broadcast request during the initialization process. When any device running RIP receives broadcast request it responds directly to sender by providing the entire

routing table. After receiving the response message containing routing table of neighbor RIP will add the route if it doesn't have any alternatives to that destination; it will ignore received route if local routing table already contains a route to the same destination with a higher metric; or update the timers of already existing route.

Routing Information Protocol version 2 has been implemented in New Network Simulator as an independent object operating in parallel. It can enabled in combination with any other network protocols in interface configuration menu. User is able to setup all timer values used by RIP and monitor inner processes taking place in protocol. RIP object is created and imitated when user enables it. Routing Information Protocol can also be disabled at any time, which results in destruction of RIP instance assigned to specific network interface. RIP is using object references in order to enable communication with simulated device and network interface to which it is assigned. RIP protocol is able to process received messages and generate outgoing ones. It is oblivious of lower layer protocols and is only working with RIP related packets and processes. When received data frame is recognized to contain RIP data as it passes though TCP/IP layers, it is eventually delegated to RIP instance assigned to network interface. RIP is solely responsible for processing the packet delegated to it, while device itself, ports assigned to it and other protocol instances continue their work.

2.4.2 Messenger Application

New Network Simulator allows user to send messages between workstations and routers. Each message can be tracked from source to destination as it passes through each TCP/IP layer. User is able to assign port numbers to each instance of messenger in order to simulate different scenarios. Each text message send over the application is encoded with UTF-8 and is divided in multiple packet if necessary. After been processed at each layer and delegated to the lower one, original text message is eventually encapsulated as payload of one or more data frames. The data is represented as a sequence of bits and is transfered to next hop as such. When data frame arrives at destination it is processed at physical layer and then delegated to higher layer protocols. Sent data is incrementally decapsulated and merged back to textual form. User is able to see all send and received messages in any instance of application running on different devices. New Network Simulator allows user to see both textual and binary representation of data. This makes it possible to simulate and monitor data transfer over one or more networks.

2.5 Graphical User Interface

This section describes tools used for development of graphical user interface components and provides their implementation details. It covers Unreal Engine 4 widgets, their capabilities and practical use-cases.

Graphical User Interface of New Network Simulator has been implemented in Unreal Engine 4. The main tool used for implementation is Unreal Motion Graphics UI Designer or UMG in for short. This component is a part of standard Unreal Engine 4 modules which have recently replaced original depreciated GUI design tools. UMG is a relatively simple tool for visual graphical use rinterface design and authoring which can be used to create various UI components. The common applications of this tools include and not limited to:

- Head-Up Display or HUD for short. HUD is a general name for user interface component, displayed to end user as two-dimensional screen overlay. Originally head-up display was used exclusively for displaying basic game information, such as status bar, health bars, navigation and others. This concept has been adopted as a part of Unreal Engine 4, but its definition has been greatly extended. In Unreal Engine, we can consider any piece of graphical user interface projected on the screen as head-up display. The engine allows is capable of using a single HUD component at a time. This doesn't create any limitations for development, because head-up display itself is composed of several independent user-interface elements.
- Menus. Unreal Motion Graphics UI Designer enables developers to create complex menus with unique look and functionality.
- Generic user-interface elements. UMG can be used to create various graphical components which are capable of user interaction. Both 2D and 3D user interface components can be design and serve various purposes. In general UMG is focused on creation of objects we can interact with. UMG's capabilities are not limited to implementation of standard UI components in the same way as majority of existing UI design tools are.

Unreal Motion Graphics UI Designer at its core is implemented as a set of pre-made graphical user interface components. This individual components are independent from each other and can be combined with the help of UMG to create more complex, unique user interface components. The basic UI components available to developers in UMG are being constantly extended with each new Unreal Engine 4 iteration. The engine version used for development of New Network simulator is 4.16. Unreal Motion Graphics UI Designer

packaged as a part of this version provides developers with the following basic user-interface components:

- Borders for UI components
- Interactive and highly configurable buttons
- Check boxes - a specific subtype of generic buttons
- Image components
- Progress bars, sliders
- Text labels and editable text-input fields
- Generic drop-down menus
- Spin-box components - a variation of input field which only allows numeric input and lets user change the value in a spinning-wheel manner
- Various panels, grids and placeholders for arranging and managing other user interface elements. There are different types of panels which can be combined together to create table-like arrangement of its child components. The child user-interface component is defined as any UI component, which is inserted in the slot of another graphical user interface element. The component with a slot, hosting the child in analogously called it's parent. The slot itself is a placeholder which is present inside some, but not all UI elements. In UMG there is no difference between slots, as they are all considered to be universally equal. If ui-element has slot, then any other component can be inserted into it, including another element of the same class as its parent. This essentially allows very complex nesting of user interface components and enables developers to create very complex structures. The only limitation when it comes to slots is the their total number that specific user interface component can support. In most cases graphical user interface components have either a single slot or unlimited number of them. When designing custom UI elements we can specify the number of slots it can support, which allows us to design various components.

Graphical user interface elements are referred to as widgets in the context of Unreal Motion Graphics UI Designer. One of the most important aspects of widgets is that they are essentially standard C++ classes with extra functionality on top of it. This in turn implies that we can embed very complex functionality into UI components. The majority of available software design tools for user interface are only allowing developers to create an outer shell.

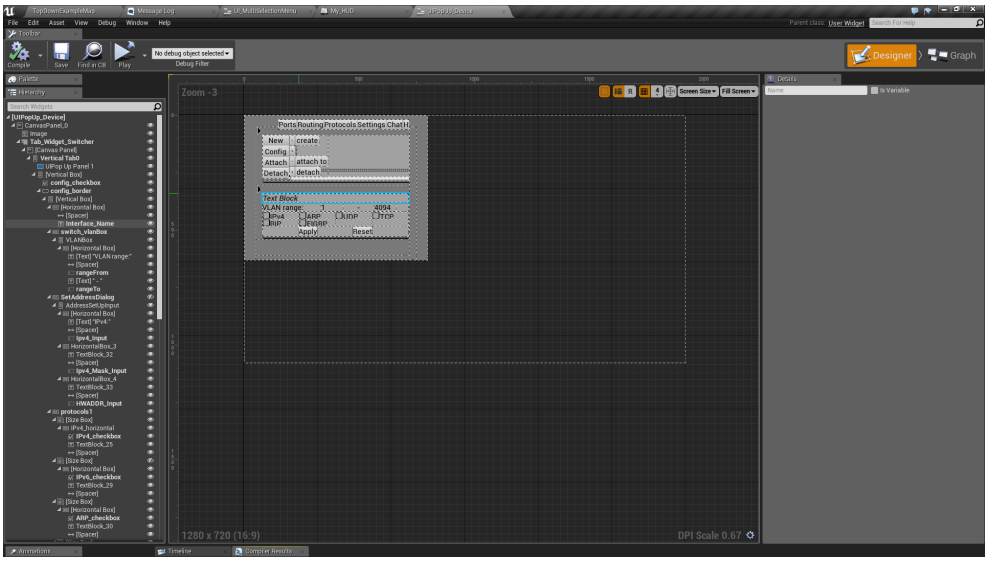


Figure 2.2: Example of Developed Widget Frontend

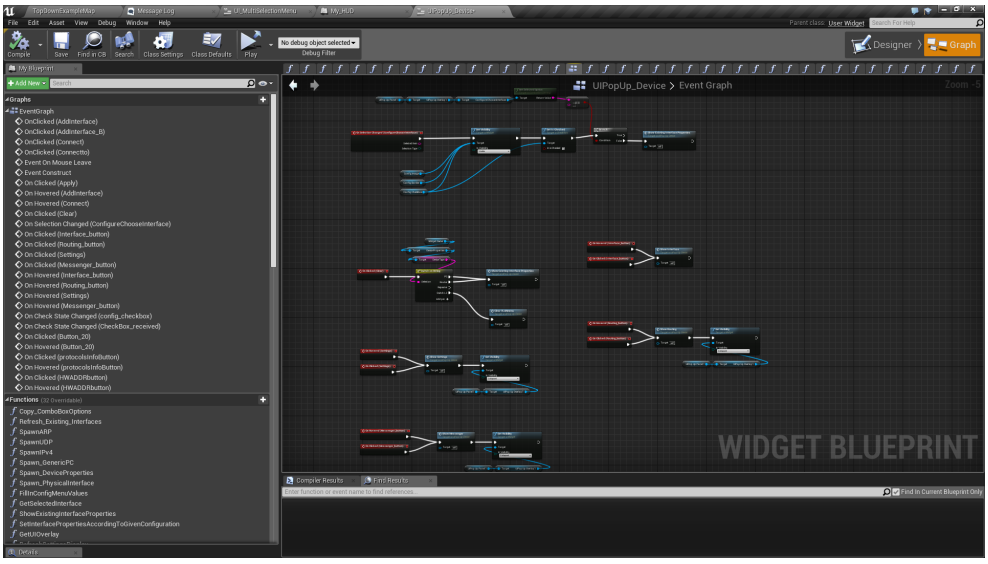


Figure 2.3: Example of Developed Widget Backend

Graphical components developed in this way are only serving for collecting user input and require implementation of comprehensive logic to verify that input and pass it to the another component for processing.

