МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РЕСПУБЛИКИ КАЗАХСТАН

НАО «Казахский агротехнический исследовательский университет им. С.Сейфуллина»

Хамзина Б. Е., Толегенова А.С.

# МОДЕЛИРОВАНИЕ СЕТЕЙ В ПРОГРАММНОМ ЭМУЛЯТОРЕ CUPCARBON

Утверждено Академическим советом университета
в качестве учебного пособия

**Астана 2024**

**УДК 621.39:004(072)**
**ББК 32.88:32.97я81**
**Х18**

Рецензенты:
Куттыкожаева Ш.Н., д.ф-м.н., профессор КГУ им. Ш. Уалиханова.
Бактыбеков К.С., д.ф-м.н., профессор КАТИУ им. С. Сейфуллина.
Разина О.В., PhD, ассоц. профессор ЕНУ им. Л.Н. Гумилева.

Хамзина Б.Е., Толегенова А.С., Моделирование сетей в программном эмуляторе CupCarbon: учебное пособие. - Астана: КАТИУ имени С.Сейфуллина, 2024. – 100 с.

**ISBN 978-601-257-407-4**

В учебном пособии изложены основы и примеры проектирования и моделирования беспроводных сенсорных сетей на основе программного пакета CupCarbon, являющийся мощным инструментом имитационного моделирования телекоммуникационных сетей. Данное издание предназначено для обучающихся по направлениям подготовки: 6В062 - «Телекоммуникации и 7М062 - «Телекоммуникации».

Рекомендовано к печати решением Ученого совета университета от 24 ноября 2022 года, протокол №5.

Утверждение присвоения грифа УМО-ГУП РУМС высшего и послевузовского образования МНиВО РК на базе некоммерческого АО «Алматинский университет энергетики и связи имени Гумарбека Даукеева», протокол №4 от 19.09.2023 г.

**УДК 621.39:004(072)**
**ББК 32.88:32.97я81**

ISBN 978-601-257-407-4 

## Обозначения и сокращения

IoT - Internet of Things - Интернет вещей;

IoE - Internet of everything - Интернет всего;

EITO - European Information Technology Observatory - Европейская комиссия по наблюдению за информационными технологиями;

RACE - Research and Development in Advanced Communications Technologies in Europe - Исследования и разработки в области передовых коммуникационных технологий в Европе (RACE);

ТФОП - Телефонная сеть общего пользования;

ISDN - Integrated Service Digital Network - Цифровые сети с интеграцией служб;

NGN - Next Generation Network - Сети следующего/нового поколения;

TCP/IP - Transmission Control Protocol/Internet Protocol - Протокол управления передачей/протокол интернета;

SDH - Synchronous Digital Hierarchy - синхронная цифровая иерархия;

DWDM - Dense Wavelength Division Multiplexing - Плотное мультиплексирование с разделением по длине;

X.25 - Стандарт передачи данных с использованием коммутации пакетов;

ATM- Asynchronous Transfer Mode - Асинхронный способ передачи данных;

QoS - Quality of Service - Уровень сервиса или обслуживания;

LCN - Logical Channel - Number - Номер логического канала;

VCI - Virual Channel Identifier - Идентификатор виртуального канала;

DARPA - Defense Advanced Research Projects Agency - управление Министерства обороны США;

ARPANet - Advanced Research Projects Agency Network - компьютерная сеть, созданная в 1969 году в США DARPA и явившаяся прототипом сети Интернет;

OSI - Open Systems Interconnection - Модель взаимодействия открытых систем;

ISO - International Organization for Standardization - Международная Организация по Стандартизации;

LLC (Logical Link Control) - Верхний подуровень управления логической связью;

MAC - Media Access Control - Надзор за доступом к среде Hardware Address, также физический адрес - уникальный идентификатор, присваиваемый каждой единице сетевого оборудования или некоторым их интерфейсам в компьютерных сетях Ethernet.

## Оглавление

# Введение

За последние годы в современном мире технология телекоммуникационных сетей стремительно развивалась благодаря экспоненциальному прогрессу в области компьютерных чипов и волоконно-оптических технологий. За последние 40 лет внедрение сетей с коммутацией пакетов и интернета увеличило требования к сетевому трафику в геометрической прогрессии и породило множество новых телекоммуникационных услуг (например, электронная почта, веб-серфинг, социальные сети, потоковые сервисы, электронная коммерция, интернет вещей и т.д.). Эта быстро меняющаяся среда (наряду с определенными техническими определениями) очень затруднила составление точных прогнозов потребностей в трафике как на краткосрочную (внутридневную), так и на долгосрочную перспективу (из года в год). Проектирование телекоммуникационных сетей эволюционировало от сетей только для передачи голоса 1970-х годов, где методология исследования операций была незаменимой, до современных мультисервисных сетей передачи данных с коммутацией пакетов, неопределенность и быстрое развитие которых снизили полезность классических моделей оптимизации, которые полагаются на точные прогнозы спроса на трафик.

Постоянное и активное развитие современных информационных и сетевых технологий приводит к необходимости создания новых проектов и модернизации существующих компьютерных сетей. Учитывая сложность данных технологий и широкий спектр предлагаемого на рынке оборудования различных фирм производителей, при разработке проектов сетей необходим этап моделирования различных проектных решений с целью выбора оптимального варианта построения сети. Данный этап должен осуществляться при помощи современных инструментальных средств моделирования и описания проектов сетей

Имитационное моделирование является одним из видов математического моделирования, которое позволяет имитировать принципы работы сложных систем, в том числе и телекоммуникационных. Высокие темпы развития компьютерных технологий в настоящее время обусловили появление множества программных средств, которые позволяют создавать модели телекоммуникационных процессов, что приводит к проблеме выбора правильного программного обеспечения, которое позволит максимально быстро построить модель, соответствующую целям проводимого исследования. Использование пакетов имитационного моделирования не требует знания каких-либо сложных математических инструментов и позволяет строить модели достаточно высокого уровня сложности [1].

В данном учебном пособии представлены основные понятия коммутации каналов и пакетов, сети интернет и интранет, протоколы

передачи данных и стеки протоколов, определения на «Всеобъемлющий Интернет» - IoE и «Интернет вещей» - IoT. Изложен анализ имитационного моделирования телекоммуникационных сетей нового поколения и приведен сравнительный анализ из имеющихся на сегодняшний день программных продуктов моделирования сетей Cisco Packet Tracer, UNetLab.Cisco Packet Tracer, CupCarbon.

В учебном пособии приведены практические имитационные модели современных инфокоммуникационных сетей, реализованные в программном пакете CupCarbon, являющийся мощным инструментом моделирования телекоммуникационных сетей нового поколения.

# 1 Тенденции развития телекоммуникационных сетей нового поколения

## 1.1 Основные этапы развития телекоммуникационных сетей

XXI век можно смело назвать веком "информационного сообщества". С уверенностью можно констатировать все возрастающий интерес государств и общественных организаций к успехам телекоммуникационных технологий как к основе для создания единого информационного пространства (информационной инфраструктуры) планеты.

Сложилось понимание информационной инфраструктуры - важнейшего компонента любого вида деятельности как совокупности информационных ресурсов и программно-аппаратных средств вычислительной и телекоммуникационной техники, информационных технологий и телекоммуникационных сетей.

Среди важнейших факторов, оказывающих политическое воздействие на процесс создания информационного сообщества, можно отметить:

- разработку проектов создания глобальной международной информационной инфраструктуры комиссиями Европейского сообщества и совещаниями глав правительств - членов большой семерки;
- EITO - European Information Technology Observatory - широкомасштабную европейскую инициативу, задача которой - выработка всеобъемлющего взгляда на европейский рынок информационных технологий и оказание услуг, представляемых данной индустрии как отдельным пользователям, так и общественным организациям;
- RACE - R&D in Advanced Communications Technologies in Europe - общеевропейскую исследовательскую программу по созданию развитых коммуникационных технологий;
- программу создания национальной информационной инфраструктуры США - National Infrastructure Plan (1993 г.) и закон США о телекоммуникациях 1996 года.

В данный момент сложно предсказать, как будут выглядеть в будущем инфокоммуникационные сети, которые далеко шагнули в своем развитии уже сейчас. Но даже сейчас можно наблюдать перспективные разработки: мощные сети передач и коммутации пакетов, высокоскоростные линии доступа, оптические телекоммуникационные технологии и т.д., которые и определяют следующие поколения телекоммуникационных сетей. Выделяются три этапа развития телефонных сетей общего пользования, которые принято считать основными [1].

К сети первого поколения принято относить традиционные

телефонные сети, или POTS (Plain Old Telephone Service). Они объединяют в себе технологические и структурно-сетевые решения и используются для построения сетей до появления концепции цифровых сетей с интеграцией служб (Integrated Service Digital Network - ISDN). Все сети, которые используют аналоговые системы передачи и узлы коммутации декадно-шаговые, координатные, квазиэлектронные или являются ранней версией цифровых систем коммутации относят к POTS.

Сети второго поколения SDN стали развиваться в1980-х годах, после того как появились цифровые системы передачи. Но несмотря на создание интегральной сети, которая позволяла предоставляла различные виды услуг связи, основным приложением по - прежнему осталась услуга телефонии.

Для того чтобы организовать взаимодействия аппаратуры узлов коммутации между собой и с подключаемым терминальным оборудованием были установлены более мощные системы сигнализации. Они позволили передавать сигнальную информацию, связанную с установлением базового вызова, а также сведения, относящиеся к состоянию элементов сети связи, маршрутизации вызовов, согласованию параметров передачи и т.д. Так как до появления ISDN уже были созданы сетевые структуры в рамках POTS, то новое оборудование должно было взаимодействовать с существующими сетевыми фрагментами без снижения качества их работы и сокращения функциональных возможностей по предоставлению услуг доступа. Поэтому существующая сетевая структура для предоставления услуг телефонии до сих пор имеет в своем составе сетевые фрагменты как на основе решений POTS, так и на основе ISDN.

Появление Интернета привело к увеличению разветвленности и повышению емкости сети. Возникла потребность в изобретении сетевой структуры, такой же масштабной как телефонная сеть общего пользования (ТфОП). Но использование двух сетевых структур было экономически не выгодно. Поэтому необходимо было разработать технологию, которая обеспечит передачу различных видов информации и предоставление различных видов услуг связи в единой сетевой структуре. Этот метод передачи информации основан на коммутации пакетов. Так появились сети третьего поколения - сети NGN (Next Generation Network).

Сети третьего поколения NGN - гетерогенная мультисервисная сеть, основанная на пакетной коммутации, и обеспечивающая предоставление практически неограниченного спектра телекоммуникационных услуг. При этом NGN в качестве технических средств использует аппаратно - программные средства, ориентированные на стек протоколов TCP/IP. Традиционные сети не могут поддерживать обмен трафиком в формате IP, поэтому необходима реконструкция всей архитектуры сети: транспортной инфраструктуры, уровня доступа и сетевой иерархии [2].

## 1.2 Коммутация каналов и коммутация пакетов

Коммутация каналов. При коммутации каналов коммутационная сеть образует между конечными узлами непрерывный составной физический канал из последовательно соединенных коммутаторами промежуточных канальных участков. Условием того, что несколько физических каналов при последовательном соединении образуют единый физический канал, является равенство скоростей передачи данных в каждом из составляющих физических каналов. Равенство скоростей означает, что коммутаторы такой сети не должны буферизовать передаваемые данные.

В сети с коммутацией каналов перед передачей данных всегда необходимо выполнить процедуру установления соединения, в процессе которой и создается составной канал. И только после этого можно начинать передавать данные.

Техника коммутации каналов имеет свои достоинства и недостатки.

Достоинства коммутации каналов:

1. Постоянная и известная скорость передачи данных по установленному между конечными узлами каналу. Это дает пользователю сети возможности на основе заранее произведенной оценки необходимой для качественной передачи данных пропускной способности установить в сети канал нужной скорости.

2. Низкий и постоянный уровень задержки передачи данных через сеть. Это позволяет качественно передавать данные, чувствительные к задержкам (называемые также трафиком реального времени) - голос, видео, различную технологическую информацию.

Недостатки коммутации каналов:

1. Отказ сети в обслуживании запроса на установление соединения.

2. Нерациональное использование пропускной способности физических каналов. Та часть пропускной способности, которая отводится составному каналу после установления соединения, предоставляется ему на все время, т.е. до тех пор, пока соединение не будет разорвано.

3. Обязательная задержка перед передачей данных из-за фазы установления соединения. Достоинства и недостатки любой сетевой технологии относительны. В определенных ситуациях на первый план выходят достоинства, а недостатки становятся несущественными. Так, техника коммутации каналов хорошо работает в тех случаях, когда нужно передавать только трафик телефонных разговоров.

