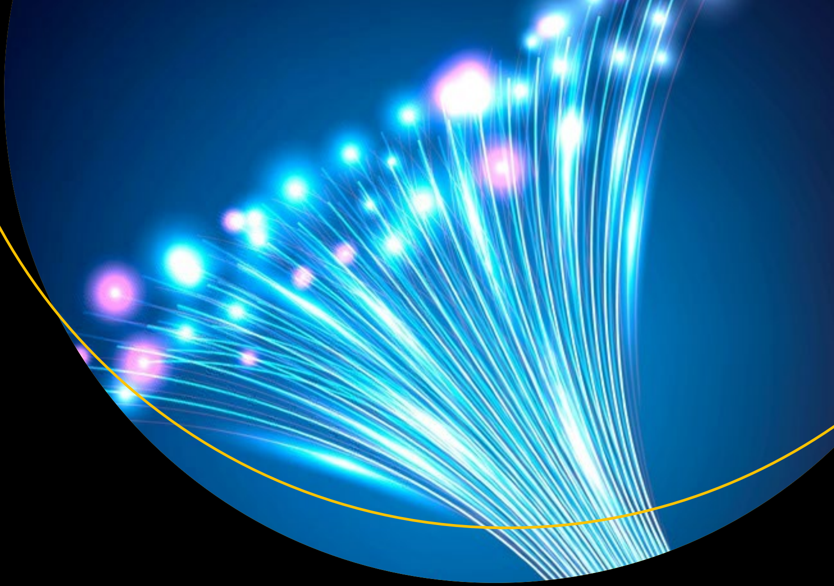

Wireless Network Simulation

A Guide using Ad Hoc Networks and the ns-3 Simulator

Henry Zárate Ceballos
Jorge Ernesto Parra Amaris
Hernan Jiménez Jiménez
Diego Alexis Romero Rincón
Oscar Agudelo Rojas
Jorge Eduardo Ortiz Triviño

Apress®

Wireless Network Simulation

A Guide using Ad Hoc Networks and the ns-3 Simulator

Henry Zárate Ceballos
Jorge Ernesto Parra Amaris
Hernan Jiménez Jiménez
Diego Alexis Romero Rincón
Oscar Agudelo Rojas
Jorge Eduardo Ortiz Triviño

Apress®

Wireless Network Simulation: A Guide using Ad Hoc Networks and the ns-3 Simulator

Henry Zárate Ceballos
Bogotá, Colombia

Hernan Jiménez Jiménez
Bogotá, Colombia

Oscar Agudelo Rojas
Mosquera, Colombia

Jorge Ernesto Parra Amaris
Montreal, QC, Canada

Diego Alexis Romero Rincón
Bogotá, Colombia

Jorge Eduardo Ortiz Triviño
Bogotá, BOGOTA, Colombia

ISBN-13 (pbk): 978-1-4842-6848-3
ISBN-13 (electronic): 978-1-4842-6849-0
https://doi.org/10.1007/978-1-4842-6849-0

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Natalie Pao
Development Editor: James Markham
Coordinating Editor: Jessica Vakili

Distributed to the book trade worldwide by Springer Science+Business Media New York,1 NY Plazar, New York, NY 10014. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/978-1-4842-6848-3. For more detailed information, please visit www.apress.com/source-code.

Printed on acid-free paper

Table of Contents

About the Authors

Henry Zárate Ceballos received his PhD in engineering computing and systems and his master's degree in telecommunications from the National University of Colombia. Henry is currently a researcher with the TLÖN Group. Henry has worked extensively with the ns-2 and ns-3 simulators and wireless distributed operating systems.

Jorge Ernesto Parra Amaris received his master's degree in telecommunication from the National University of Colombia, and a master's degree in electronics engineering from the Colombian School of Engineering Julio Garavito. Jorge's master's thesis proposed a unique algorithm that was validated through simulation using ns-3.

Hernán Jiménez Jiménez received his postgraduate master's in telecommunications from the National University of Colombia. Hernán is currently a researcher at TLÖN Group.

Diego Alexis Romero Rincón received his master's in electronics from the National University of Colombia and is currently a researcher with the TLÖN Group. Diego focused his master's thesis on the ns-3 simulator. Diego is currently a lecturer at the National University of Colombia.

Oscar Agudelo Rojas is a systems engineer and lecturer at the National University of Colombia, where he also received his master's degree in telecommunications. His research work includes networks (wired and wireless), network coding, simulation (ns-2 and ns-3), and parallel and distributed systems.