In Unreal Engine 4, UMG tool allows creation of very complex and innovative user interface components. Unreal Engine 4 includes a base class for UI elements, called widget. Unreal Motion Graphics UI Designer enables implementation of such widgets from both graphical and logical perspective. UMG includes an advanced user interface editor, where each widget is represented as having backend and fronted end at the same time. The frontend of UMG widget is essentially a graphical representation of that UI component. It is displayed to the end user, capable of accepting user-input and provides complex user interactions.

UMG allows developers to design the visual look of widget and configure its various properties. Unreal Engine 4 includes various flexible generic use interface components which are used by developers as building blocks. It includes visual representation of application window, which corresponds to configured screen resolution and display ratio. This in turn allows developers to place widget components at desired positions relative to display size. This is useful for developers, as it removes the need to manually adjust positions of elements and provides greater flexibility.

Each generic user interface component has various properties specific to its type as well as those which are universal to all widgets. It is possible to configure the visual look and feedback of widgets through these properties. In addition developers can freely change these widget parameters dynamically at the run time. This allows us to create complex user interactions which cannot be accomplished with static user interface elements.

Unreal Motion Graphics UI Designer allows developers to program the logic of UI component in addition to its visual representation. The backend part of widget can be programmed as if it was a regular class. This results in a component, capable of any functions that are available to standard Unreal Engine 4 components, but in addition also having a visual representation. This removes the need for developers to develop front-end and backend of UI elements separately, and instead encapsulates it together in the scope of Unreal Motion Graphics UI Designer. This design choice for UMG makes it possible for applications designers and developers to collaborate on the development of UI widgets with ease. UMG makes it possible to make UI components react to various events, such as:

- CPU tick events
- Widget construction and destruction

- Mouse hovering over UI elements
- Clicking on the components
- Keyboard key press
- Events specific to widget class. Check-box is capable of firing evens in response to being enabled/disabled by a user. Dropdown widget can respond to change of currently selected item. Input values are capable of detecting changes. There are several other events that specific widgets are capable of triggering

Unreal Motion Graphics UI Designer can enable user interface elements to fire various component-specific events. The events can then be captured and processed. UMG allows developers to define callback functions as a part of widget's backend implementation. Each callback function has to have the same name as fired event and can either have several or no input values passed to it during event generation. For example it is possible to configure a specific widget element to fire an event when it is hovered by mouse. We can then define a callback function which will immediately capture generated event and execute specific code, e.g. we can hide that user interface component from the user. We can combine various UI elements are events to implement very complex graphical user interface, capable of advanced user interactions.

UMG widget is at it's core a standard Unreal Engine 4 class. It can be used together with classes lacking visual representation in order to implement complex logic. Unreal Engine 4 allows developers to freely work with widgets, embedding them into any other classes and interconnecting these elements to enable direct communication between them. Widgets can be implemented to provide comprehensive visual menus to end user, capture their interactions, process them independently inside the widget itself and then pass them to lower lever classes. This specific approach has been use for development of New Network Simulator. UMG made it possible to decouple underlying application logic from the graphical user interface, while at the same time embedding complex logic for visual effects and user interactions into widgets.

Chapter 3

Features and Capabilities

This chapter covers GUI capabilities of New Simulator. It includes Main Menu designed to mainly to provide Save/Load functionality; Graphical menus and user interactions for adding, removing and moving simulated devices; GUI elements which allow auto-generation of whole networks; Configuration menus for manipulating device properties and protocols; as well as tools for monitoring network activities.

3.1 Main Menu

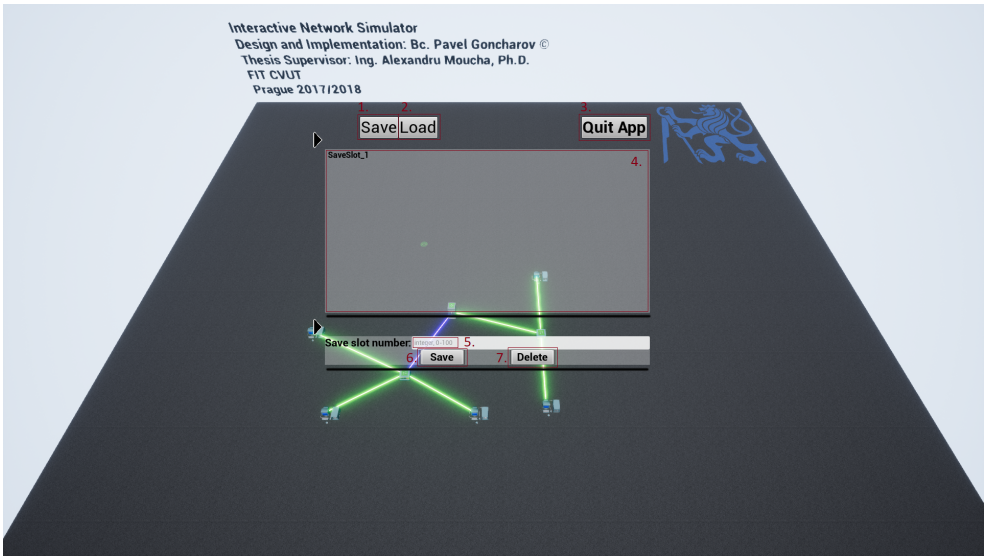


Figure 3.1: Main Menu of New Network Simulator

The main menu of New Network Simulator allows the user to save created network topology to a file as well as to load previously saved topologies. In addition main menu can be used to properly terminate the application. Main Menu can be accessed by pressing "Esc" or "Tab" keyboard keys.

1. Save Tab Button. This button allows user to open tab containing list of existing save files. It binds "Save Topology" action to button 6.

2. Load Tab Button. This button allows user to open tab containing list of existing save files. It binds "Load Topology" action to button 6.

3. Quit Button. This button will close Net Network Simulator as soon as it was triggered, therefore terminating the process. Please ensure that you network topology has been saved before triggering this button. You would now be pro

4. Save Files Panel. This panel contains a list of all existing network topology save files. The panel itself cannot be interacted with and provides read-only information. The save files listed in this panel are located in the New Network Simulator root folder under the following path: "NetSim\Saved\SaveGames". Each save file follows the following naming pattern: "[0-100].sav". The user can freely copy save files to a different directory for backup purposes. Save files can also be renamed if required, however only files following the valid naming pattern would be read by New Network Simulator.

5. Input Field for Reading Save Slot Number. This input field accept an integer in range [0-100]. When loading/saving network topology, this number will be match against existing save files.

6. Save/Load Button. This button operates in two different modes - Save and Load. In order to change to mode of the button user is required to trigger buttons 1. and 2. respectively. Upon triggering this button New Network Simulator will attempt to perform the corresponding operation. It will first read the value provided in input field 5. and verify it. If button 5. is set to "Load Topology" mode, New Network Simulator will save current network topology to a save file named according to slot number specified in input field 4. If save file with that name already exists, New Network Simulator will implicitly override that file with a new topology. Save file override confirmation prompt will not be displayed. If entered value is in correct format and button 5. is set to "Load Topology" mode, New Network Simulator will attempt to find existing save file with the same name as the provided slot number and load corresponding network topology from that file.

7. Delete Button. This button is displayed to use and configured to perform the same action regardless of mode set to button 5. New Network

Simulator will attempt to find and existing network topology save file and if it exists, delete it without providing confirmation prompt to the user.