Коммутация пакетов. Эта техника коммутации была специально разработана для эффективной передачи компьютерного трафика.

Коэффициент пульсации трафика отдельного пользователя сети, равный отношению средней интенсивности обмена данными к максимально возможной, может достигать 1:50 или даже 1:100. При коммутации пакетов все передаваемые пользователем сообщения

разбиваются в исходном узле на сравнительно небольшие части, называемые пакетами. Сообщением называется логически завершенная порция данных - запрос на передачу файла, ответ на этот запрос, содержащий весь файл и т.д. Сообщения могут иметь произвольную длину, от нескольких байт до многих мегабайт. Напротив, пакеты обычно тоже могут иметь переменную длину, но в узких пределах, например от 46 до 1500 байт. Каждый пакет снабжается заголовком, в котором указывается адресная информация, необходимая для доставки пакета на узел назначения, а также номер пакета, который будет использоваться узлом назначения для сборки сообщения (рисунок 1.1) [3].

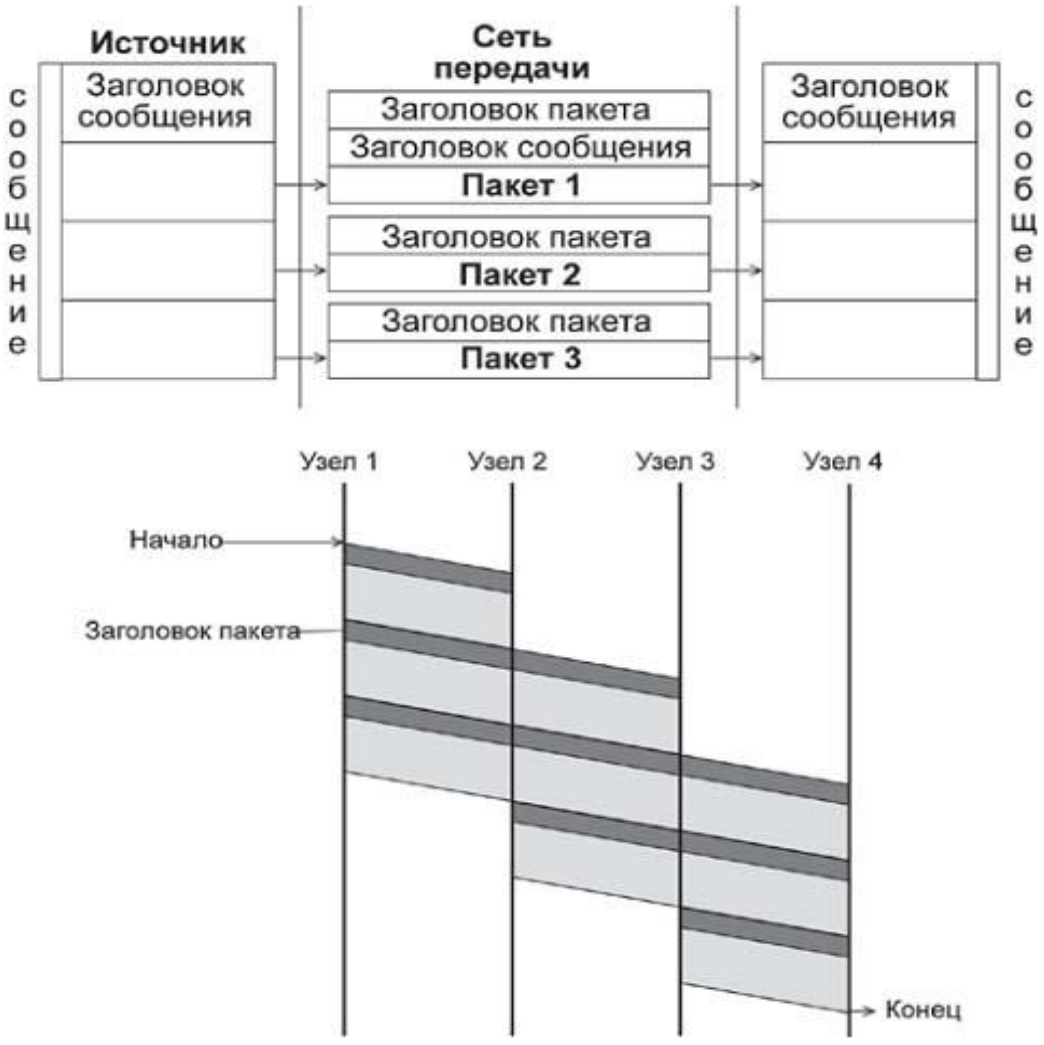



Рисунок 1.1 - Разбиение сообщения на пакеты

Пакеты транспортируются по сети как независимые информационные блоки. Коммутаторы сети принимают пакеты от конечных узлов и на основании адресной информации передают их друг другу, а в конечном итоге - узлу назначения.

Коммутаторы пакетной сети отличаются от коммутаторов каналов тем, что они имеют внутреннюю буферную память для временного

хранения пакетов, если выходной порт коммутатора в момент принятия пакета занят передачей другого пакета (рис. 3). В этом случае пакет находится некоторое время в очереди пакетов в буферной памяти выходного порта, а когда до него дойдет очередь, он передается следующему коммутатору. Такая схема передачи данных позволяет сглаживать пульсацию трафика на магистральных связях между коммутаторами и тем самым наиболее эффективно использовать их для повышения пропускной способности сети в целом.

Достоинства коммутации пакетов:

1. Высокая общая пропускная способность сети при передаче пульсирующего трафика.
2. Возможность динамически перераспределять пропускную способность физических каналов связи между абонентами в соответствии с реальными потребностями их трафика.

Недостатки коммутации пакетов:

1. Неопределенность скорости передачи данных между абонентами сети, обусловленная тем, что задержки в очередях буферов коммутаторов сети зависят от общей загрузки сети.
2. Переменная величина задержки пакетов данных, которая может быть достаточно продолжительной в моменты мгновенных перегрузок сети.
3. Возможные потери данных из-за переполнения буферов. В настоящее время активно разрабатываются и внедряются методы, позволяющие преодолеть указанные недостатки, которые особенно остро проявляются для чувствительного к задержкам трафика, требующего при этом постоянной скорости передачи. Такие методы называются методами обеспечения качества обслуживания (Quality of Service, QoS).

Сети с коммутацией пакетов, в которых реализованы методы обеспечения качества обслуживания, позволяют одновременно передавать различные виды трафика, в том числе такие важные как телефонный и компьютерный. Поэтому методы коммутации пакетов сегодня считаются наиболее перспективными для построения конвергентной сети, которая обеспечит комплексные качественные услуги для абонентов любого типа. Тем не менее, нельзя сбрасывать со счетов и методы коммутации каналов.

Сегодня они не только с успехом работают в традиционных телефонных сетях, но и широко применяются для образования высокоскоростных постоянных соединений в так называемых первичных (опорных) сетях технологий SDH и DWDM, которые используются для создания магистральных физических каналов между коммутаторами телефонных или компьютерных сетей. В будущем вполне возможно появление новых технологий коммутации, в том или ином виде комбинирующих принципы коммутации пакетов и каналов [3]. Сравнение методов коммутации приведены в таблице 1.1.

Таблица 1.1 - Сравнение коммутации каналов и коммутации пакетов

| Коммутация каналов | Коммутация пакетов |
|---|---|
| Гарантированная пропускная способность (полоса) для взаимодействующих абонентов | Пропускная способность сети для абонентов неизвестна, задержки передачи носят случайный характер |
| Сеть может отказать абоненту в установлении соединения | Сеть всегда готова принять данные от абонента |
| Трафик реального времени передается без задержек | Ресурсы сети используются эффективно при передаче пульсирующего трафика |
| Адрес используется только на этапе установления соединения | Адрес передается с каждым пакетом |

Постоянная и динамическая коммутация. Как сети с коммутацией пакетов, так и сети с коммутацией каналов можно разделить на два класса:

- сети с динамической коммутацией;
- сети с постоянной коммутацией.

В сетях с динамической коммутацией:

- разрешается устанавливать соединение по инициативе пользователя сети;
- коммутация выполняется только на время сеанса связи, а затем (по инициативе одного из пользователей) разрывается;
- в общем случае пользователь сети может соединиться с любым другим пользователем сети;
- время соединения между парой пользователей при динамической коммутации составляет от нескольких секунд до нескольких часов и завершается после выполнения определенной работы - передачи файла, просмотра страницы текста или изображения и т.п.

Примерами сетей, поддерживающих режим динамической коммутации, являются телефонные сети общего пользования, локальные сети, сети TCP/IP.

Сеть, работающая в режиме постоянной коммутации [4]:

- разрешает паре пользователей заказать соединение на длительный период времени;
- соединение устанавливается не пользователями, а персоналом, обслуживающим сеть;
- период, на который устанавливается постоянная коммутация, составляет обычно несколько месяцев;
- режим постоянной (permanent) коммутации в сетях с коммутацией каналов часто называется сервисом выделенных (dedicated) или арендуемых (leased) каналов;
- в том случае, когда постоянное соединение через сеть

коммутаторов устанавливается с помощью автоматических процедур, инициированных обслуживающим персоналом, его часто называют полупостоянным (semi-permanent) соединением, в отличие от режима ручного конфигурирования каждого коммутатора.

Наиболее популярными сетями, работающими в режиме постоянной коммутации, сегодня являются сети технологии SDH, на основе которых строятся выделенные каналы связи с пропускной способностью в несколько гигабит в секунду.

Некоторые типы сетей поддерживают оба режима работы. Например, сети X.25 и ATM могут предоставлять пользователю возможность динамически связаться с любым другим пользователем сети и в то же время отправлять данные по постоянному соединению определенному абоненту.

Пропускная способность сетей с коммутацией пакетов. Одним из отличий метода коммутации пакетов от метода коммутации каналов является неопределенность пропускной способности соединения между двумя абонентами. В случае коммутации каналов после образования составного канала пропускная способность сети при передаче данных между конечными узлами известна - это пропускная способность канала.

При выборе размера пакета необходимо также учитывать интенсивность битовых ошибок канала. На ненадежных каналах необходимо уменьшать размеры пакетов, так как это сокращает объем повторно передаваемых данных при искажениях пакетов. Ethernet - пример стандартной технологии коммутации пакетов

Дейтаграммная передача. В сетях с коммутацией пакетов сегодня применяется два класса механизмов передачи пакетов:

- дейтаграммная передача;
- виртуальные каналы.

Примерами сетей, реализующих дейтаграммный механизм передачи, являются сети Ethernet, IP и IPX. С помощью виртуальных каналов передают данные сети X.25, frame relay и ATM. Сначала мы рассмотрим базовые принципы дейтаграммного подхода.

Дейтаграммный способ передачи данных основан на том, что все передаваемые пакеты обрабатываются независимо друг от друга, пакет за пакетом. Принадлежность пакета к определенному потоку между двумя конечными узлами и двумя приложениями, работающими на этих узлах, никак не учитывается.

Выбор следующего узла - например, коммутатора Ethernet или маршрутизатора IP/IPX - происходит только на основании адреса узла назначения, содержащегося в заголовке пакета. Решение о том, какому узлу передать пришедший пакет, принимается на основе таблицы, содержащей набор адресов назначения и адресную информацию, однозначно определяющую следующий (транзитный или конечный) узел. Такие таблицы имеют разные названия - например, для сетей Ethernet они

обычно называются таблицей продвижения (forwarding table), а для сетевых протоколов, таких как IP и IPX - таблицами маршрутизации (routing table). Далее для простоты будем пользоваться термином "таблица маршрутизации" в качестве обобщенного названия такого рода таблиц, используемых для дейтаграммной передачи на основании только адреса назначения конечного узла.

В таблице маршрутизации для одного и того же адреса назначения может содержаться несколько записей, указывающих, соответственно, на различные адреса следующего маршрутизатора. Такой подход используется для повышения производительности и надежности сети. В примере на рисунке 1.2 пакеты, поступающие в маршрутизатор R1 для узла назначения с адресом N2, A2, в целях баланса нагрузки распределяются между двумя следующими маршрутизаторами - R2 и R3, что снижает нагрузку на каждый из них, а значит, уменьшает очереди и ускоряет доставку [5].