ABOUT THE AUTHORS

Jorge Eduardo Ortiz Triviño received his PhD in engineering computing systems and master's degrees in telecommunications, statistics, and philosophy from the National University of Colombia. Jorge is currently is Associate Professor in the Department of Systems and Industrial Engineering and Director of the TLÖN research group of the Universidad Nacional de Colombia, while also working as a network specialist.

About the Technical Reviewer

John Edwar Gonzalez Ortiz Electronic Engineer John Edwar Gonzalez has more than five years of experience in the telecommunications industry. He has a master's degree in telecommunications engineering from the Universidad Nacional of Colombia. He has focused his projects on the area of ad hoc networks and in his work, it is possible to see the focus on the nodes that make up the network and their interaction when a social inspired behavior is applied to the network. John Edwar has belonged to the TLÖN research group for more than 5 years and his research can be seen in various journals where the behavior of the nodes is analyzed. Topics such as decision making in nodes, resource negotiation, game theory, altruism and selfishness have been addressed in his articles.

Preface

Today connectivity is the principal need in our technologically linked society. In this information society, users from children to elders share their information, show their feelings, and publish their lives on the information networks. Distributed and highly complex systems established between machines support these networks, which interact in fractions of seconds over long distances, delivering all kind of services. Both machines and services are transforming our environment, with engineers' new ideas about computing devices, data networks, and information systems. This high demand for services is the result of the evolution of several elements: first, the growth of the Internet due to the changing nature of user preferences, the increasing number of connections, and the development and diffusion of social networks. Another factor is the emergence of mobility features that add dynamic and random behavior to linked devices, systems, and users.

Network services are support services at cities, government institutions, university campuses, and companies, to name a few. These networks provide service to the Internet and intranets, allowing shared information, services, and stablishing users communications. Access to these services is through different means such as optical fiber, copper, and air. Commonly, the interactions between users happen over several networks and mediums. The change of mediums is one of the critical processes for the throughput and quality of network services and the management of the systems supported by them across all communications channels and network components. Network components are usually diverse, and with only a few of them, it is possible to build relatively complex systems. It is difficult to predict their performance or characterize their operation when there are too many nodes, a heterogeneity of components, multiple layers of specialized functions, different services, and different mediums.

With all these factors, how do you know what the network behavior will be? There are two ways: first you can emulate it or determine the key points of the traffic behavior virtually through modeling or by reproducing the logical processes involved. The reliable option to emulate is intended to reproduce the network, routers, switches, nodes, and users; however, it is quite extensive and expensive. Another solution is the use of simulators, which are computational tools that allow the generation of a similar scenario to a real one. The use of simulators can help to explore interactions, component performance, and theoretical limits. Simulations are useful tools for empirical research because they permit us to generate data from a real network that can be high priced or difficult or impossible to control when designing a new network model that needs novel hypotheses for experimentation.

Setting up a virtual environment is useful to re-create a massive network with thousands of nodes. For instance, to evaluate mobile data traffic in IoT, Cisco [1] estimates that the monthly global mobile data traffic will be 49 exabytes by 2021, and the annual traffic will exceed half a zettabyte. The IoT environment has produced an increase in mobile devices, which will represent 20 percent of the total IP traffic. The platform business creates real Big Data scenarios and connects consumers with producers who share information, goods, and services through the Internet.

Simulation is a type of research methodology to compare some models, identify hypotheses, and understand the behavior and interactions between services, users, devices, and architectures. Since a network simulator can be event-based, each event represents an abstraction of a network and a computer system. For instance, nodes and physical networks can be represented in classes such as node and channel classes. The tools and components used, and the explanations, revolve around the ns-3 simulator.

The ns-3 simulator allows the simulation and emulation of networks. It is an open and free simulator that emulates networks using the network interface card (NIC) of the computer that tests and transports the traffic generated by the simulation script and saves the simulation data in different traces for post-simulation data analysis. In this sense, it is important to discuss many concepts related to simulators, the abstractions used for the ns-3 simulator, the application of the stack protocols (TCP, UDP, OLSR, and so on), and the computational model created to imitate the NICs, routers, and other network devices.