3.2 Adding/Removing/Moving Devices

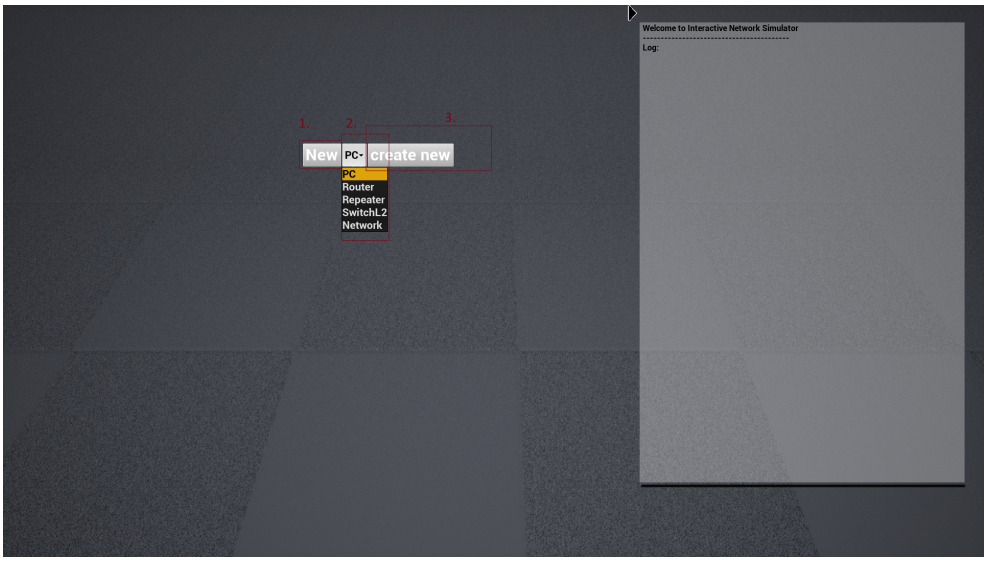


Figure 3.2: Creating New Simulated Device

New Network Simulator allows the user to create any number of simulated devices and interconnect in order to create comprehensive simulated networks. Graphical user interface of New Network Simulator has been designed to be as user friendly as possible, limiting the number of cumbersome operations for constructing network topology and configuring simulated devices. The mouse cursor of New Network Simulator is highlighted to help the user determine it's exact position. This is an important feature due the way new devices are constructed. In order to create new simulated device the user is required to right click at any free space on the field. This will in turn show device creation menu to the user, originating from the point which user clicked on. Device creation menu allows the user to create individual simulated devices of different types as well as entire auto-configured networks.

1. New Button. After following the instructions to open device creation menu outlined previously, the user will be able to see corresponding menu. User will first of all see a "New" floating button on the screen. In order to proceed, the user has either hover over the "New" button or

right click on it. This in turn will show a dropdown menu containing a set of available device types.

2. Dropdown Menu Containing Available Device Types. This menu allows the user to select one option, which will determine the type of device that will be crated after successful completion of the action. The list of available simulated devices contain options for creation of a single device as well as whole network. In order to create a single device, user is required to select one of the following options: "PC", "Router", "Repeater", "SwitchL2". These options correspond to: workstation, network router, repeater also referred to as network hub and layer two switch device respectively. It is also possible to create an entire auto-configured network according to user specifications by selected option "Network", which is covered in greater details here: [3.3]. After desired option has been selected from the dropdown menu, "Create New Button" will appear to user.

3. Create New Button. This button enables user to confirm the selection of desired option and complete the operation. If a device type has been selected, such as "PC", "Router", "Repeater", "SwitchL2" then after clicking on "create new" button a new device of that type will be immediately created. The position at which new device will be created is determined by the position of mouse cursor at which user initially clicked on with right mouse button. In other words, a position of the cursor which was selected before device creation menu was opened, and at the same time the point of origin from which that menu was originally spawned. The point of origin device creation menu, as well as all other menus of New Network Simulator is considered to be the leftmost corner of corresponding menu. In addition to creation of single simulated device of required type, user is also able to create an entire auto-configured network with required properties by selection option "Network" from dropdown menu 2. If that specific option was selected, then instead of immediately creating a device, New Network Simulator will close device creation menu and show additional menu for creation of auto-configured networks. More details on it are provided here: [3.3].

New Network Simulator is able to create individual simulated network devices as well as entire auto-configured networks according to user defined properties. This functionality can be accessed by opening device creation menu [3.2] and selecting "Network" option from dropdown menu [2].

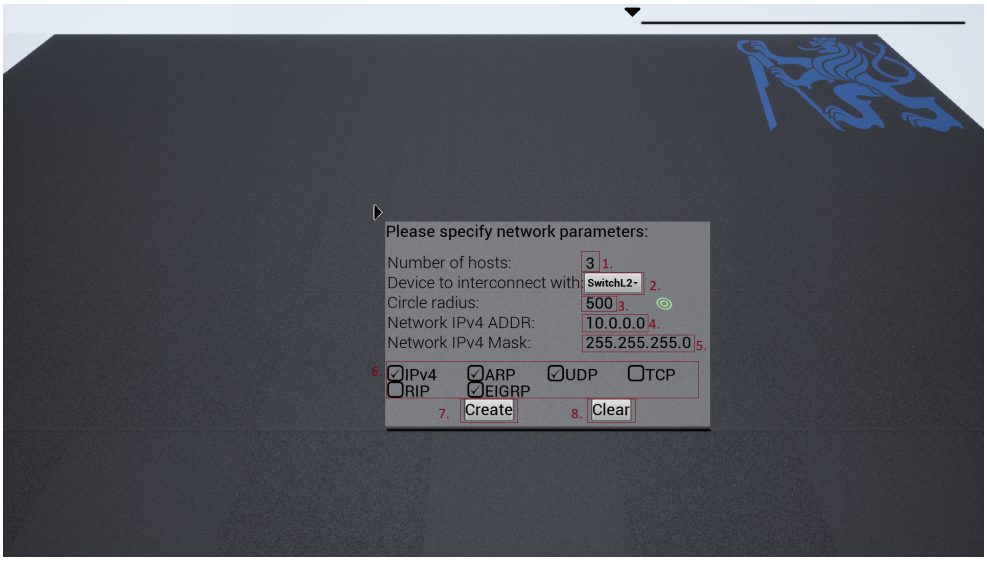


Figure 3.3: Creating Auto-Configurable Simulated Network

1. Number of Hosts. This input field accepts an integer, which specifies a total number of simulated hosts also referred to as workstations or PCs that will be created in order to form a new simulated network. This number doesn't include a device used to interconnect hosts together. There is no upper limit on the total number of hosts that user can specify, however user should be advised to keep that number reasonable, according to the capabilities of machine running New Network Simulator. In addition it is up to the end user to ensure that specified network parameters, provided in input fields 4. and 5. form sufficiently large range of assignable IPv4 addresses. If the range of available host IPv4 addresses is smaller than the value provided in field 1., then New Network Simulator will create as many hosts as possible and ignore the remaining ones.

2. Type of Device Used for Interconnection of Hosts. This dropdown menu allows user to select a type of device that will be created in the scope of newly generated auto-configured network. This device will be placed in the middle of created network and connected to every hosts. This in turn will create a single network with directly connected hosts.

3. Circle Radius. New Network Simulator creates simulated auto-configured networks by placing host devices in a circular manner with interconnecting device in the middle. The value provided in "circle radius" input field is primarily used to compute the distance between all generated devices and corresponding rotation angles. This allows users to create well organized and visually pleasing network topologies, while avoiding

unnecessary cable clutter and collision of devices. Users are encourage to experiment with radius and number of hosts to find desired ratio. New Network Simulator will not allow devices to collide with each other and spawn beyond boundaries, therefore user is not required to provide exact values. In case a circle on which newly generated hosts will be placed is not sufficient to accommodate all of devices, New Network Simulator will only create as many of them as possible and ignore the rest.

4. Network IPv4 Address. This input field accepts a string in the format 'ocet.ocet.ocet.ocet', e.g. '10.0.0.0'. This value is treated as IPv4 address of simulated network. User is responsible to ensure that provided value follows correct format.

5. Network IPv4 Mask. This input field accepts a string in the format 'ocet.ocet.ocet.ocet', e.g. '255.255.0.0'. This value is treated as IPv4 network mask of simulated network. User is responsible to ensure that provided value follows correct format.

6. Check-boxes for selecting enabled network protocols for all created devices, which can support them. New Network Simulator allows the user to select a set of network protocols which will be enabled on every device of generated auto-configured network, provided that individual device types support it. Each protocol can be enabled/disabled by triggering corresponding check-box. Setting individual check-boxes to checked/unchecked state will result in corresponding protocols to be enabled/disabled. New Network Simulator allows the user to enable or disable any protocol on any device at anytime. It is therefore not necessaries to specify exact set of protocols during generation of auto-configured network. It is possible to adjust set of protocols for individual devices as required later on.