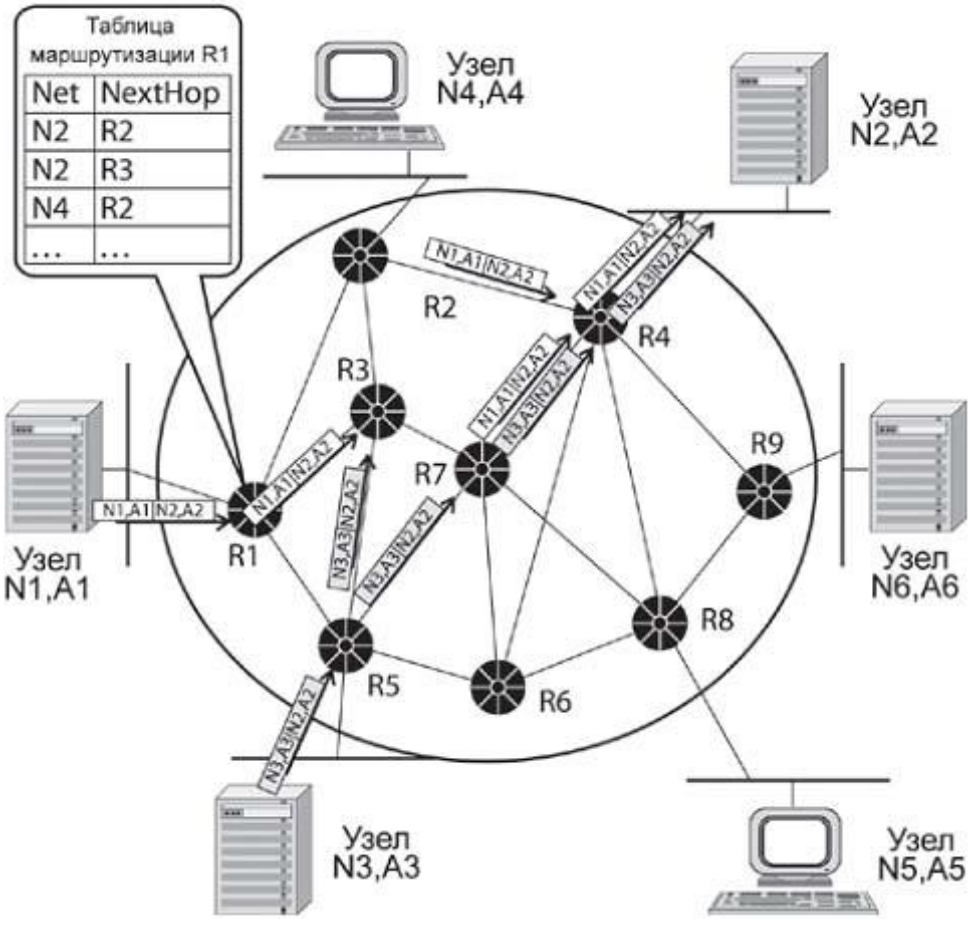



Рисунок 1.2 - Дейтаграммный принцип передачи пакетов

Некоторая "размытость" путей следования пакетов с одним и тем же адресом назначения через сеть является прямым следствием принципа независимой обработки каждого пакета, присущего дейтаграммным протоколам. Пакеты, следующие по одному и тому же адресу назначения,

могут добираться до него разными путями и вследствие изменения состояния сети, например отказа промежуточных маршрутизаторов.

Такая особенность дейтаграммного механизма как размытость путей следования трафика через сеть также в некоторых случаях является недостатком. Например, если пакетам определенной сессии между двумя конечными узлами сети необходимо обеспечить заданное качество обслуживания. Современные методы поддержки QoS работают эффективней, когда трафик, которому нужно обеспечить гарантии обслуживания, всегда проходит через одни и те же промежуточные узлы. Виртуальные каналы в сетях с коммутацией пакетов

Механизм виртуальных каналов (virtual circuit или virtual channel) создает в сети устойчивые пути следования трафика через сеть с коммутацией пакетов. Этот механизм учитывает существование в сети потоков данных.

Если целью является прокладка для всех пакетов потока единого пути через сеть, то необходимым (но не всегда единственным) признаком такого потока должно быть наличие для всех его пакетов общих точек входа и выхода из сети. Именно для передачи таких потоков в сети создаются виртуальные каналы. На рисунке 1.3 показан фрагмент сети, в которой проложены два виртуальных канала [5].

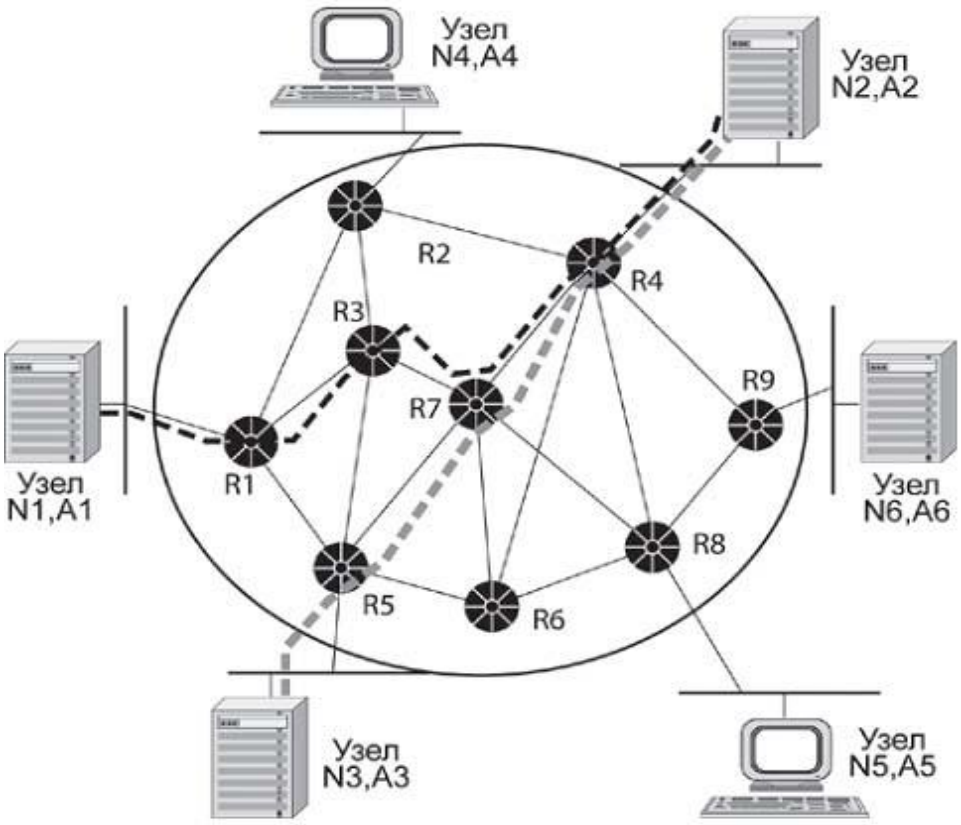



Рисунок 1.3 - Принцип работы виртуального канала

Первый проходит от конечного узла с адресом N1, A1 до конечного узла с адресом N2, A2 через промежуточные коммутаторы сети R1, R3, R7 и R4. Второй обеспечивает продвижение данных по пути N3, A3 - R5 - R7 - R4 - N2, A2. Между двумя конечными узлами может быть проложено несколько виртуальных каналов, как полностью совпадающих в отношении пути следования через транзитные узлы, так и отличающихся.

Сеть только обеспечивает возможность передачи трафика вдоль виртуального канала, а какие именно потоки будут передаваться по этим каналам, решают сами конечные узлы. Узел может использовать один и тот же виртуальный канал для передачи всех потоков, которые имеют общие с данным виртуальным каналом конечные точки, или же только части из них. Например, для потока реального времени можно использовать один виртуальный канал, а для трафика электронной почты - другой. В последнем случае разные виртуальные каналы будут предъявлять разные требования к качеству обслуживания, и удовлетворить их будет проще, чем в том случае, когда по одному виртуальному каналу передается трафик с разными требованиями к параметрам QoS.

Важной особенностью сетей с виртуальными каналами является использование локальных адресов пакетов при принятии решения о передаче. Вместо достаточно длинного адреса узла назначения (его длина должна позволять уникально идентифицировать все узлы и подсети в сети, например технология ATM оперирует адресами длиной в 20 байт) применяется локальная, то есть меняющаяся от узла к узлу, метка, которой помечаются все пакеты, перемещаемые по определенному виртуальному каналу. Эта метка в различных технологиях называется по-разному: в технологии X.25 - номер логического канала (Logical Channel number, LCN), в технологии frame relay - идентификатор соединения уровня канала данных (Data Link Connection Identifier, DLCI), в технологии ATM - идентификатор виртуального канала (Virual Channel Identifier, VCI). Однако назначение ее везде одинаково - промежуточный узел, называемый в этих технологиях коммутатором, читает значение метки из заголовка пришедшего пакета и просматривает свою таблицу коммутации, в которой указывается, на какой выходной порт нужно передать пакет.

Таблица коммутации содержит записи только о проходящих через данный коммутатор виртуальных каналах, а не обо всех имеющихся в сети узлах (или подсетях, если применяется иерархический способ адресации). Обычно в крупной сети количество проложенных через узел виртуальных каналов существенно меньше количества узлов и подсетей, поэтому по размерам таблица коммутации намного меньше таблицы маршрутизации, а, следовательно, просмотр занимает гораздо меньше времени и не требует от коммутатора большой вычислительной

мощности [6].

Идентификатор виртуального канала (именно такое название метки будет использоваться далее) также намного короче адреса конечного узла (по той же причине), поэтому и избыточность заголовка пакета, который теперь не содержит длинного адреса, а переносит по сети только идентификатор, существенно меньше.

### 1.3 Развитие сетей интернет и интранет

В октябре 1962 года Дж. Ликлайдер стал первым руководителем исследовательского компьютерного проекта в Управлении перспективных исследований и разработок Министерства обороны США (Defense Advanced Research Projects Agency, DARPA).

К концу 1969 года четыре компьютера были объединены в сеть, получившую название ARPANet. Предполагалось, что эта сеть будет объединять компьютеры военных научно-исследовательских и учебных заведений и использоваться для связи в случае третьей мировой войны. В последующие годы число компьютеров, подключенных к Arpanet, росло.

Параллельно с Arpanet развивались и другие компьютерные сети. Проблема заключалась в том, что все они работали по-разному. Для того чтобы они могли работать совместно, необходимо было выработать общий сетевой протокол.

В 1973 году была начата работа над проектом Internetting Project (Проект объединения сетей). Руководитель этого проекта Роберт Кан высказал идею открытой сетевой архитектуры. Открытая сетевая архитектура подразумевает, что отдельные сети могут проектироваться и разрабатываться независимо.

В ходе выполнения проекта был разработан протокол, удовлетворяющий требованиям окружения с открытой сетевой архитектурой. Этот протокол был впоследствии назван TCP/IP (Transmission Control Protocol/Internet Protocol - Протокол управления передачей/Межсетевой протокол) [6].

1 января 1983 года был осуществлен одновременный переход всех компьютеров в составе ARPANET на протокол TCP/IP. Так был установлен стандарт, согласно которому могла развиваться сеть Интернет, согласно которому она развивается и поныне.

Интернет является глобальной информационной паутиной со всеми вытекающими последствиями - обилие данных разного типа и разной степени полезности, широкие возможности для пользователя, отсутствие четкого контроля со стороны каких-либо органов власти и прочее.

Интранет можно назвать «мини-Интернетом» - это внутренняя сеть предприятия, основанная на таких же принципах, что и глобальная паутина. Как правило, интранет содержит сведения о сотрудниках и

партнерах компании, а также информацию о различных мероприятиях и нововведениях.

Сравнение. Во-первых, вполне очевидным является то, что Интернет является значительно более крупной и динамичной системой. Интранет - это замкнутая сеть, отрезанная от внешнего виртуального мира, и развивается она исключительно по инициативе людей, ее контролирующих. Интернет же не контролирует никто, поэтому развитие глобальной сети происходит исключительно естественным образом.

Интранет имеет куда менее богатый функционал, существование интранета посвящено одной-единственной функции - систематизация и обработка данных компании. Хотя можно сказать, что в больших организациях интранет выполняет и коммуникативную функцию. Поэтому, несмотря на схожесть системы работы и применяемых технологий, термины «Интернет» и «интранет» имеют всё же совершенно различные значения.

Internet предоставляет пользователям всевозможные информационные и коммуникационные услуги.

Информационные услуги - услуги доступа к информации:

- доступ к информационные ресурсам сети, то есть можно получить необходимую информацию, имеющуюся на серверах сети, например, документы, файлы, информацию из различных баз данных и т.п.;
- размещение собственной информации в сети. Существует множество серверов, предоставляющих возможность бесплатно разместить на них информацию.

Если информация размещается в целях публикации, то любые пользователи Internet могут получить доступ к этой информации и получать и просматривать ее в любое время.