With simulation, it is easier to get quantitative results, identify relationships, establish system interactions, determine component performance, and reach theoretical limits. One of the best ways to improve and check the simulation results is to share their results and scripts. In a huge system like the Internet, due to scale, heterogeneity, and level of interaction, the exclusive analytical option is to simulate. It is useful when it is necessary to perform statistical models for data interpretation, with one simulation or with a set of simulations. Each simulation has stages and requires a working methodology. The main objective of this book is to show the mechanism and techniques to design and create simulation models, use the simulator and analyze the results, and find the factors that affect and describe the simulation or the model created.

The book has three parts. The first part covers simulation basics including general information about network simulation and wireless and ad hoc networks and some techniques for experiment design. The second part covers Network Simulator 3 (ns-3) and gives some examples and techniques for analyzing results. The third part covers wireless network simulators on ns-3 that conclude with examples and models to simulate wireless, wired, and mixed networks with ns-3.

Specifically, the first part has three chapters that explain network simulation, wireless networks, ad hoc networks, and experiment design. Chapter 1 explains simulation features, objectives, and the techniques and steps to do simulations.

Chapter 2 gives some insights about wireless and wired networks. Taking elements from the real world and applying them to the simulation world, we explain the evolution and principles of operation on architecrures dynamic and stochastic, such as the Internet of Things (IoT), fog computing, edge computing, and the mobile cloud. These are the new trends in Internet service delivery. In addition, the chapter explains the concept of cyberspace and of interactions on the Internet.

Chapter 3 shows some techniques for experiment design, the key issues for the script design, and the event selection over the network. After the simulation, the most important activity to be performed is the analysis of results, where events are reported, and of the network behavior, including problems and improvements that a network, a model, or a new protocol could have.

The second part of this book covers ns-3. Chapter 4 introduces the ns-3 simulator, including the main abstractions, code style, tracing, and logging. Chapter 5 shows the techniques to analyze the results post-simulations, take information from the generated traces, and determine the reliability of the simulation and the relevance of the simulation model.

Finally, in the third part, Chapters 6 and 7 include examples of mobile ad hoc networks (MANETS) with all the necessary steps for the simulations, to give you more clarity about the use of ns-3 and the process of analyzing the results. Chapter 6 show how to build an ad hoc network and analyze it with artificial agents using the ns-3gym and Open AI Gym tools. Chapter 6 introduces an example that links the ad hoc networks with power line communications (PLC). It is an approximation for the IoT environment. At the end, we present the conclusions and prospects of the network simulations and the future needs in this research field.

For the authors, this book is not just a dream come true but an effort of a team of friends, researchers, and fellow students. With this book, we want to inspire others to write, learn, and apply their knowledge to share it with others.

CHAPTER 1

Introduction to Simulation

> *The sheer volume of answers can often stifle insight...The purpose of computing is insight, not numbers.*
>
> —[2]

Framework

Computers have become one of the main resources for research. They are essential to analyze models through simulations, giving more options to verify the interactions between the components of a model, and essential to analyze large amounts of data.

Simulation is used for theoretical and empirical research since it provides the means to explore all the capacities and limits of theoretical models and because it helps to create synthetic conditions that are difficult to re-create in a real experiment. In some research specialties, this field is considered a third methodology [3]. For instance, any tangible laboratory sample can be re-created with a model in the computing world; the physical device would be the computer program or software, and the measurements would be the computer tasks [4]. A simulation is an application or a computer process that attempts to imitate a physical

H. Zárate Ceballos et al., *Wireless Network Simulation*,
https://doi.org/10.1007/978-1-4842-6849-0_1

process by producing a similar response that allows someone to make predictions about the expected behavior of a system. As a result, it can be used as an experimental setup or as a support to make operational decisions. It is also employed to study difficult and complex systems before spending resources on a real experiment.

Simulations, Models, and Their Importance in Research

Before any simulation, it is essential to have a model. It is a conceptual representation of a real system whose level of abstraction depends on the research question and previous knowledge from the system. A simulation cannot be executed by itself, since it requires a tool (programming framework) and a platform (computer, server, etc.) to execute and produce a response. The computational cost of a simulation depends on the complexity of the real system and the level of abstraction used to model it.

Even though some models can be validated using mathematical formalisms, some systems are complex, involving many variables and input parameters that make mathematical validation challenging. For these kinds of models, simulation provides a form of understanding at different levels; however, the knowledge acquired from these models is useful in a limited way, since the behavior is seen in conditions that are difficult to test or that are generally not seen in real systems.