7. Create Button. This button can be used to finalize configuration of new simulated auto-configured network and immediately create it according to specified parameters. The placement of created network depends on the original mouse cursor position, determined by a point of the field, clicked on with a right mouse button. In other words, it corresponds to the position of a mouse cursor, right before menu for creation of network devices was opened [3.2].

8. Clear Button. This button allows the user to reset all configurable input field for creation of auto-configured simulated network. This will set all input fields to their corresponding default values.

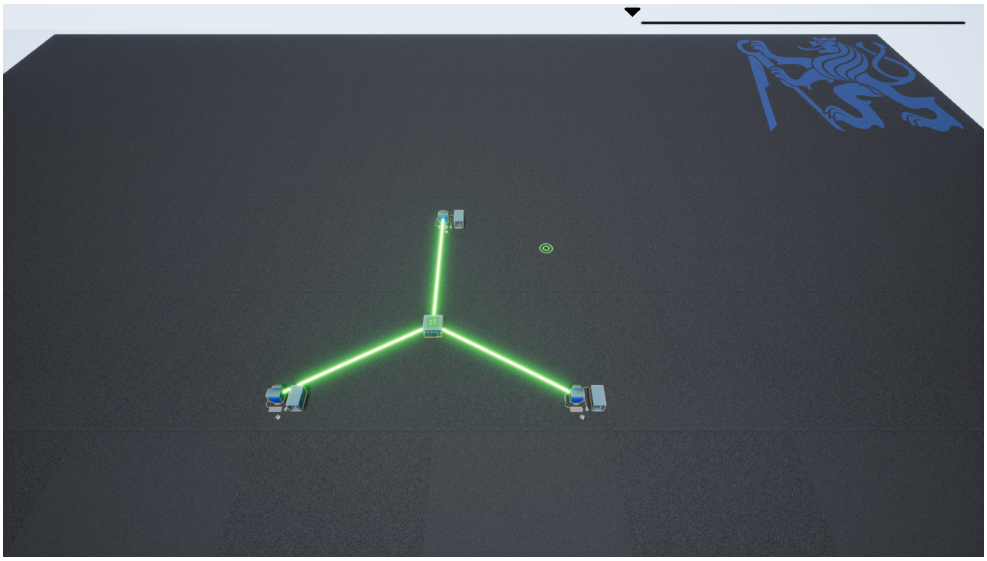


Figure 3.4: Example of Created Network

New simulated network with auto-configured properties will be created upon successful completion of input form [3.3]. The center of a network will be placed at the mouse cursor position, while all created hosts will be located around it. In situation when there is not enough space to place every device, New Network Simulator will attempt to populate available space with a smaller number of simulated devices than was originally specified. Network illustrated in figure 3.4 has been generated from configuration provided in [3.3]. New Network Simulator is capable to efficiently allocated simulated devices in the required area. In addition it is able to display physical mediums, used to interconnect devices in a clear and coherent way.

3.3 Configuring Devices

After we have successfully created simulated networking device of specific type with the help of device creation menu described in 3.2, we can further configure it according to out needs. In order to configure device of any type it is sufficient to click on right mouse button while hovering over that device. This will open configuration menu to the user, which corresponds to type of selected device. In New Network Simulator different device types has common and unique configuration options.

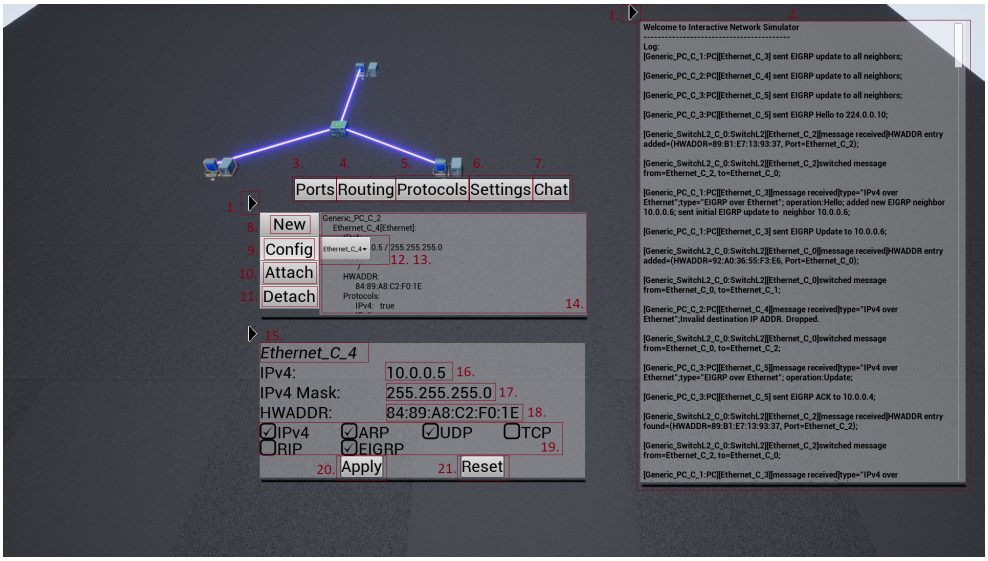


Figure 3.5: Creating and Configuring Simulated Network Interfaces

In the scope of New Network Simulator host device is also referred to as workstation or PC. We can see configuration menu of a host device on figure [3.5]. It consists of the following graphical user interface components:

1. Hide/Show Panel Button. Majority of user interface panels implemented in NewSim can be minimized/maximized to save space and focus on specific UI elements. This can be achieved by left-clicking triangle buttons next to their respected panels.

2. Network Events Log. Event Log is one of the key components of New Network Simulator. This panel allows user to monitor all processes and events that takes place in simulated network. This processes include, but not limited to sending/receiving data frames, packets encapsulation/decapsulation, routing decisions, construction processes of various internal tables and data structures.

3. Ports Tab. User is able to open network interface creation and configuration menu by left-clicking or hovering over this button.

4. Routing Tab. User is able to open menu containing device routing table representing best known routes to available destinations. It can be accessed by left-clicking or hovering over this button.

5. Protocols Tab. User is able to open protocol inspection and configuration UI panel by left-clicking or hovering over this button. More details regarding protocols tab are provided here:[3.3].

6. Settings Tab. It is possible to open general device settings menu by lieft-clicking or hovering over this button. Detailed description of settings tab can be found here:[3.6].

7. Chat Tab. User is able to send/receive messages with required properties between simulated devices using Chat Tab. It can be accessed by left-clicking or hovering over this button. Detailed description of messenger application can be found here:[2.4.2].

8. New Button. Hovering over this button or lift-clicking on it allows user to open Interface Type Selection Dropdown.

9. Config Button. User can left-click or hover over this button to open list of created network interfaces. Selecting any of them will open port configuration menu which allows to modify the settings of that interface.

10. Attach Button. In order to connect any two network interfaces of any devices, user first have to left-click or hover over this button. As a result, dropdown menu containing list of existing interfaces will be displayed. User has to select and item from dropdown menu and then click on "attach to" button. New Network Simulator will save a reference to selected network interface to shared memory. After that user has to perform the same operation with a second interface. As a result these too network interfaces will be interconnected via shared medium.

11. Detach Button. User can click on this button in order to open dorpdown menu containing list of interfaces which are currently connected to any other interface. User is able to select an item from the list and click on "detach" button in order to disconnect that network interface.

12. Interface Type Selection Dropdown. This dropdown menu contains a list of available network interface types. User is required to select an item from this list in order to proceed with network port creation.

13. Create Button. Left-clicking on this button will result in creation of a new simulated interface of selected type.

14. Interface Configuration Information Panel. This panel contains a list of all created network interfaces. In addition it displays configuration summary and status of each interface.

15. Interface Name. This field displays unique name of currently selected interface to which configuration will be applied.

16. Interface IPv4 Address. This field displays current value of corresponding interface property and can be modified.

17. Interface IPv4 Network Mask. This field displays current value of corresponding interface property and can be modified.

18. Interface Hardware Address (MAC). This field displays current value of corresponding interface property and can be modified.

19. Active Protocols Configuration Panel. This panel allows user to enable or disable simulated network protocols for selected interface. In order to enable/disable any protocol user is required to left-click on corresponding check-box.

20. Apply Button. User can apply all currently selected values of corresponding network interface by left-clicking on this button.

21. Reset Button. This button allows user to clear all modified configuration fields and set them to their original values.

Settings Tab contains general configuration properties of workstation, router and switch device types.