Коммуникационные услуги - услуги обмена информацией, общения:

- обмен информацией в отсроченном режиме. Так работает, например, электронная почта. Отправитель направляет письмо в почтовый ящик получателя, который просмотрит это письмо в удобное для него время.
- обмен в режиме реального времени. Например, разговоры в сети. Люди набирают свои реплики с клавиатуры и посылают их на разговорный сервер, и эти реплики видят все участники разговора одновременно.

Эволюция вычислительных технологий в конце концов привела к объединению разрозненных ЭВМ в называемые вычислительные сети - совокупности обменивающихся информацией вычислительных машин. Пройдя путь от простых систем пакетной обработки и терминальных систем, построенных на базе мощных компьютеров и терминалов, до глобальных информационных сетей, построенных на технологиях открытых систем (OSI).

Под основными свойствами открытых систем понимаются: ν

переносимость и переиспользуемость программного обеспечения, данных, моделей и опыта; ν масштабируемость как свойство сохранения работоспособности системы ИТ в условиях варьирования значений параметров, определяющих технические и ресурсные характеристики системы и/или поддерживающей среды.

Виды компьютерных сетей [7]:

- ЛВС - Локальные Вычислительные Сети, Local Area Networks - LANs;
- СК - Сети кампусов - Campus Networks - CN;
- Городские Вычислительные Сети - Metropolian Area Networks - MAN;
- Глобальные Вычислительные Сети - Wide Area Networks - WAN.

Топологии сетей. Сетевая технология - это согласованный набор стандартных протоколов и программно-аппаратных средств (например, сетевых адаптеров, драйверов, кабелей и разъемов), достаточный для построения вычислительной сети.

Самым, на первый взгляд, универсальным способом объединения компьютеров в сеть является соединение по типу “полного графа”, при котором каждый участник сетевого взаимодействия непосредственно соединен со всеми другими участниками такового. Такой способ полностью решает проблему разделения доступа к физической среде передачи данных. К сожалению, указанный выше способ хорош только для ситуаций, когда в сети присутствует очень малое количество компьютеров. Чаще используются топологии типа “звезда” (star), “шина” (bus), “кольцо” (ring) и их комбинации.

Сеть можно усовершенствовать, например, за счет выделения в ней подсетей, что сразу потребует кроме протоколов стандарта Ethernet применения протокола IP, а также специальных коммуникационных устройств - маршрутизаторов. Усовершенствованная сеть будет, скорее всего, более надежной и быстродействующей, но за счет надстроек над средствами технологии Ethernet, которая составила базис сети.

Термин "сетевая технология" чаще всего используется в описанном выше узком смысле, но иногда применяется и его расширенное толкование как любого набора средств и правил для построения сети, например "технология сквозной маршрутизации", "технология создания защищенного канала", "технология IP-сетей".

Иногда сетевые технологии называют базовыми технологиями, имея в виду, что на их основе строится базис любой сети. Примерами базовых сетевых технологий могут служить наряду с Ethernet такие известные технологии локальных сетей как Token Ring и FDDI, или же технологии территориальных сетей X.25 и frame relay. Для получения работоспособной сети в этом случае достаточно приобрести программные и аппаратные средства, относящиеся к одной базовой технологии - сетевые адаптеры с драйверами, концентраторы,

коммутаторы, кабельную систему и т.п., и соединить их в соответствии с требованиями стандарта на данную технологию.

Итак, для сетевой технологии Ethernet характерны:

- коммутация пакетов;
- типовая топология "общая шина";
- плоская числовая адресация;
- разделяемая передающая среда.

Основной принцип, положенный в основу Ethernet, - случайный метод доступа к разделяемой среде передачи данных. В качестве такой среды может использоваться толстый или тонкий коаксиальный кабель, витая пара, оптоволокно или радиоволны (кстати, первой сетью, построенной на принципе случайного доступа к разделяемой среде, была радиосеть Aloha Гавайского университета).

В стандарте Ethernet строго зафиксирована топология электрических связей. Компьютеры подключаются к разделяемой среде в соответствии с типовой структурой "общая шина" (рисунок 1.4). С помощью разделяемой во времени шины любые два компьютера могут обмениваться данными. Управление доступом к линии связи осуществляется специальными контроллерами - сетевыми адаптерами Ethernet. Каждый компьютер, а точнее, каждый сетевой адаптер, имеет уникальный адрес. Передача данных происходит со скоростью 10 Мбит/с. Эта величина является пропускной способностью сети Ethernet [7].


Рисунок 1.4 - Сеть Ethernet

Суть случайного метода доступа состоит в следующем. Компьютер в сети Ethernet может передавать данные по сети, только если сеть свободна, то есть если никакой другой компьютер в данный момент не занимается обменом. Поэтому важной частью технологии Ethernet является процедура определения доступности среды.

После того как компьютер убедился, что сеть свободна, он начинает передачу и при этом "захватывает" среду. Время монопольного использования разделяемой среды одним узлом ограничивается временем передачи одного кадра. Кадр - это единица данных, которыми обмениваются компьютеры в сети Ethernet. Кадр имеет фиксированный формат и наряду с полем данных содержит различную служебную информацию, например адрес получателя и адрес отправителя.

Сеть Ethernet устроена так, что при попадании кадра в разделяемую среду передачи данных все сетевые адаптеры начинают одновременно принимать этот кадр. Все они анализируют адрес назначения, располагающийся в одном из начальных полей кадра, и, если этот адрес совпадает с их собственным, кадр помещается во внутренний буфер сетевого адаптера. Таким образом компьютер-адресат получает предназначенные ему данные.

Может возникнуть ситуация, когда несколько компьютеров одновременно решают, что сеть свободна, и начинают передавать информацию. Такая ситуация, называемая коллизией, препятствует правильной передаче данных по сети. В стандарте Ethernet предусмотрен алгоритм обнаружения и корректной обработки коллизий. Вероятность возникновения коллизии зависит от интенсивности сетевого трафика.

После обнаружения коллизии сетевые адаптеры, которые пытались передать свои кадры, прекращают передачу и после паузы случайной длительности пытаются снова получить доступ к среде и передать тот кадр, который вызвал коллизию.

Основные достоинства технологии Ethernet:

1. Главным достоинством сетей Ethernet, благодаря которому они стали такими популярными, является их экономичность. Для построения сети достаточно иметь по одному сетевому адаптеру для каждого компьютера плюс один физический сегмент коаксиального кабеля нужной длины.

2. Кроме того, в сетях Ethernet реализованы достаточно простые алгоритмы доступа к среде, адресации и передачи данных. Простота логики работы сети ведет к упрощению и, соответственно, снижению стоимости сетевых адаптеров и их драйверов. По той же причине адаптеры сети Ethernet обладают высокой надежностью.

3. И, наконец, еще одним замечательным свойством сетей Ethernet является их хорошая расширяемость, то есть возможность подключения новых узлов.

Другие базовые сетевые технологии, такие как Token Ring и FDDI, хотя и обладают индивидуальными чертами, в то же время имеют много общего с Ethernet. В первую очередь, это применение регулярных фиксированных топологий ("иерархическая звезда" и "кольцо"), а также разделяемых сред передачи данных.

Существенные отличия одной технологии от другой связаны с особенностями используемого метода доступа к разделяемой среде. Так, отличия технологии Ethernet от технологии Token Ring во многом определяются спецификой заложенных в них методов разделения среды - случайного алгоритма доступа в Ethernet и метода доступа путем передачи маркера в Token Ring [7].

## 1.4 Протоколы передачи данных, стеки протоколов

Модель OSI опубликовали в 1984 года как международный стандарт ISO 7498 и рекомендации X.200. Но разработка слишком затянулась, уже 1 января 1983 года минобороны США опубликовало распоряжение об обязательном использовании стека TCP/IP в сети ARPANET. Этот день считается датой рождения современного Интернета.

Информационный обмен - процесс многофункциональный. Родственные функции группируются по назначению и эти группы называют "уровнями взаимодействия". Унификация уровней позволяет создавать гетерогенные сети со сложной топологией. В основе унификации - понятие эталонной сетевой модели. Модель как таковая лишь описывает порядок сетевого взаимодействия, который реализуется в виде стека протоколов.

Обмен информацией между компьютерами, объединенными в сеть, очень сложная задача. Это связано с тем, что существует много производителей аппаратных и программных средств вычислительных систем. Единственный выход - унифицировать средства сопряжения систем, а именно использовать открытые системы. Открытая система взаимодействует с другими системами на основе единых общедоступных стандартов и спецификаций.

В 1984г. Международная Организация по Стандартизации (ISO) представила индустриальный стандарт - модель взаимодействия открытых систем (Open System Interconnection Reference Model - OSI/RM), чтобы помочь поставщикам создавать совместимые сетевые аппаратные и программные средства. В соответствии с этой моделью выделяются уровни, представленные на рисунке 1.5.

В соответствии с эталонной моделью OSI эти уровни взаимодействуют так, как показано на рисунок 1.6 Таким образом, сложная задача обмена информацией между компьютерами в сети разбивается на ряд относительно независимых и менее сложных подзадач взаимодействия между смежными уровнями.

Связь между уровнями двух сетевых узлов *(*горизонтальное взаимодействие*)* выполняется в соответствии с унифицированными правилами - протоколами взаимодействия

В автономной системе передача данных между уровнями *(*вертикальное взимодействие) реализуется через интерфейсы API.

Границу между сеансовым и транспортным уровнями можно рассматривать как границу между протоколами прикладного уровня и протоколами низших уровней. Если прикладной, представительный и сеансовый уровни обеспечивают прикладные процессы сеанса взаимодействия, то четыре низших уровня решают проблемы транспортировки данных.

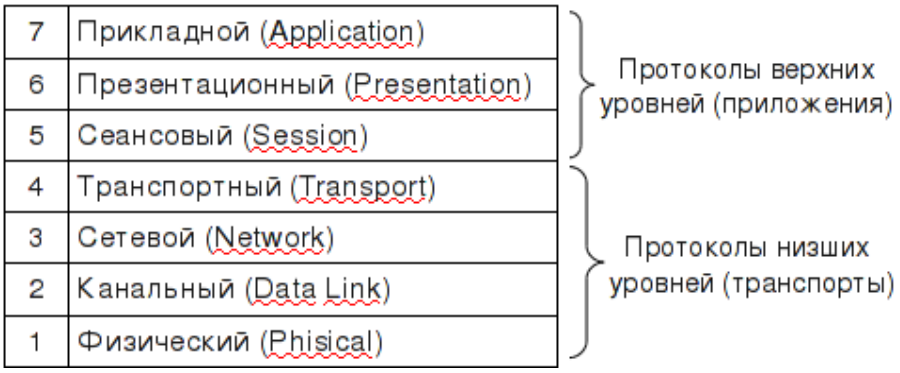



Рисунок 1.5 - Эталонная модель OSI

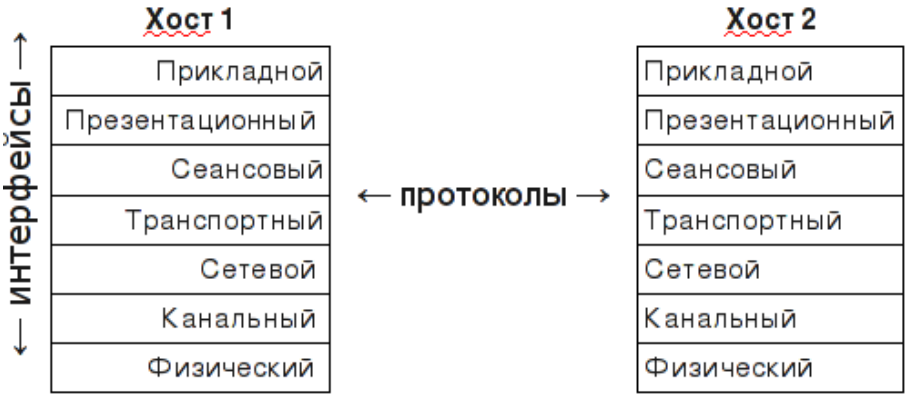



Рисунок 1.6 - Взаимодействие между уровнями OSI

Два самых низших уровня - физический и канальный - реализуются аппаратными и программными средствами, остальные пять более высоких уровней реализуются, как правило, программными средствами.

При передаче информации от прикладного процесса в сеть на физический уровень происходит ее обработка, которая заключается в разбиении передаваемых данных на отдельные блоки, преобразовании формы представления или кодировки данных в блоке и добавлении к каждому блоку заголовка (header) соответствующего уровня. Каждый заголовок характеризует используемый протокол обработки данных, причем каждый уровень воспринимает в качестве данных весь блок, полученный от предыдущего уровня, включая присоединенный заголовок. Такое построение эталонной модели позволяет заложить (*инкапсулировать*) в каждый передаваемый по физической среде информационный блок сведения, необходимые для выбора последовательности протоколов для осуществления обратных преобразований на принимающей информацию стороне [7].