If the theory is accurate, simulation is a great tool to study theoretical models. It also allows discovering how the responses would be in different scenarios. Simulation cannot validate a model by itself, only instantiate it. Therefore, to validate it, the same test scenario must be implemented under real-world conditions to compare its results with the simulation output to gain enough accuracy of the model and validate it.

Theoretical models represent the behavior of the system based on its knowledge and not the behavior of a real system. These models need validation before being considered empirical. An ideal way to validate them is through simulation. When simulating a theoretical model under a determined set of conditions, the result works as a hypothesis for the behavior of the real system if it is tested under the same circumstances. If the experiment data is statistically close to the simulation output, it is feasible to infer that the model is accurate. If the model does not seem satisfactory, it does not imply that there are errors in it. There could be, but there could also be errors in instantiating the model, which could serve as a guideline for telling what not to do for a future experiment. Simulation is a powerful tool. This whole process is a method to validate simulation models through experimentation. However, it is not a substitute for real experimentation, since the simulation results are only as good as the models used. Therefore, it is mandatory to validate the model and question their results and applicability if this has not been done.

The quality of the simulation results is directly associated with the quality of the model. This implies that it is necessary to validate a model before deploying it. Model validation is a process in which the experiment is evaluated if it is an accurate representation from a real system. Empirical studies are used to ensure their accuracy. However, according to the research needs, not every model needs to be validated with the same level of accuracy. In general, to validate a model, it is possible to use two methodologies: observational methods and the experimentation, exposed earlier.

The observational methods are usually aimed at answering the research question, but in the case of simulation models, they are used to ask questions to the model output data to determine its validity. Thanks to machine-learning techniques and statistical methods, it is possible to carry out observation methods. On the one hand, machine-learning techniques employ algorithms that learn distributions and correlations to produce a model from the output data. On the other hand, to ask questions and get answers from the output data, statistical methods are used if the data has a behavior that can match certain distributions.

Types of Simulation Techniques

There are two types of systems: discrete and continuous. In a discrete system, the state variables change instantly at different points in time. On the other hand, in a continuous system, the state variable change continuously over time.

In computer networks, many systems function as discrete systems (LAN, cellular infrastructure, wireless networks); in them, specific events or interactions change the state and the behavior of the entire system. In the simulation program, these events are inserted and read as states, variables, and routines sequentially; this approach is known as *next-event time advance*. All these attributes and events are enabled in the debugging and execution processes along with the input scripts. The general orientation of the processing is carried out through modeling, which is usually formulated in a general-purpose language.

Table 1-1 describes the most important types of simulations that are of particular importance to engineers [5].

Table 1-1. *Types of Simulations*

Type of Simulations	**Description**
Emulation	This is the process of designing and building a model that uses real system functionality. A study case is the prototyping process.
Monte Carlo simulation	This is a simulation process without time reference. Monte Carlo simulation techniques are used to model any probabilistic phenomenon that does not change over time as an independent variable.
Trace-driven simulation	This simulation uses as input an ordered list equivalent to real-world events. In this type of simulation, the time variable is an attribute of the event.
Continuous-event simulation	A function can model this type of simulation, and the changes occur permanently. An issue is to determinate the scale and the scope of the experiment to identify the factors and events that influence the results.
Discrete-event simulation (DES)	Discrete event simulation is a type of simulation that uses "events" to specify details of an experiment that occur over time. Discrete mathematical analysis can model the process and have a medium level of abstraction. Each event is a function or class call with a unique identifier.

A particular case of discrete event simulation could have the following components:

- *Event queue*: This contains all the events waiting to happen. The implementation of the event list and the functions to be performed on it can significantly affect the efficiency of the simulation program.

- *Simulation clock*: This is a global variable that represents the simulation time; the simulator advances in the simulation time until the next scheduled event. During event execution, the simulation time is frozen; however, in the ns-3 simulator, it is possible to work with the real-time scheduler integrated with the hardware clock to perform the progression of the simulation clock in a synchronized way with the machine or reference external clock.
- *State variables*: These variables help to describe the state of the system.
- *Event routines*: These routines handle the occurrence of events. Once an event is successfully executed, the simulator updates the state variables and the event queue.
- *Input routine*: This routine obtains the user input parameters and supplies them to the model.
- *Output generation routine*: This routine is in charge of creating the output of the events and the abstraction of the simulator. In ns-3, there are two kinds of outputs: `.pcap` and `.tr` files.
- *Main program*: This is the entry point on the ns-3 simulator where it is possible have C++ and Python's `main()` function program. The main program is used to call the classes, functions, libraries, and methods useful to execute the simulation. The simulation on ns-3 begins with the `Simulator::Run()` routine and ends with the `Simulator::Destroy()` routine.