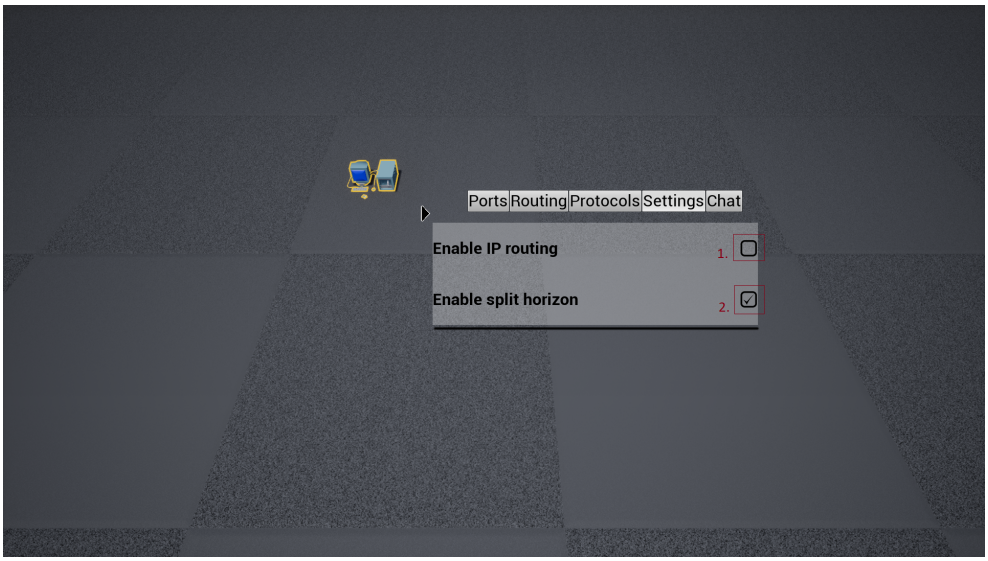


Figure 3.6: Configuring Workstation

1. Enable IP Routing Checkbox. User is able to enable/disable routing functionality of simulated device by changing state of corresponding checkbox. When this option is enabled, all received packets, addressed to another recipient will be forwarded according to routing table instead of been dropped

2. Enable Split Horizon Checkbox. This checkbox can be used to enable/disable split horizon for all protocols running on device. Split horizon prevents routing network protocols from advertising routes over interface at which they have been learned.

New Network Simulator includes comprehensive configuration options of implemented network protocols. Protocol properties can be accessed from device configuration menu [3.5] by clicking on Protocols Tab.

EIGRP configuration menu allows user to setup protocol properties, such as timers and K-values. In addition it contains EIGRP Neighbor and Topology tables.

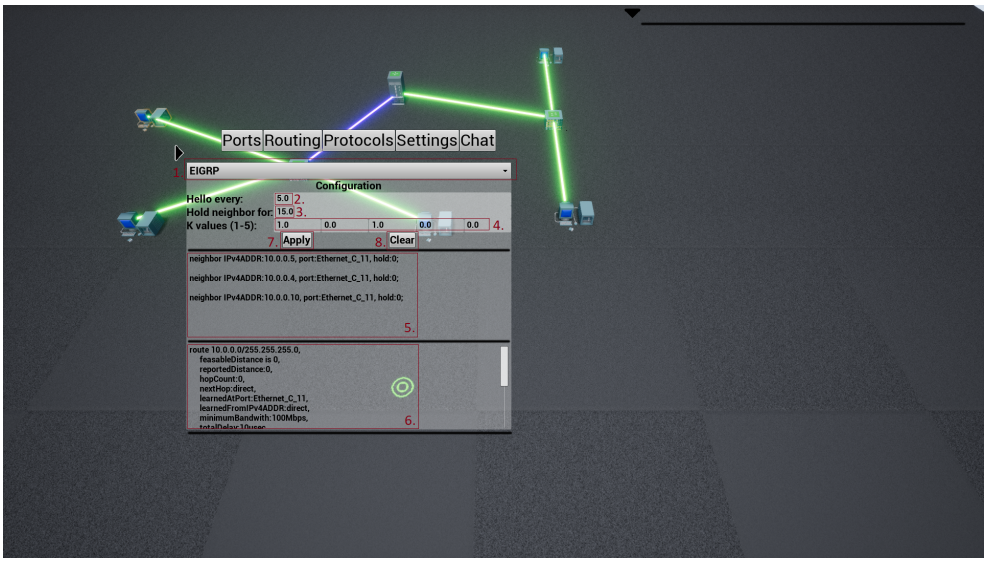


Figure 3.7: Configuring EIGRP

1. Protocol Selector Dropdown. User can switch between protocol configuration menus by selecting appropriate item from dropdown. If particular protocol is not enabled then menu will still be displayed, indicating that no settings/activity has been recorded yet.

2. Hello Packet Timer Configuration. EIGRP will send hello packet from corresponding interface after this time period expires (in seconds).

3. Route Hold Timer Configuration. EIGRP will retain neighbor adjacency relationship for this period of time at most (in seconds), if no hello packets have been received.

4. K-values Settings. These five fields can be used to set corresponding K-values (K1-K5 from left to right). K-values are used by EIGRP to compute metrics of route.

5. EIGRP Neighbor Panel. This panel shows a list of all currently established EIGRP neighbor relations.

6. EIGRP Topology Table. This table contains EIGRP successors and feasible successors for all known destinations. Device routing table panel contains bets known routes for all available destinations, including best EIGRP routes.

7. Apply Button. This button is used to set all configurable properties to values currently displayed in corresponding fields.

8. Clear Button. This button can be used to reset all configuration fields back to their current values.

RIP settings tab makes it possible for a user to set required protocol properties such as timers. This menu also allows user to see RIP routing table.

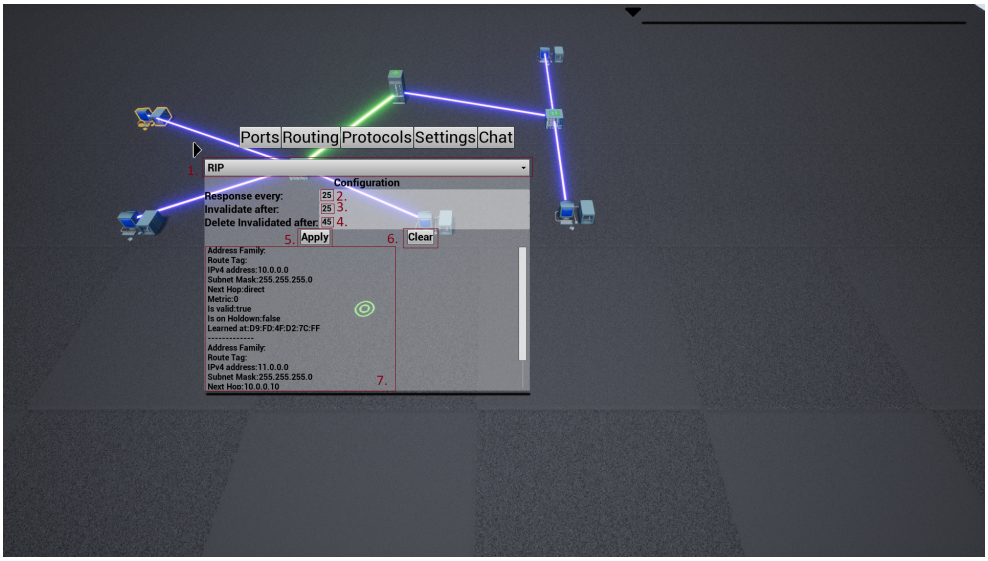


Figure 3.8: Configuring RIP

1. Protocol Selector Dropdown. User can switch between protocol configuration menus by selecting appropriate item from dropdown. If particular protocol is not enabled then menu will still be displayed, indicating that no settings/activity has been recorded yet.

2. RIP Response Packet Timer Configuration. RIP will send unrequested response packet from corresponding interface after this time period expires (in seconds).

3. RIP Route Invalidation Timer Configuration. RIP will consider existing topology table entry as invalid if no response packet is received during this time interval(in seconds).

4. RIP Deletion of Invalidated Route Timer Configuration. RIP will delete entry from routing table which has been marked as invalid for this time period (in seconds).

5. Apply Button. User is able to set protocol properties corresponding to currently displayed configuration values.

6. Clear Button. This button can be used to reset all configuration fields back to their current values.

7. RIP Routes Panel. This table displays up-to date state of RIP routing table. It can be refreshed by switching between currently active menu tabs.