Физический уровень. Этот уровень определяет механические, электрические, процедурные и функциональные характеристики установления, поддержания и размыкания физического соединения между конечными системами. Физический уровень определяет такие характеристики соединения, как уровни напряжений, синхронизацию и

физическую скорость передачи данных, максимальные расстояния передачи, конструктивные параметры разъемов и другие аналогичные характеристики. Известные стандарты RS-232-C, V.24 и IEEE 802.3 (Ethernet).

Канальный уровень. Канальный уровень (уровень звена данных, информационно-канальный уровень) отвечает за надежную передачу данных через физический канал, а именно:

- обеспечивает физическую адресацию (в отличие от сетевой или логической адресации);
- обеспечивает обнаружение ошибок в передаче и восстановление данных;
- отслеживает топологию сети и обеспечивает дисциплину использования сетевого канала конечной системой;
- обеспечивает уведомление о неисправностях;
- обеспечивает упорядоченную доставку блоков данных и управление потоком информации.

Для ЛВС канальный уровень разбивается на два подуровня:

- LLC (Logical Link Control) - обеспечивает управление логическим звеном, т.е. собственно функции канального уровня;
- MAC (Media Access Control) - обеспечивает специальные методы доступа к среде распространения.

Сетевой уровень. Этот уровень обеспечивает возможность соединения и выбор маршрута между двумя конечными системами, подключенными к разным подсетям (сегментам), которые могут быть разделены множеством подсетей и могут находиться в разных географических пунктах. Протоколы маршрутизации позволяют сети из маршрутизаторов выбирать оптимальные маршруты через связанные между собой подсети.

Транспортный уровень. Транспортный уровень обеспечивает высшим уровням услуги по транспортировке данных, а именно:

- обеспечивает надежную транспортировку данных через объединенную сеть;
- обеспечивает механизмы для установки, поддержания и упорядоченного завершения действия виртуальных каналов;
- обеспечивает обнаружение и устранение неисправностей транспортировки;
- следит за тем, чтобы конечная система не была перегружена слишком большим количеством данных [6].

Другими словами, транспортный уровень обеспечивает интерфейс между процессами и сетью, устанавливает логические каналы между процессами и обеспечивает передачу по этим каналам информационных блоков. Эти логические каналы называются транспортными.

Сеансовый уровень. Сеансовый уровень реализует установление, поддержку и завершение сеанса взаимодействия между прикладными

процессами абонентов. Сеансовый уровень синхронизирует диалог между объектами представительного уровня, определяет точки синхронизации для промежуточного контроля и восстановления при передаче файлов. Этот уровень также позволяет производить обмен данными в режиме, заданном прикладной программой, или предоставляет возможность выбора режима обмена.

Кроме основной функции управления диалогом сеансовый уровень предоставляет средства для выбора класса услуг и уведомления об исключительных ситуациях (проблемах сеансового, представительного и прикладного уровней).

Представительный уровень. Представительный уровень (уровень представления данных) определяет синтаксис, форматы и структуры представления передаваемых данных (но не затрагивает семантику, значение данных). Для того, чтобы информация, посылаемая из прикладного уровня одной системы, была читаемой на прикладном уровне другой системы, представительный уровень осуществляет трансляцию между известными форматами представления информации за счет использования унифицированного формата представления информации.

Таким образом, этот уровень обеспечивает служебные операции, выбираемые на прикладном уровне, для интерпретации передаваемых и получаемых данных: управление информационным обменом, отображение данных и управление структурированными данными. Эти служебные данные позволяют связывать воедино терминалы и вычислительные средства различных типов. Примером протокола этого уровня является XDR.

Прикладной уровень. В отличие от других уровней прикладной уровень - самый близкий к пользователю уровень OSI - не предоставляет услуги другим уровням OSI, однако он обеспечивает прикладные процессы, лежащие за пределами масштаба модели OSI.

Прикладной уровень обеспечивает непосредственную поддержку прикладных процессов и программ конечного пользователя (СУБД, текстовых процессоров, программ банковских терминалов и т.д. ) и управление взаимодействием этих программ с сетью передачи данных [7]:

- идентифицирует и устанавливает наличие предполагаемых партнеров для связи;
- синхронизирует совместно работающие прикладные программы;
- устанавливает соглашение по процедурам устранения ошибок и управления целостностью информации;
- определяет достаточность наличных ресурсов для предполагаемой связи.

Модель OSI не является реализацией, она лишь предлагает порядок организации взаимодействия между компонентами системы.

Реализациями этих правил являются стеки протоколов.

Протоколы стека OSI и их распределение по уровням сетевой модели приведены на рисунке 1.7.

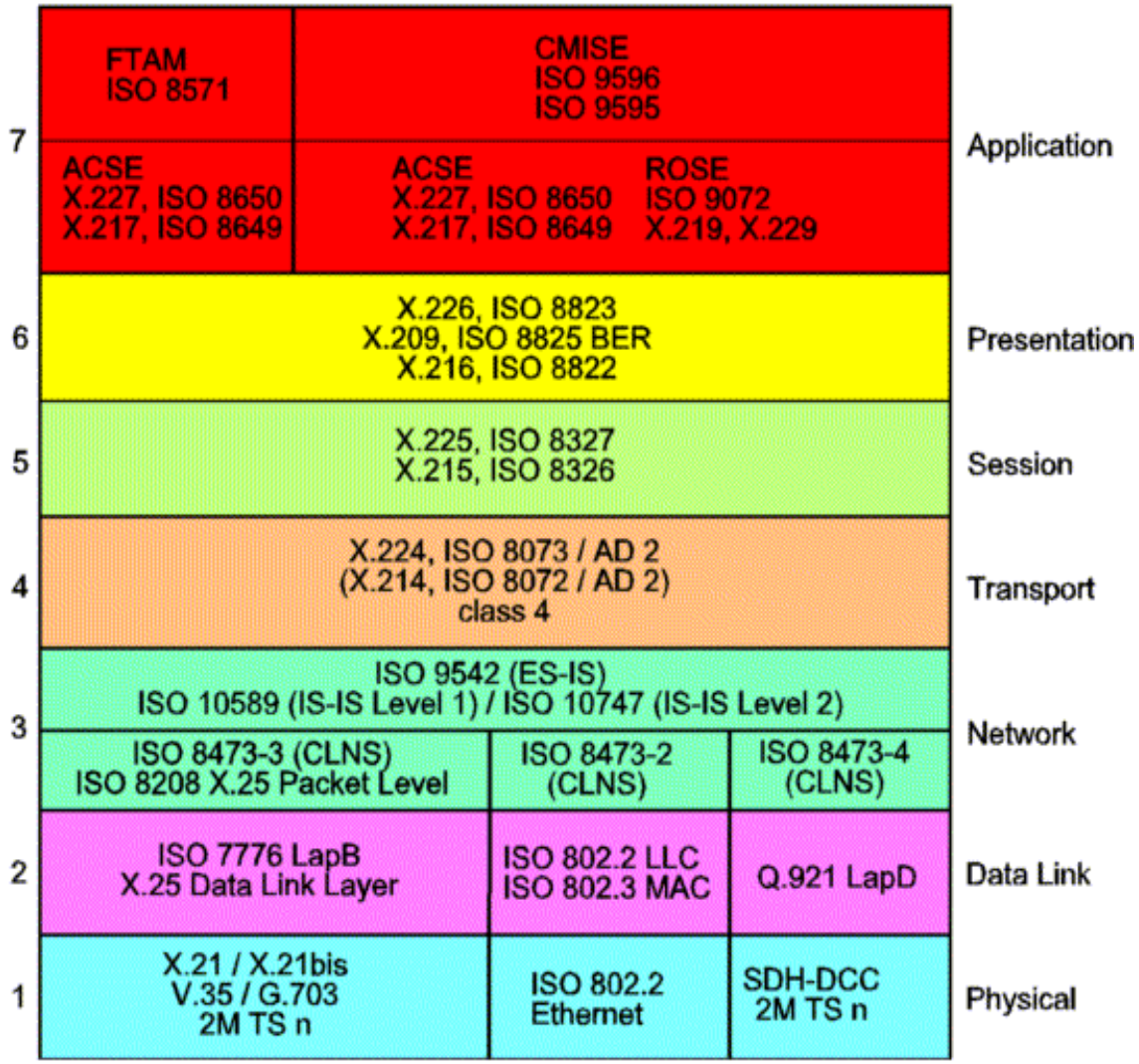



Рисунок 1.7 - Протоколы стека OSI ISO

Стек NetBIOS/SMB. Фирмы Microsoft и IBM совместно работали над сетевыми средствами для персональных компьютеров, поэтому стек протоколов NetBIOS/SMB является их совместным детищем.

Средства NetBIOS появились в 1984 году как сетевое расширение стандартных функций базовой системы ввода/вывода (BIOS) IBM PC для сетевой программы PC Network фирмы IBM, которая на прикладном уровне (рисунок 1.8) использовала для реализации сетевых сервисов протокол SMB.

Это ограничивает применение протокола NetBIOS локальными сетями, не разделенными на подсети. NetBIOS поддерживает как дейтаграммный обмен, так и обмен с установлением соединений.

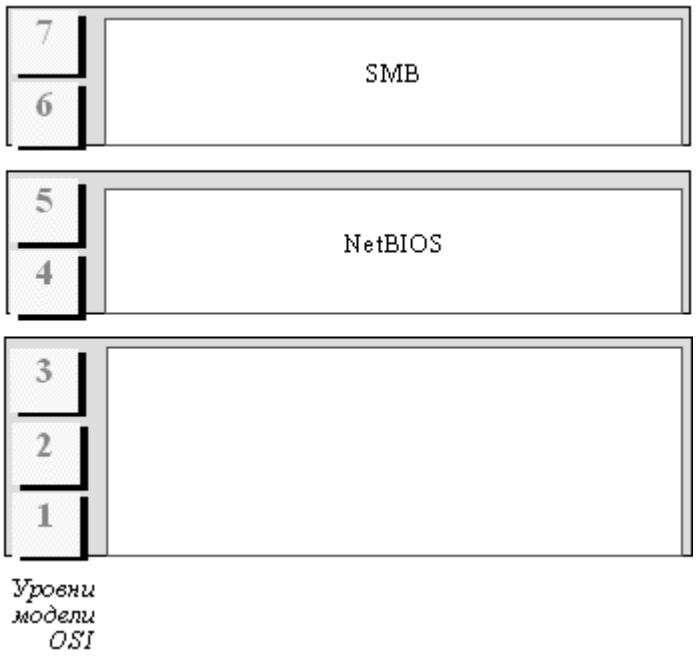



Рисунок 1.8 - Стек NetBIOS/SMB

Протокол SMB, соответствующий прикладному и представительному уровням модели OSI, регламентирует взаимодействие рабочей станции с сервером. В функции SMB входят следующие операции:

- Управление сессиями. Создание и разрыв логического канала между рабочей станцией и сетевыми ресурсами файлового сервера.
- Файловый доступ. Рабочая станция может обратиться к файл-серверу с запросами на создание и удаление каталогов, создание, открытие и закрытие файлов, чтение и запись в файлы, переименование и удаление файлов, поиск файлов, получение и установку файловых атрибутов, блокирование записей.
- Сервис печати. Рабочая станция может ставить файлы в очередь для печати на сервере и получать информацию об очереди печати.
- Сервис сообщений. SMB поддерживает простую передачу сообщений со следующими функциями: послать простое сообщение; послать широковещательное сообщение; послать начало блока сообщений; послать текст блока сообщений; послать конец блока сообщений; переслать имя пользователя; отменить пересылку; получить имя машины.

Из-за большого количества приложений, которые используют функции API, предоставляемые NetBIOS, во многих сетевых ОС эти функции реализованы в виде интерфейса к своим транспортным протоколам. В NetWare имеется программа, которая эмулирует функции NetBIOS на основе протокола IPX, существуют программные эмуляторы NetBIOS для Windows NT и стека TCP/IP.

Стек TCP/IP. Стек TCP/IP, называемый также стеком DoD и стеком

Internet, является одним из наиболее популярных стеков коммуникационных протоколов. Стек был разработан по инициативе Министерства обороны США (Department of Defence, DoD) для связи экспериментальной сети ARPAnet с другими сателлитными сетями как набор общих протоколов для разнородной вычислительной среды. Сеть ARPA поддерживала разработчиков и исследователей в военных областях. В сети ARPA связь между двумя компьютерами осуществлялась с использованием протокола Internet Protocol (IP), который и по сей день является основным в стеке TCP/IP и фигурирует в названии стека.