Formal Systems Concepts

Usually, simulation demands a previous conceptualization effort. In some cases, because of the scope of work, it is a demanding task and difficult to understand. On this subject, there are available formal works, and some of them are based on demi-philosophical principles that could be useful. Therefore, we recommend becoming familiar with the following definitions, which are frequently used in this book.

- *Behavior*: This is the relationship between any input/output pair in a system at different times. It can be obtained from external measurement to know the internal set of events and states that characterize the system [6].
- *Emulation*: A partial or complete construction of a system that is functional and artificial, whose behavior mimics that of an analyzed reference system, this is the process of simulating the inner workings of a systems to produce a realistic output [7].
- *Event*: This is the source of the changes in a finite state machine.
- *Inference*: This is an activity oriented to deduce the internal structure of a system from its behavior. (This definition is close to the simulation world.)
- *Structure*: This is an internal characteristic that defines a set of system states and relations [6].

Regarding the real experimenting analogies, when the scope of a simulation process is to imitate a real physical process, it is important to consider an experimental orientation for collecting process data and for data analysis techniques that is similar to a scientific inference laboratory. Otherwise, in computer systems, simulations are sort of hybrid experiments, because just one side of the processes comes from the real world, like propagation media features, transmission lines parameters, delays, failures, and other common behaviors of hardware. The other side consists of software processes.

The creation of different kinds of models is the result of efforts to simulate and imitate real systems. Essentially, real-life systems and phenomena are continuous models, which means that the variables of the process can be set at any time. Unlike real-word systems, computational processing uses discrete models, which are models that change state at certain times and have a limited number of possible states.

In the description of discrete events of a system, there are instantaneous changes of discrete variables that allow imitating a real dynamic system. A combination of differential equation system specifications and discrete event system specification, inherent in the continuous and discrete descriptions respectively, allows the computational models to simulate real systems in an approximate way.

Simulation and Emulation

The simulation allows reaching a higher level that implies the fidelity to a real system. While emulation is a superior level in which all the components are simulated to produce a realistic response, as shown in Figure 1-1. However, emulation can be more computationally expensive and harder to model since its level of detail is superior and finer.

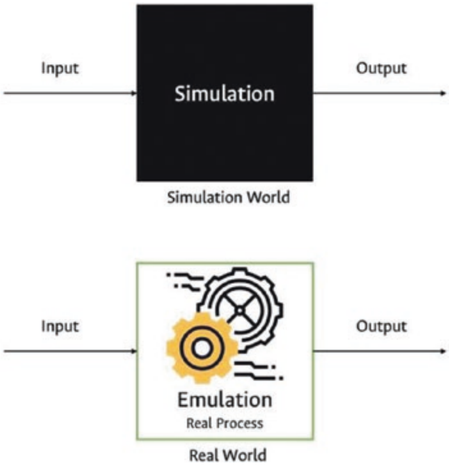

Figure 1-1. *Simulation versus emulation*

There are two domains when a simulation begins: the real world and the simulation world (Figure 1-2). It is necessary to define the elements that compose the real world to create new hypotheses and experiments. Among them are the system theories, their relationships with the data results and the preliminary hypotheses, and the system or main problem. The interactions between them are hypothesizing, abstracting, and experimenting.