New Network Simulator provides user with different network interface configuration menus, based on type of underling simulated device. Figure [3.9] contains layer two network switch configuration menu.

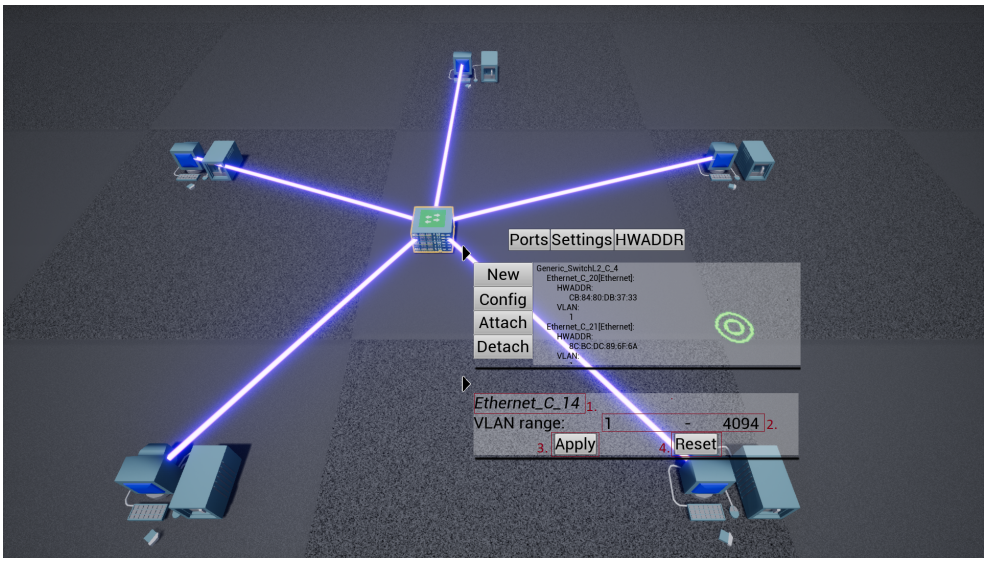


Figure 3.9: Configuration of Switch

1. Network Interface Name. This fields displays unique name of network port currently being configured. It cannot be modified by user.

2. VLAN Range. Virtual local area network range can be specified by providing two corresponding values. It is limited from 1 to 4094. Several

ranges can be specified for the same network interface. Some of the examples are: "1-1" will assign VLAN number 1; "1-2" assigns VLAN numbers 1,2; etc. It is possible to specify, e.g. "5-10" and then "15-20" in order to assign 5,6,7,8,9,19,15,16,17,18,19,29 VLAN numbers. Assigning multiple ranges requires user to repeat the process corresponding number of times.

3. Apply Button. When user left-clicks on this button, currently selected VLAN range will be read and immediately applied to simulated network interface. It can be used to add and remove VLAN numbers. When specified range includes all or some of already assigned numbers then matched values will be removed, and those which are missing will be added. In other words a set union of already assigned numbers and those given by the range is computed, which results in those number been unassigned. Unique values between these two sets (complement) are added to already assigned VLAN numbers.

4. Reset Button. This button can be used to reset all configuration fields back to their current values.

Simulated layer two network switch is using table for mapping between hardware address of interface from which packets originates, and interface of switch to which that packet should be forwarded. User is able to access this table by right-clicking on desired switch and then navigating to HWADDR Tab [3.10].

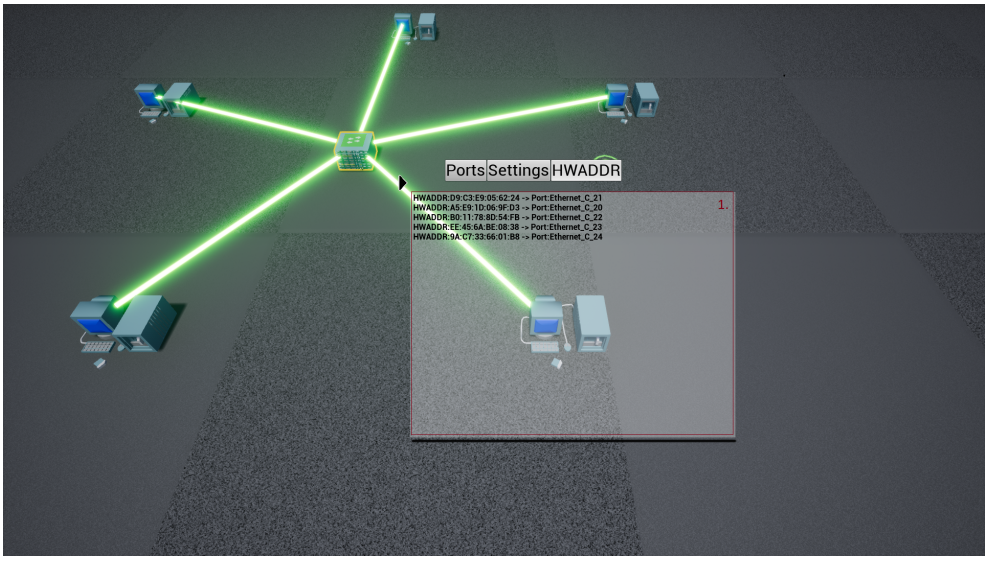


Figure 3.10: Content-Addressable Memory Table of a Switch

1. Hardware Address Mapping Panel. This panel displays the contents of CAM table which is used to map destination hardware addresses to outbound network interfaces of the switch. This table is constructed by learning hardware addresses of packets received at corresponding interfaces. Switch uses this table to forward packet to required destination if this table contains corresponding entry. If mapping between hardware address and switch port is not yet present in the table then switch performs flooding operation.

New Network Simulator allows user to access routing table of workstations and routers containing best known routing paths. This table is constructed based on directly connected physical interfaces and routes inserted by running routing protocols. In addition, each device is able to automatically select routes with best administrative distance and metrics in order to select best route to every known destination. Routing table can be accessed by right-clicking on workstation or router and then navigation to Routing Tab [3.11].

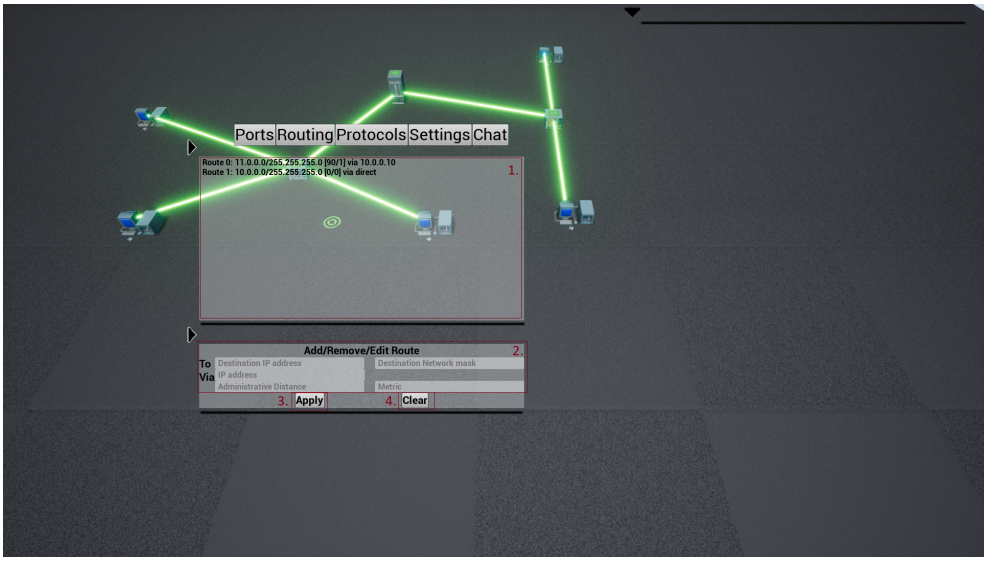


Figure 3.11: Routing Table Containing Best Paths

1. Best Routes Panel. This panel contains best known routes to all available destinations. Every routing protocol running on device is able to insert its best route for corresponding destination. This device-wide routing table contains only routes provided by running protocols with best administrative distance. It is used to make routing decisions when sending data.

2. Static Route Creation Panel. This panel allows user to insert static route into routing table. It is also possible to remove any route from the table

by specifying it in coresponding fields and clicking on "Apply" button. If provided values match already existing route in the table, then it will be removed.