Большой вклад в развитие стека TCP/IP внес университет Беркли, реализовав протоколы стека в своей версии ОС UNIX. Широкое распространение ОС UNIX привело и к широкому распространению протокола IP и других протоколов стека. На этом же стеке работает всемирная информационная сеть Internet, чье подразделение Internet Engineering Task Force (IETF) вносит основной вклад в совершенствование стандартов стека, публикуемых в форме спецификаций RFC.

Так как стек TCP/IP был разработан до появления модели взаимодействия открытых систем ISO/OSI, то, хотя он также имеет многоуровневую структуру, соответствие уровней стека TCP/IP уровням модели OSI достаточно условно.

Структура протоколов TCP/IP приведена на рисунок 1.9. Протоколы TCP/IP делятся на 4 уровня.

Самый нижний (уровень IV) - уровень межсетевых интерфейсов - соответствует физическому и канальному уровням модели OSI. Этот уровень в протоколах TCP/IP не регламентируется, но поддерживает все популярные стандарты физического и канального уровня: для локальных каналов это Ethernet, Token Ring, FDDI, для глобальных каналов - собственные протоколы работы на аналоговых коммутируемых и выделенных линиях SLIP/PPP, которые устанавливают соединения типа "точка - точка" через последовательные каналы глобальных сетей, и протоколы территориальных сетей X.25 и ISDN. Разработана также специальная спецификация, определяющая использование технологии ATM в качестве транспорта канального уровня.

Следующий уровень (уровень III) - это уровень межсетевого взаимодействия, который занимается передачей дейтаграмм с использованием различных локальных сетей, территориальных сетей X.25, линий специальной связи и т. п.

В качестве основного протокола сетевого уровня (в терминах модели OSI) в стеке используется протокол IP, который изначально проектировался как протокол передачи пакетов в составных сетях, состоящих из большого количества локальных сетей, объединенных как локальными, так и глобальными связями.

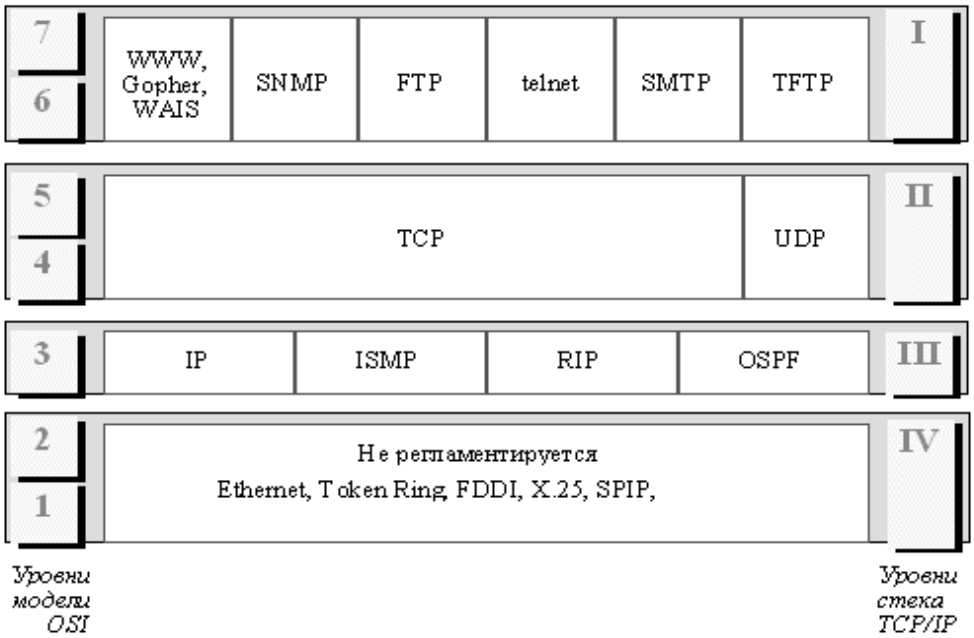



Рисунок 1.9 - Стек TCP/IP

Поэтому протокол IP хорошо работает в сетях со сложной топологией, рационально используя наличие в них подсистем и экономно расходуя пропускную способность низкоскоростных линий связи. Протокол IP является дейтаграммным протоколом.

К уровню межсетевого взаимодействия относятся и все протоколы, связанные с составлением и модификацией таблиц маршрутизации, такие как протоколы сбора маршрутной информации RIP (Routing Internet Protocol) и OSPF (Open Shortest Path First), а также протокол межсетевых управляющих сообщений ICMP (Internet Control Message Protocol). Последний протокол предназначен для обмена информацией об ошибках между маршрутизатором и шлюзом, системой-источником и системой-приемником, то есть для организации обратной связи. С помощью специальных пакетов ICMP сообщается о невозможности доставки пакета, о превышении времени жизни или продолжительности сборки пакета из фрагментов, об аномальных величинах параметров, об изменении маршрута пересылки и типа обслуживания, о состоянии системы и т.п.

Следующий уровень (уровень II) называется основным. На этом уровне функционируют протокол управления передачей TCP (Transmission Control Protocol) и протокол дейтаграмм пользователя UDP (User Datagram Protocol). Протокол TCP обеспечивает устойчивое виртуальное соединение между удаленными прикладными процессами. Протокол UDP обеспечивает передачу прикладных пакетов дейтаграммным методом, то есть без установления виртуального соединения, и поэтому требует меньших накладных расходов, чем TCP.

Верхний уровень (уровень I) называется прикладным. За долгие годы использования в сетях различных стран и организаций стек TCP/IP

накопил большое количество протоколов и сервисов прикладного уровня: протокол копирования файлов FTP, протоколы удаленного управления telnet и ssh, почтовый протокол SMTP, гипертекстовые сервисы доступа к удаленной информации, такие как WWW и многие другие. Кратко остановимся на некоторых из протоколов стека, наиболее тесно связанных с тематикой данного курса.

Протокол SNMP (Simple Network Management Protocol) используется для организации сетевого управления. Проблема управления разделяется здесь на две задачи. Первая задача связана с передачей информации. Протоколы передачи управляющей информации определяют процедуру взаимодействия сервера с программой-клиентом, работающей на хосте администратора. Они определяют форматы сообщений, которыми обмениваются клиенты и серверы, а также форматы имен и адресов. Вторая задача связана с контролируемыми данными. Стандарты регламентируют, какие данные должны сохраняться и накапливаться в шлюзах, имена этих данных и синтаксис этих имен. В стандарте SNMP определена спецификация информационной базы данных управления сетью. Эта спецификация, известная как база данных MIB (Management Information Base), определяет те элементы данных, которые хост или шлюз должен сохранять, и допустимые операции над ними.

Протокол пересылки файлов FTP (File Transfer Protocol) реализует удаленный доступ к файлу. Для того, чтобы обеспечить надежную передачу, FTP использует в качестве транспорта протокол с установлением соединений - TCP.

Кроме пересылки файлов протокол, FTP предлагает и другие услуги. Так пользователю предоставляется возможность интерактивной работы с удаленной машиной, например, он может распечатать содержимое ее каталогов, FTP позволяет пользователю указывать тип и формат запоминаемых данных. Наконец, FTP выполняет аутентификацию пользователей. Прежде, чем получить доступ к файлу, в соответствии с протоколом пользователи должны сообщить свое имя и пароль.

В стеке TCP/IP протокол FTP предлагает наиболее широкий набор услуг для работы с файлами, однако он является и самым сложным для программирования. Приложения, которым не требуются все возможности FTP, могут использовать другой, более экономичный протокол - простейший протокол пересылки файлов TFTP (Trivial File Transfer Protocol).

Этот протокол реализует только передачу файлов, причем в качестве транспорта используется более простой, чем TCP, протокол без установления соединения - UDP.

Протокол telnet обеспечивает передачу потока байтов между процессами, а также между процессом и терминалом. Наиболее часто этот протокол используется для эмуляции терминала удаленной ЭВМ [7].

## 1.5 Структурные элементы Интернет

Инфокоммуникационные сети типа Интернет состоят, в основном, из семи типов структурных элементов:

1) Повторители (repeaters) работают на физическом уровне модели ISO/OSI и обычно применяются в локальных сетях. Работают как усилители принятых сигналов и исполь-зуются также для увеличения длины сегмента в сетях топологии «шина». На рисунке 1.10 изображена типовая схема: благодаря повторителю длина сетевой шины увеличивается за счет сегмента 2.

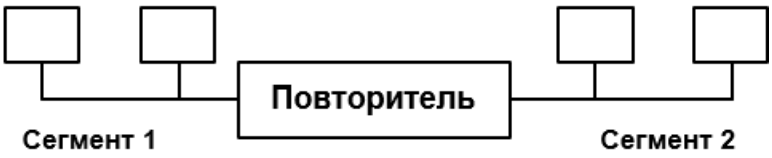



Рисунок 1.10 - Повторитель и сегменты

2) Мосты (bridges) работают на канальном уровне, имеющем подуровень MAC. В каждый узел сети входит плата сетевого интерфейса, имеющая уникальный MAC-адрес. Мосты выделяют MAC-адреса из принимаемых кадров данных и избирательно пересыла-ют эти кадры в соответствующие порты (рисунок 1.11) [8].

Мост игнорирует кадры, передаваемые между узлами, расположенными по одну сторону от него. Например, кадры, отправленные от узла 1 к узлам 2 и 3, мост пересылать не будет. Он отправляет кадры только узлам, находящимся по другую сторону от него, например, от узла 1 к узлу 4.

Одна из функций моста - выделять из кадров указания по их маршрутизации. Мост работает аналогично маршрутизатору, разница только в том, что мост устанавливает соединение на канальном уровне, а маршрутизатор - на сетевом.

Мост способен работать как накопительно-передающее устройство - принимать кадр целиком перед отправкой его в нужный порт. При этом кадр проверяется с помощью циклической контрольной суммы. Поврежденные кадры не пересылаются. Это снижает нагрузку на сеть, т. к. плохой кадр проходит только один сегмент и не попадает в другие.

3) Маршрутизаторы (routers) работают на сетевом уровне. В отличие от мостов, которые пересылают пакеты на основе таблиц физических адресов (например, адресов Ethernet), маршрутизаторы используют таблицы логических адресов (например, IP-адресов). Встречается термин многопротокольный маршрутизатор (multiprotocolrouter), который означает, что маршрутизатор понимает несколько протоколов сетевого уровня, например TCP и IPX.

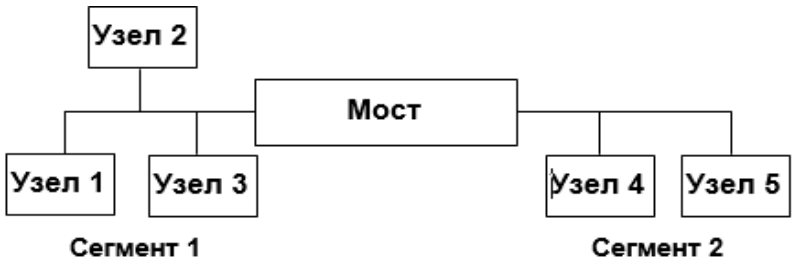



Рисунок 1.11 - Пример работы мостов

Разработчики отечественной аппаратуры для устройств, аналогичных маршрутизаторам, использовали термин «центр коммутации пакетов».

4) Коммутаторы (switch). В толковании этого термина есть разногласия. Согласно классическому определению на канальном уровне коммутатор работает почти как мост, но в отличие от моста он не накапливает кадр, а сразу после декодирования адреса назначения отправляет его в соответствующий порт. Этим достигается высокая скорость работы коммутатора. Недостаток в том, что коммутатор пересылает все кадры, даже поврежденные. Формально этот тип коммутатора называется LAN-коммутатором, но в литературе его часто именуют просто коммутатором. Современные коммутаторы определяют адрес сетевого уровня, который разными способами сопоставляется с портом коммутатора и последующие пакеты данных от того же отправителя к тому же получателю коммутируются уже на канальном уровне, в то время как маршрутизаторы делают это на сетевом уровне [8].

Другими словами, современный коммутатор больше напоминает скоростной маршрутизатор, а классический коммутатор - скоростной мост.

5) Шлюзы (gateway) - это, как правило, аппаратное и программное обеспечение, соединяющее две разные сети, в которых используются разные протоколы. Работают на сетевом и более высоких уровнях. Так называемые прикладные шлюзы при пересылке данных из одной сети в другую выполняют трансляцию протоколов, как это делает почтовый шлюз, конвертирующий два разных протокола электронной почты. Шлюз характеризуется наличием нескольких адресов сетевого уровня, например, нескольких IP-адресов.

6) Хост (host) - это компьютер, на котором работает сетевой протокол, например, TCP/IP. Хост обменивается данными с другими хост-компьютерами и значительная доля деятельности в Интернет обусловлена управлением информационными потоками между

хост-компьютерами. Типичные примеры хостов: маршрутизаторы, ПК, серверы, прокси-серверы, шлюзы и т. д.