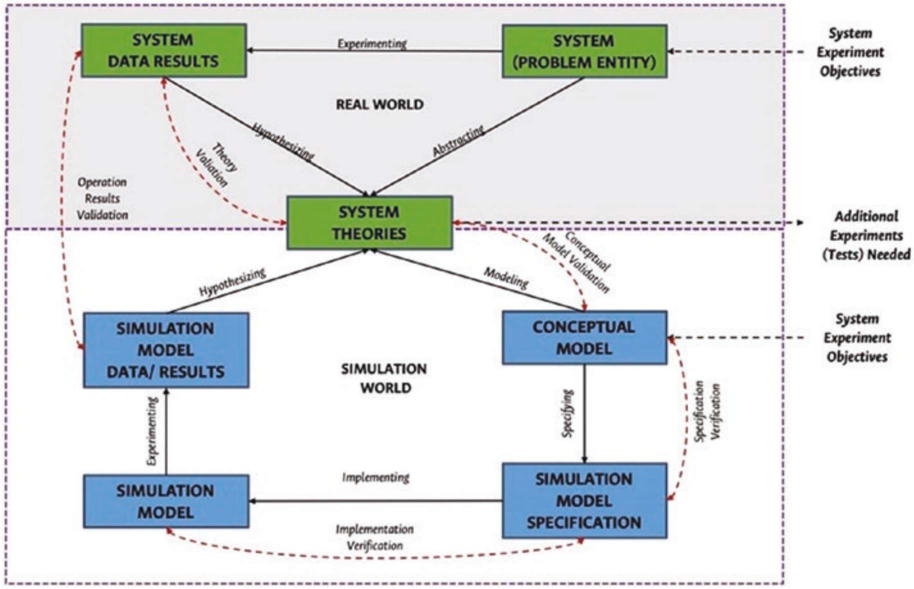

Figure 1-2. *Real world versus simulation world*

To design a simulation experiment, it is significant to define the abstract model and follow the next steps, as shown in Figure 1-3.

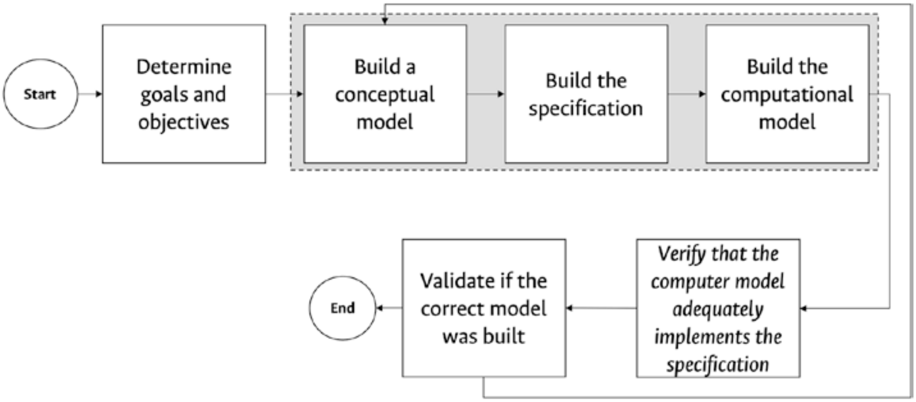

Figure 1-3. *Steps simulation*

1. Determine goals and objectives.
    - *Boolean decisions*: Should another component be added to the model?
    - *Numerical decisions*: How many servers in parallel offer optimal performance?
2. Build a conceptual model.
    - What are the important state variables?
    - How exhaustive should the model be?
3. Build the specification.
    - Collect and statistically analyze data to have “input” models that control the simulation.
    - In the absence of data, the “input” models should be built using stochastic models that are appropriate for the problem.
4. Build the computational model.
    - Select the language or the simulation tool.
5. Verify that the computer model implements the specification properly.
    - Still not the right model?
6. Validate if the correct model was built.
    - An expert compares the results of the real system with the results of the simulated system.
    - The system’s animations are useful.

Network Simulators

In communication networks, the development of new routing protocols, algorithms, and architectures is usual. The performance evaluation of these new systems through experimentation can be expensive, the resources may not be available, and valuable features such as scalability are not easy to test in that way. Consequently, simulation becomes an important tool for research since it does not require any physical hardware other than a computer to run the simulations. It provides an economical alternative to evaluate the behavior of these new systems or to test the performance of the existing ones, which under different circumstances are hard to re-create in a laboratory.

Today, it is possible to find different simulation frameworks created by network companies, universities, and academics, whose goal is to offer alternatives, covering different aspects and functionalities of networks. The selection depends on the needs and objectives of the researchers. Besides, it is recommendable to check in bibliographic databases, such as Scopus, for the number of papers that have used a certain simulator and its role on the research.

In Table 1-2, you can see some of the most commonly used network simulations for research. However, keep in mind that there are many networks simulators available, and your selection depends on the objectives of your research and your experience with different programming languages.

ns-3 Simulator General Features

ns-3 is a discrete event network simulator that uses a set of abstractions (node, application, channel, net device, and topology helpers) to simulate devices in communication networks, as well as their services, protocols, and interfaces. The interactions between them are given through multiple channels of communication like Ethernet cables, wireless channels, and power line communication channels, among others.