3. Apply Button. User is able to add/remove routing table entries by left-clicking on this button.

4. Clear Button. This button can be used to reset all configuration fields back to their current values.

3.4 Network Events and Topology Views

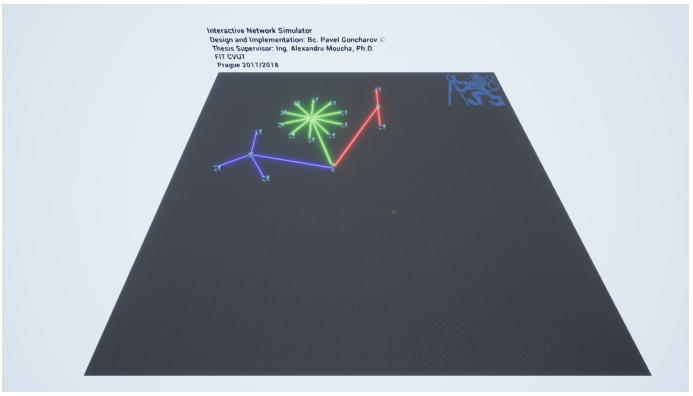
New Network Simulator provides user with a selection of several different views of network topology. This provides user with an overview of entire topology while making it possible to focus on individual parts.

- Mouse wheel can be used to zoom in and out of topology.

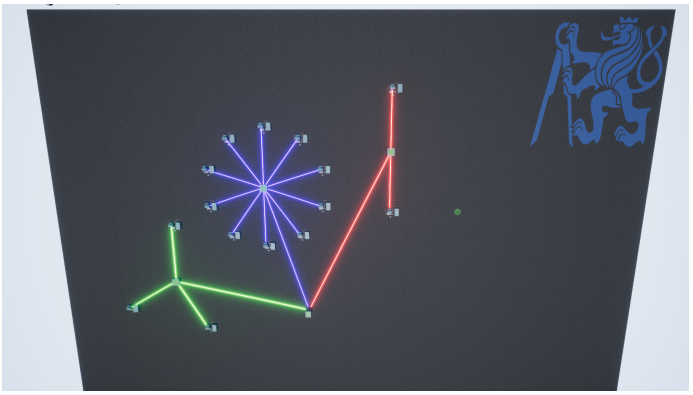
- The view angle can be changed by pressing "V" button on the keyboard. Pressing "V" repeatedly allows user to circle between different topology vies [3.12a][3.12b][3.12c].

- Shared medium connecting simulated network interfaces changes its color based on the link state. Red identifies that link has not been established yet or is currently down. Green means that there is no data in transit and link itself is operational. Blue indicates that data is currently being transfered over corresponding link.

- All types of simulated network devices have unique identification symbol on top of them which corresponds to standard conventions.



(a) First View Option



(b) Second View Option

(c) Third View Option

Figure 3.12: Different Views of Network Topology

New Network Simulator includes text messenger which can be used to communicate between all connected workstations and routers. This application layer component makes it possible to verify connection status between devices and monitor behavior of enabled network protocols. New Network Simula-

tor makes it possible to track transmission process of packets, including all aspects of protocol behavior.

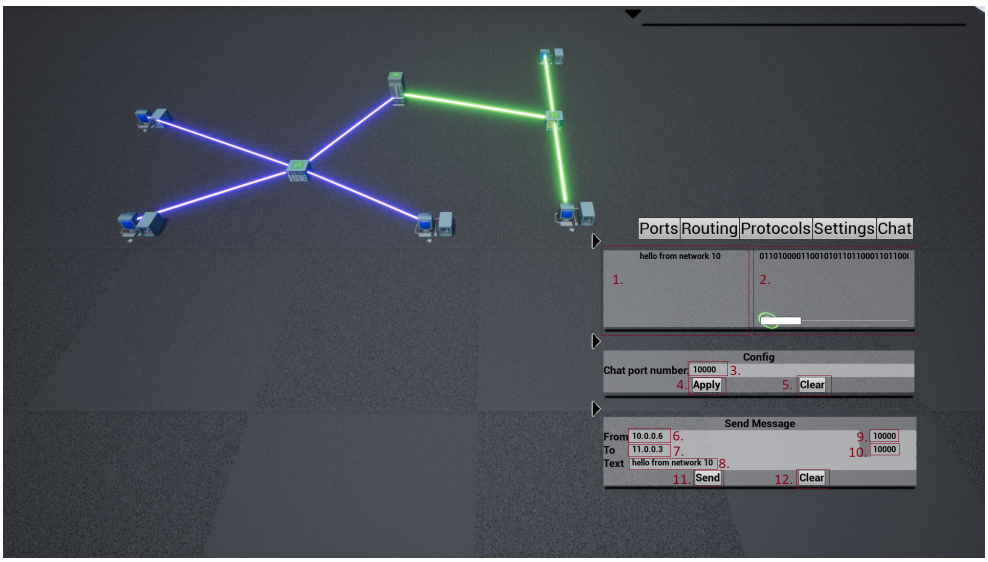


Figure 3.13: Text Messenger

1. Panel Containing Received Text. This panel displays all messages received by this device from every port. The output is displayed in human-readable text form using UTF-16 encoding.

2. Panel Containing Binary Representation of Received Text. This panel contains concatenated payload of binary payload retrieved at transport layer.

3. Application Port Number. This field indicates currently set port number for Messenger application.

4. Apply Button. User is able to set currently selected port number by left-clicking on this button.

5. Clear Button. This can be used to reset entered port number and display currently set value instead

6. Source IPv4 Address. Device will attempt to send message from one of its interfaces which has this IPv4 address,

7. Destination IPv4 Address. The destination address network interface to which message should be sent.

8. Text Message. Message payload which can have arbitrary size. Large messages will be segmented and sent as a sequence of data frames.

9. Source Application Port. This port number will be included in outbound message.

10. Destination Application Port. The required port number on device to which this message is sent.

11. Send Button. User can send the message according to specified values by left-clicking on this button.

12. Clear Button. This button can be used to reset all input fields back to their current/default values.

Conclusions

This research has successfully shown that new innovative tools for network design and planning are required in order to support the grows and advancement of emerging technologies. This work was able to demonstrate advantages of network simulation technology and identify numerous practical applications for it. A detailed analysis of existing development tools and methods have been provided. This in turn has allowed us to plan and design New Network Simulator in a coherent and efficient way. A multitude of innovative ideas for improvement upon exiting technology has been identified. These potential improvements have become the foundation for development of the brand new network simulator.

This work has analyzed simulation software in great detail, covering all of the key aspects. Network topology design has been studied in order to identify essential requirements for comprehensive and innovative simulation software. This Thesis has provided an insight into design and evaluation of network protocols and communication devices. It has demonstrated practical ways in which network simulation can help us to optimize these processes and achieve better overall results. Comprehensive study of individual TCP/IP model layers has been performed. Analysis of Link Layer has included network interfaces, Ethernet technology and the ways in which it can be simulated. Different aspects of networking devices have been studied, including their fundamental functionality, capabilities and relative performance. This work has made it possible to simulate workstations, network hubs, switches and routers. Specifications of Link layer protocols, such as Ethernet and ARP have been provided. This Thesis has analyzed IPv4 protocol and identified possible ways to simulate it. Fundamental transport layer protocols, such as TCP, UPD and EIGRP have been studied and simulated according to their specifications. This work was able to successfully analyze application layer communication software in addition to RIP routing protocol. This comprehensive work has been used as a foundation for design and planning of the

New Network Simulator.

This research has ultimately resulted in the successful implementation of a new network simulator, which has incorporated best practices of the existing simulation software and introduced new unique features. The implementation of such complex application would not have been possible without the use of advanced modern software development tools and frameworks. The Agile approach has proved to be the right choice as it facilitated gradual implementation of new features and delivery of fully functional prototypes after each iteration. The most significant risk associated with this Thesis was excessive implementation complexity, which could have potentially taken years to complete. The most crucial tool to overcome this obstacle was Unreal Engine 4, which was opened to the general public on March 2, 2015. UE4 made it possible to implement very complex graphical user interface and enabled smooth user interactions with simulated network topology. All four layers of TCP/IP model were included in the final version of the New Network Simulator allowing simulation of hardware and software components. Some of the most fundamental and widespread protocols have been implemented according to their specifications.

The design and development of New Simulator has been an astounding success despite overwhelming complexity associated with its development and time limitation. The New Simulator turns topology construction and protocol evaluation into an easy and fast task for inexperienced users, while at the same time underling framework is even more suited for design and planning of hardware and software network components and protocols from ground-up, with a possibility to evaluate new model at the same time.