7) Узлы. Этим термином, как правило, называют мост,

маршрутизатор, коммутатор, шлюз или хост.

Таким образом, в понятиях и терминологии традиционных сетей связи и компьютерных сетей пока еще имеются существенные различия, которые будут постепенно исчезать в процессе их конвергенции и перехода к единой терминологии инфокоммуникационных сетей [7].

## 1.6 Всеобъемлющий интернет, IoE и Интернет вещей, IoT

Всеобъемлющий Интернет (Internet of Everything, IoE) как сетевые соединения людей, процессов, данных и материальных объектов. Ценность Всеобъемлющего Интернета объясняется совокупным эффектом таких соединений и той пользой, которую они будут приносить по мере того, как все больше людей, процессов, данных и физических предметов будут охватываться Всемирной паутиной.

Интернетом же вещей (Internet of Things, IoT) обозначают лишь сетевые соединения физических предметов, то есть он не включает такие компоненты Всеобъемлющего Интернета, как «люди» и «процессы». Интернет вещей представляет собой переход от одной технологии к другой, а Всеобъемлющий Интернет - переход ко множеству технологий, включая Интернет вещей.

Сейчас за системой Всеобъемлющего интернета следят со всех уголков планеты, а крупные IT-компании стараются скорее внедрить его в производство: свои экосистемы создают Apple, Google, Yandex и другие корпорации. Рынок изобилует гаджетами и приложениями, связывающих устройства между собой в единую сеть. Концепция Всеобъемлющего интернета - это довольно утопическое представление о будущем мире, в котором все устройства связаны как единый организм, складно анализируют все возможные внешние и внутренние факторы, облекая их в данные и изменяя свое поведение в соответствии с полученной аналитикой.

Мы подходим к Всеобъемлющему интернету поэтапно: мобильная связь объединила нас в одну сеть, облачные вычисления дали возможность хранить и анализировать данные, не ограничиваясь при этом физическим расположением серверов, благодаря BigData мы научились обрабатывать терабайты данных и работать с ними. Сейчас мы на этапе внедрения Интернета вещей.

В 2021 году в мире насчитывалось более 10 миллиардов устройств Интернета вещей; а к 2025 году, по прогнозам IDC, общемировой объем генерации данных превысит 73 зеттабайта, или 73 триллиона гигабайт.

Интернет вещей (англ. *internet of things, IoT*) - концепция сети передачи данных между физическими объектами (*«вещами»*), оснащёнными встроенными средствами и технологиями для взаимодействия друг с другом или с внешней средой.

Устройства Интернета вещей - это наши глаза и уши там, где мы физически не можем быть. Они получают любые данные, на сбор которых запрограммированы. Затем мы собираем эти данные и анализируем их, чтобы получить информацию и автоматизировать последующие действия или решения. Этот процесс состоит из четырех ключевых этапов [8]:

1. Сбор данных. С помощью датчиков устройства Интернета вещей получают данные из своей среды. Эти данные могут быть простыми - такими как температура - или сложными, например, передача потокового видео в режиме реального времени.

2. Обмен данными. Используя доступные сетевые соединения, устройства Интернета вещей отправляют эти данные в общедоступную или частную облачную систему (устройство-система-устройство) или на другое устройство (устройство-устройство), либо хранят их локально для обработки на периферии.

3. Обработка данных. На этом этапе программные решения запрограммированы таким образом, чтобы сделать что-то на основе этих данных, например, включить вентилятор или отправить предупреждение.

4. Действия на основе данных. Здесь анализируются накопленные данные со всех устройств в сети Интернета вещей. Этот этап дает мощную аналитическую информацию для уверенных действий и принятия бизнес-решений.

Спектр возможных технологий передачи данных охватывает все возможные средства беспроводных и проводных сетей.

Для беспроводной передачи данных особо важную роль в построении «интернета вещей» играют такие качества, как эффективность в условиях низких скоростей, отказоустойчивость, адаптивность, возможность самоорганизации. Основной интерес в этом качестве представляет стандарт IEEE 802.15.4, определяющий физический слой и управление доступом для организации энергоэффективных персональных сетей, и являющийся основой для таких протоколов, как ZigBee, WirelessHart, MiWi, 6LoWPAN, LPWAN.

Среди проводных технологий важную роль в проникновении «интернета вещей» играют решения PLC - технологии построения сетей передачи данных по линиям электропередачи, так как во многих приложениях присутствует доступ к электросетям (например, торговые автоматы, банкоматы, интеллектуальные счётчики, контроллеры освещения изначально подключены к сети электроснабжения). 6LoWPAN, реализующий слой IPv6 как над IEEE 802.15.4, так и над PLC, будучи открытым протоколом, стандартизуемым IETF, отмечается как особо важный для развития «интернета вещей».

Полная система интернета вещей состоит из четырех отдельных компонентов. Датчики устройств, средства подключения, инструменты обработки данных и пользовательский интерфейс.

Датчики устройств.

Датчики устройств собирают данные в определенной среде. Устройство может иметь несколько датчиков, например, смартфон оснащен GPS, камерой, акселерометром и другими датчиками. Датчики собирают данные из окружающей среды для решения определенных задач.

Средства подключения.

После сбора данных устройство должно отправить их в облако. Это делается это по-разному: по Wi-Fi или Bluetooth, посредством спутниковой связи, через энергоэффективные сети дальнего радиуса действия (LPWAN) или при подключении напрямую к интернету через Ethernet. Вариант подключения зависит от области применения конкретного устройства интернета вещей.

Инструменты обработки данных.

Как только данные попадают в облако, осуществляется их программная обработка с целью последующего решения о выполнении определенных действий. Эти действия могут включать отправку предупреждений или автоматическую настройку датчиков устройства без участия пользователя. Однако иногда требуется ввод данных со стороны пользователя. В этом случае требуется пользовательский интерфейс [9].

Пользовательский интерфейс.

Интерфейс позволяет осуществить ввод данных со стороны пользователя или выполнить проверку работоспособности системы. Все действия пользователя передаются через систему: от пользовательского интерфейса в облако, а затем к датчикам устройств для внесения запрошенных изменений.

Протоколы подключения и сетевого взаимодействия, используемые веб-устройствами, различаются в зависимости от области применения устройства интернета вещей. Для упрощения и ускорения процессов сбора данных при работе интернета вещей все чаще используется искусственный интеллект и машинное обучение.

Архитектура.

Архитектура системы интернета вещей в упрощенном виде состоит из трех уровней: Уровень 1: Устройства, Уровень 2: Пограничный шлюз и Уровень 3: Облако. Устройства включают сетевые устройства, такие как датчики и исполнительные механизмы, используемые в оборудовании Интернета вещей, особенно те, которые используют такие протоколы, как Modbus, Bluetooth, Zigbee или собственные протоколы, для подключения к пограничному шлюзу. Уровень пограничного шлюза состоит из систем агрегирования данных датчиков, называемых пограничными шлюзами, которые обеспечивают функциональность, такую как предварительная обработка данных, обеспечение подключения к облаку, использование таких систем, как WebSockets, концентратор событий и, даже в некоторых случаях, пограничная аналитика или облачные вычисления. Уровень пограничного шлюза также необходим для предоставления общего

представления об устройствах на верхних уровнях для облегчения управления. Последний уровень включает облачное приложение, созданное для Интернета вещей с использованием архитектуры микросервисов, которые обычно являются многоязычными и по своей сути безопасными с использованием HTTPS/OAuth. Он включает в себя различные системы баз данных, которые хранят данные датчиков, такие как базы данных временных рядов или хранилища активов с использованием внутренних систем хранения данных (например, Cassandra, PostgreSQL). Облачный уровень в большинстве облачных систем Интернета вещей включает систему организации очередей событий и обмена сообщениями, которая обрабатывает связь, происходящую на всех уровнях. Некоторые эксперты классифицировали три уровня в системе интернета вещей как пограничный, платформенный и корпоративный, и они связаны сетью близости, сетью доступа и сетью обслуживания соответственно [9].

Основываясь на Интернете вещей, web of things - это архитектура прикладного уровня Интернета вещей, ориентированная на конвергенцию данных с устройств Интернета вещей в веб-приложения для создания инновационных вариантов использования. Для программирования и управления потоком информации в Интернете вещей прогнозируемое архитектурное направление называется BPM Everywhere, которое представляет собой сочетание традиционного управления процессами с интеллектуальным анализом процессов и специальными возможностями для автоматизации управления большим количеством скоординированных устройств.

По данным исследования IoT Analytics, в 2022 году самый высокий уровень проникновения технологии IoT наблюдался в транспорте, энергетике, ретейле, управлении жизнью города, здравоохранении и промышленности.

Примеры систем IoT, используемых сегодня: Носимые устройства, умные дома, умные города, беспилотные автомобили, розничная торговля, телемедицина, умное сельское хозяйство, производство [10].

Преимущества интернета вещей

Эффективность. Взаимодействие между устройствами повышает эффективность процессов и экономит время людей, позволяя им работать над другими задачами.

Автоматизация. Автоматизированное выполнение единообразных задач может повысить качество обслуживания и снизить потребность в человеческом вмешательстве.

Снижение издержек. Повышение эффективности и автоматизация процессов может позволить сократить как отходы, так и трудозатраты, что удешевляет производство и доставку товаров.

Контроль качества. Интернет вещей улучшает обмен данными между устройствами и обеспечивает лучший контроль качества.

Прозрачность. Возможность доступа к информации из любого места, в любое время, с любого устройства упрощает принятие решений и увеличивает прозрачность.

Недостатки интернета вещей.

Совместимость. Отсутствие международных стандартов совместимости может привести к возникновению проблем при взаимодействии устройств разных производителей.

Снижение количества рабочих мест. Интернет вещей ускоряет автоматизацию, в результате чего может сократиться количество требуемых рабочих мест.

Сложность. В огромной сети интернета вещей всего один сбой в программном или аппаратном обеспечении может привести к катастрофическим последствиям.

Конфиденциальность и безопасность. Значительное количество ежедневно используемых подключенных к интернету устройств ведет к тому, что в сети хранится существенный объем информации. Это создает риски конфиденциальности и безопасности, которые описаны ниже более подробно.

В будущем можно рассчитывать на более тесную интеграцию технологий и опыта сотрудников. Хотя до мета вселенной еще несколько лет, благодаря 3D-аудио, передовым возможностям виртуальной реальности, тактильным ощущениям и персонализации в реальном времени на базе ИИ взаимодействие человека с окружающими его устройствами позволит ему получать все более реалистичный сенсорный опыт. Кроме того, с развитием технологии 5G и повсеместным распространением быстрой связи люди получат способность делиться этими ощущениями на любом расстоянии. Последствия этого грандиозны и потенциально способны изменить наши подходы к некоторым из наиболее фундаментальных видов деятельности и институтов, таких как рабочие места, хирургическая и медицинская помощь, недвижимость, торговля, путешествия и отношения с людьми в целом [11].

## 1.7 Имитационное моделирование телекоммуникационных сетей нового поколения

Повсеместное создание компьютерных сетей для обеспечения пользователей удалённым доступом к ресурсам сети обуславливает резкое развитие моделирования сетей с помощью программных эмуляторов. Фактически все компании, имеющие более одного компьютера, объединяют их в локальные сети, чтобы она работала бесперебойно, была надёжной, лучше обрабатывала информацию, циркулирующую между сотрудниками компании, и позволяла принимать им значимые и оптимальные решения. Для этого разрабатывается сетевое

оборудование, как разные маршрутизаторы, коммутаторы различных уровней.

Высокая стоимость оборудования, сегодня является одним из камней преткновения в исследованиях в телекоммуникации. Не каждая лаборатория может позволить себе покупку необходимого оборудования и в достаточном количестве, для проведения тестирования новых протоколов, решений по оптимизации архитектур, подбора определённых топологий для применения новых сетевых решений. Именно поэтому были созданы программы, позволяющие выполнять имитационное моделирование телекоммуникационных систем. Создание подобного рода продукции позволило проводить необходимые исследования и эксперименты намного экономнее и получать практически те же результаты, что и на реальном оборудовании. Кроме экономии, использование симуляторов позволяет проводить эксперименты, не строя реальную сеть, который достаточно трудоёмкий и требовательный ко времени процесс. В программных продуктах можно использовать лишь определённые модули оборудования, что подтверждает выигрыш от их использования [12].