It is very important to state that the project itself is in fact a framework upon which various components can be constructed. One of the main goals was to keep the internal implementation of New Simulator as close as possible to the processes taking place in real-world devices. There were two possible approaches to fundamental design, each having its own advantages and drawbacks. A major decision had to be made at the very early stage of development which determined the overall direction and definition of success for this project. On one hand, simulation of protocols themselves could have established as a goal, while simplifying the implementation of individual devices, treating them as black boxes with a centralized control mechanism. This in turn would have implied that internal logic of the simulator, such as hardware, communication principles and independence of processes would have been downsized. Such decision would have made a big negative impact on the future-proof of this work, limiting the code-base to a specific domain without possibility to extend project in the future. On the other hand, a heavy focus on decoupling, parallelization and adherence to realism would re-

sult in a comprehensive, extremely flexible framework which can be used for implementation of variety of tools.

After careful consideration the later approach have been chosen. This has proved to be very practical and beneficial, leading to the design capable of addressing problems on a large scale. The proportion of time spent on the development of simulation framework far exceeds the time used for implementing individual network protocols. This demonstrates how big of a challenge it was to implement authentic hardware and software components, but it also tells us that New Network Simulator is extremely flexible and powerful tool for implementation and evaluation of network protocols, not only in the scope of TCP/IP networks, but also entirely new models.

Bibliography

[1] Collier, K. W. *Agile Analytics: A Value-Driven Approach to Business Intelligence and Data Warehousing*. Pearson Education, 2011.

[2] Alliance, A. Agile 101. 2013, [Online; accessed 1-December-2017]. Available from: `https://www.agilealliance.org/agile101/`

[3] Alliance, A. Manifesto for Agile Software Development. 2001, [Online; accessed 2-December-2017]. Available from: `http://agilemanifesto.org/`

[4] Martin, J. *Rapid Application Development*. Macmillan, 1991.

[5] Inc., A. Examining the Agile Manifesto. 2011, [Online; accessed 2-December-2017]. Available from: `http://www.ambysoft.com/essays/agileManifesto.html`

[6] Alliance, A. Principles behind the Agile Manifesto. 2001, [Online; accessed 3-December-2017]. Available from: `http://agilemanifesto.org/principles.html`

[7] Moran, A. *Agile Risk Management*. Springer Verlag, 2014.

[8] Vasiliauskas, V. *Developing agile project task and team management practices*. Eylean, 2014.

[9] Jeffries, R.; Anderson, A.; Hendrickson, C. *Extreme Programming installed*. Addison-Weslsy, 2001.

[10] Larman, C. *Agile and Iterative Development: A Manager's Guide*. Addison-Weslsy, 2004.

[11] Steiner, B. How the Unreal Engine Became a Real Gaming Powerhouse. 2013, [Online; accessed 1-December-2017]. Available from: `http://www.popularmechanics.com/culture/gaming/a9178/how-the-unreal-engine-became-a-real-gaming-powerhouse-15625586/`

[12] Records, G. W. Most successful videogame engine. [Online; accessed 4-December-2017]. Available from: http://www.popularmechanics.com/culture/gaming/a9178/how-the-unreal-engine-became-a-real-gaming-powerhouse-15625586/

[13] Gies, A. Talking the future of Unreal with Epic's Tim Sweeney. 2015, [Online; accessed 2-December-2017]. Available from: https://www.polygon.com/2015/3/5/8152541/talking-the-future-of-unreal-with-epics-tim-sweeney

[14] Thomsen, M. History of the Unreal Engine. 2010, [Online; accessed 3-December-2017]. Available from: http://www.ign.com/articles/2010/02/23/history-of-the-unreal-engine

[15] Burnes, A. Epic Reveals Stunning Elemental Demo, Tim Sweeney On Unreal Engine 4. 2012, [Online; accessed 4-December-2017]. Available from: https://www.geforce.com/whats-new/articles/stunning-videos-show-unreal-engine-4s-next-gen-gtx-680-powered-real-time-graphics

[16] Thier, D. Epic's Tim Sweeney on How Unreal Engine 4 Will Change The Way Games Are Made, and Why You Care. 2012, [Online; accessed 2-December-2017]. Available from: https://www.forbes.com/forbes/welcome/?toURL=https://www.forbes.com/sites/davidthier/2012/06/29/epics-tim-sweeney-on-how-unreal-engine-4-will-change-the-way-games-are-made-and-why-you-care/&refURL=https://duckduckgo.com/&referrer=https://duckduckgo.com/

[17] Totilo, S. How Unreal Engine 4 Will Change The Next Games You Play. 2012, [Online; accessed 1-December-2017]. Available from: https://kotaku.com/5916859/how-unreal-engine-4-will-change-the-next-games-you-play

[18] Dyer, M. GDC: Epic Games' Unreal Engine 4 adopts subscription model. 2014, [Online; accessed 2-December-2017]. Available from: http://www.ign.com/articles/2014/03/19/gdc-epic-games-unreal-engine-4-adopts-subscription-model

[19] Connors, D. Epic Games Opens Unreal Engine Marketplace to Developers. 2014, [Online; accessed 2-December-2017]. Available from: http://www.escapistmagazine.com/news/view/137229-Epic-Games-Unreal-Engine-Marketplace-Open-UE4-Unreal-Tournament-2014

[20] Batchelor, J. Putting Unreal Engine in the classroom. 2014, [Online; accessed 1-December-2017]. Available from: https://www.develop-online.net/interview/putting-unreal-engine-in-the-classroom/0197275

[21] Rad, C. Epic Games Wants To Give 5 Million USD In Grants To Unreal Engine Devs. 2015, [Online; accessed 2-December-2017]. Available from: http://www.ign.com/articles/2015/02/19/epic-games-wants-to-give-5-million-in-grants-to-unreal-engine-4-devs

[22] Sirani, J. Unreal Engine 4 is Free for Everyone. 2015, [Online; accessed 3-December-2017]. Available from: http://www.ign.com/articles/2015/03/02/unreal-engine-4-is-free-for-everyone

[23] BBC. Unreal games engine licensed to FBI and other US agencies. 2012, [Online; accessed 5-December-2017]. Available from: http://www.bbc.co.uk/news/technology-17535906

[24] Engine, U. Connected Vehicle Research. 2010, [Online; accessed 1-December-2017]. Available from: https://www.unrealengine.com/en-US/showcase/connected-vehicle-research

[25] Williams, M. IPKeys licenses Unreal Engine 3 for military simulations. 2012, [Online; accessed 2-December-2017]. Available from: http://www.gamesindustry.biz/articles/2012-12-06-ipkeys-licenses-unreal-engine-3-for-military-simulations

[26] Barrie, A. Army, DHS join forces for virtual training tech for first responders. 2013, [Online; accessed 3-December-2017]. Available from: http://www.foxnews.com/tech/2013/11/21/army-dhs-join-forces-for-virtual-training-tech-for-first-responders.html

[27] Comer, D. E. *Internetworking with TCP/IP âĂŞ Principles, Protocols and Architecture*. Prentice Hall, 2000.

Appendix A

Acronyms

NewSim New Network Simulator

TCP/IP Internet Protocol suite

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

ARP Address Resolution Protocol

UDP User Datagram Protocol

TCP Transmission Control Protocol

RIP Routing Information Protocol

CAM Content-Addressable Memory

EIGRP Enhanced Interior Gateway Routing Protocol

BGP Border Gateway Protocol

UE4 Unreal Engine 4

UMG Unreal Motion Graphics User Interface Designer

GUI Graphical User Interface

VLAN Virtual Local Area Network

HWADDR Hardware Address

MAC Media Access Control

Appendix B

Contents of Enclosed Data Storage

- `P.Goncharov_Master_Thesis.pdf` ..... Thesis in PDF format
- `application` ..... directory with New Network Simulator
  - `Linux64` ..... contains application built for Linux/Unix
    - `NetSim.sh` ..... bash script for application startup
    - `NetSim` .....
      - `Saved` .....
        - `SaveGames` . directory containing saved network topology files
  - `Windows64` ..... contains application built for Windows7/8/10
    - `NetSim.exe` ..... application startup executable
    - `NetSim` .....
      - `Saved` .....
        - `SaveGames` . directory containing saved network topology files
    - `Engine` .....
      - `Extras` .....
        - `Redist` .....
          - `en-us` .....
            - `UE4PrereqSetup_x64.exe` ... installation package containing all required dependencies for Windows7/8/10 64-bits platform