Основной особенностью программных пакетов, позволяющих имитировать работу сетей связи, является возможность отслеживания (трассировки) событий, происходящих на различных уровнях (физическом, канальном, сетевом и т.д.) модели сети. Они призваны достаточно точно воспроизводить все основные особенности и параметры реального оборудования. Имитационные модели позволяют значительное сокращение времени исследования моделируемых систем за счет подмены процесса, происходящего в реальном масштабе времени, на ускоренный или замедленный процесс смены событий в симуляторе. Кроме того, симуляторы позволяют оценить работу сети в значительно более широком диапазоне изменения ее параметров, таких как число сетевых узлов, типы используемых протоколов, характеристики среды распространения радиосигналов и многих других. Имитационное моделирование незаменим и для анализа работоспособности сложных стохастических систем, какими и являются мобильные самоорганизующиеся сети связи. В отличие от математической модели, имитационная модель не всегда содержит в себе строгие математические описания всех зависимостей между параметрами моделируемых процессов. Напротив, имитационное моделирование используются тогда, когда строгое математическое описание процессов в моделируемой системе невозможно или крайне затруднено, когда речь идет о сложной системе, зависимости между параметрами которой не известны в необходимой степени, а полное воспроизведение их приводит к излишне громоздким системам уравнений. В таких системах подробное математическое описание используется лишь для некоторых уровней моделирования. Например, физический слой, описывающий среду

передачи, можно описать как имитационной моделью, что будет весьма ресурсоемко, так и аналитическим выражением. Сетевые симуляторы в ходе своей работы могут собирать статистику о наиболее важных событиях, происходящих в модели: количестве потерянных пакетов данных, времени задержки распространения пакетов, коэффициентах использования каналов и узлов и т.п [12].

На данный момент рынок программных пакетов для имитационного моделирования сетей наполнен множеством решений. По области решаемых задач эти решения варьируются от небольших узкоспециализированных утилит, нацеленных на моделирование конкретного типа сетей (например, только беспроводных сенсорных сетей) до крупных программных пакетов, стремящихся охватить все многообразие телекоммуникационных сетей. С позиции разработки моделей возникает дилемма. С одной стороны, необходимо использовать низкоуровневые языки, например, Си или Си++. Программный код на них хорошо оптимизируется и требует малых вычислительных затрат. Проблемы же низкоуровневых языков заключаются в сложности разработки и необходимости времени для компилирования модели. Это заметно снижает темпы разработки. С другой стороны, использование высокоуровневых языков позволяет упростить программный код и вносить поправки без перекомпилирования, однако скорость работы таких языков ощутимо ниже. В качестве таких языков могут использоваться Tcl, Python, Java. Различные разработчики программных продуктов решают эту задачу по-своему. Некоторые используют связку из двух языков: один, низкоуровневый, они отводят для описания ресурсоемких частей модели, которые не требуют постоянного редактирования (узлы, физические модели), другой, высокоуровневый - для часто изменяемых частей модели (топология сети). Сложность такого решения заключается в необходимости специальных средств для сопряжения этих языков. Второй способ решения - использование средств, ускоряющих работу высокоуровневых языков, таких как «байт-код». Таким образом, за счет времени, потраченного на компиляцию, повышается общая скорость работы модели. Третий способ - надстройка над существующими низкоуровневыми языками специализированных фреймворков, упрощающих разработку различных частей модели. Из-за повышения уровня абстракции несколько снижается скорость модели, однако ощутимо упрощается разработка. Также выбором языка программирования определяются такие параметры, как мультиплатформенность и возможность параллельного вычисления на кластерах.

Существует ряд систем имитационного моделирования, которые не дают пользователю доступа к уровню программирования. Такие системы обладают малой гибкостью и применимы для очень узкого круга задач. С точки зрения политики лицензирования, можно выделить платные

пакеты, пакеты бесплатные для академического использования и пакеты, распространяющиеся свободно. Платные пакеты являются собственностью компаний-разработчиков, которые позволяют пользоваться своим продуктом за деньги. Стоимость такого пакета может достигать нескольких десятков тысяч долларов США.

Существует множество программных продуктов, разрешающих бесплатное использование в некоммерческих целях для частных лиц, некоммерческих организаций, учебных заведений и т.д. Однако они требуют оплаты в случае использования программного продукта с целью извлечения прибыли. Такие ограничения оговариваются в лицензии продукта. Исходный код находится либо в свободном доступе, либо высылается по запросу.

Цена, как правило, не указана и является предметом договора пользователя с разработчиками. Наиболее предпочтительные, с этой точки зрения, так называемые открытые программные пакеты. Исходный код таких программ доступен для просмотра, изучения и изменения, что позволяет пользователю принять участие в доработке самой программы, использовать код для создания новых программ и исправления в них ошибок через заимствование или через изучение использованных алгоритмов и методов [13].

Так появилась необходимость в применении программных эмуляторов сетевого оборудования для создания и администрирования моделей сетей. На сегодняшний день известно достаточно много сетевых симуляторов и у пользователей есть возможность выбора. Одними из популярных продуктов являются OPNET, OMNET, OMNET++, NS2, NS3, PacketTracer, созданный компанией Cisco, dynamips, созданный для эмуляции маршрутизаторов Cisco и GNS3, который представляет собой графический интерфейс для симулятора dynamips.

Функциональный потенциал самых распространенных сегодня эмуляторов, как Cisco Packet Tracer и UNetLab.Cisco Packet Tracer - это мощный программный эмулятор, дающий возможность пользователям моделировать сети, организовывая их с практически безграничным количеством устройств, находить применение оборудованию и налаживать его под определенные задачи той или иной среды. Программа дает возможность вырабатыванию качества скорости принятия решения, креативного подхода и критического мышления, настраивать конфигурацию и устранять неполадки сетей с применением виртуального оборудования и имитацией соединения. Не смотря на столь большие преимущества данного программного эмулятора, есть и недостатки, так, например, в контексте «умных городов» и IoT их модели радиоканалов и помех очень просты и не учитывают реальную городскую среду. Кроме того, программа не интегрируют визуализацию, чтобы легко проверить разработанный алгоритм, например, отсутствие возможности моделирования в 2D/3D среде и добавления городских и естественных

помех; не полная эмуляция IOS; практически всё, что выходит за рамки CCNA, на нем собрать тоже не получится [14], возможные проявления разнообразных глюков, которые лечатся только перезапуском программы [15].

Программный эмулятор UNenLab (Unified Networking Lab, UNL) - это многопользовательская платформа для создания и моделирования самых различных лабораторий и дизайнов, позволяющая смоделировать виртуальную сеть из маршрутизаторов, коммутаторов, устройств безопасности и др. Использование и анализ данного эмулятора, выявил существенные плюсы в возможности его применения: полностью бесплатен; практически полноценная поддержка L2; широкая поддержка Cisco оборудования; число запускаемых узлов ничем неограничен; многопользовательский функционал; низкие требования к ресурсам ПК и т.д.

В образовательных целях, данная платформа подойдет как новичкам для подготовки к CCNA/CCNP, так и для профессионалов для подготовки CCIE Routing and Switching, CCIE Security, CCIE Service Provides, CCIE Data Centers и т.д, а также для других разнообразных инженерных и образовательных задач [15].

В отличие от предыдущего эмулятора IOU-WEB, в UNetLAB реализован полностью графический интерфейс дизайна топологии, как в GNS, т.е. нет необходимости писать команду netmap для каждой топологии. Сегодня UNetLab остается лучшим инструментом как для подготовки к CCNP/CCIE, так и для моделирования разнообразных инженерных задач.

Стоит отметить и ряд его существенных недостатков, как трудоемкий процесс установки; отсутствие процесса визуализации моделирования; отсутствие возможности использования на реальных картах местности; отсутствие совместимости с предыдущим проектом программы; отсутствие добавления городских и естественных помех в процесс моделирования. Проведенный функциональный и сравнительный анализ эмуляторов CiscoPacketTracer и UnetLab, выявил, что данные эмуляторы не удовлетворяют требованиям, предъявляемые при моделировании сетей для IoT, так как они не визуализируют процесс моделирования, не имеют возможности использования на реальных картах местности и добавления городских и естественных помех.

Одним из самых распространённых мощных инструментов моделирования телекоммуникационных сетей, является CupCarbon. Альтернативная им платформа CupCarbon-Lab, основанная на существующем симуляторе CupCarbon, предназначенная для проектирования и моделирования беспроводных сенсорных сетей для приложений Smartcity и IoT. Она должна позволить проверять распределенные алгоритмы в 2D/3D-среде с учетом городских зданий, в которых будут развернуты сети, мобильные телефоны, использовать

точные модели распространения радиоволн и помех в этой среде [16]. Платформа может автоматически генерировать из программного обеспечения реальную сеть IoT, даже если она уже развернута, перенастроить без необходимости прохождения через каждый узел, а также поможет проверить выполнимость и масштабируемость алгоритма в реальных условиях [13].

Это универсальная система и имитатор беспроводной сенсорной сети дискретных событий. Сети можно конструировать и проектировать в эргономичном удобном интерфейсе с использованием OpenStreetMap путем развертывания датчиков непосредственно на карте. Основными задачами CupCarbon являются также и образовательные, т.е. оказание помощи тренерам и преподавателям объяснить основные концепции IoT, работу сенсорных сетей, проверить беспроводные топологии, протоколы, изучить поведение сети и ее элементов, например, для изучения схемы питания каждого датчика и всей сети в целом, расчета силовых схем и отображения, как функции от смоделированного времени [17]. Проектирование сетей на этой универсальной системе и имитаторе беспроводной сенсорной сети дискретных событий является более реалистичным по сравнению с рассмотренными нами раннее эмуляторами, как CiscoPacketTracer и UnetLab [18].

Они используются главным образом для разработки новых протоколов маршрутизации, но в контексте «умных городов» и IoT, их модели радиоканалов и помех очень просты и не учитывают реальную городскую среду, а также они не интегрируют визуализацию, чтобы легко проверить разработанный алгоритм (Рисунок 1.12).

| Возможности эмулятора | Cisco Packet Tracer | UnetLAB | CupCarbon |
|---|---|---|---|
| Симуляция сети в режиме реального времени | ✓ | ✓ | ✓ |
| Добавление естественных условий | — | — | ✓ |
| Работа в 3-D среде | — | — | ✓ |
| Поддержка картографии | — | — | ✓ |

Рисунок 1.12 - Сравнительный анализ эмуляторов

# 2 Программный эмулятор CupCarbon и его потенциал

Рассмотренные ранее симуляторы, такие как Cisco Packet Tracer и UNetLab, используются главным образом для разработки новых протоколов маршрутизации. Основным вкладом этой междисциплинарной работы является сохранение очень короткого времени моделирования с учетом 3D, точно смоделированного радиоканала с воздействием препятствий на окружающую среду и реалистичной оценки помех. Платформа была разработана со следующими целями:

- изучить развертывание беспроводных сенсорных сетей с учетом мобильности и доступности спектра;
- имитация производительности и услуг беспроводной сенсорной сети в 2d/3d реалистичной среде;
- изучить возможности связи, надежность сети и ее стоимость;
- обнаруживать любые зоны помех для улучшения качества развертывания, точно и быстро моделировать распространение радиосигнала в реальной городской среде.

Узел беспроводного датчика в основном состоит из микроконтроллера, нескольких датчиков, аккумулятора и радио-модуля. Это разработано, чтобы снизить потребление энергии, так как очень важна продолжительность автономной работы датчика. Одним из способов снижения этого потребления является уменьшение радиосвязи, как в случае со стандартом ZigBee 802.15.4, который разработан специально для приложений беспроводной сенсорной сети (WSN). Поскольку дальность радиосвязи мала, требуется использовать множество сенсорных узлов, чтобы направлять любую информацию координатору или базовой станции. Поскольку эти узлы обмениваются данными по беспроводной сети, предлагается использовать недорогой алгоритм связи. Другой способ - направить сообщения непосредственно на базовую станцию всего за один переход. Это возможно благодаря использованию недорогих и беспроводных модулей связи, как в случае протоколов LoRa и SigFox.

Поскольку ожидается, что в ближайшем будущем более 50% населения будет проживать в городах, количество подключенных устройств значительно возрастет. Это приведет к использованию огромного количества устройств, осуществляющих беспроводную связь. Поэтому наша среда будет насыщена с точки зрения коммуникационных сигналов и спектра. Это может быть ограничением для любой будущей установки WSN. Именно поэтому необходимо тщательно изучить любой новый проект перед его реальной установкой. Для этого необходимо использовать симуляторы для изучения влияния, с точки зрения перегрузки и безопасности беспроводного сигнала, любой новой установки до ее реального развертывания [11].

# 3 Пользовательский интерфейс и карта эмулятора

Графический интерфейс пользователя состоит из 5 основных частей в соответствии рисунком 3.1:

1. Карта (в центре);
2. Строка меню (вверху);
3. Панель инструментов (под меню);
4. Меню параметров (слева);
5. Строка состояния (внизу);
6. Консоль.

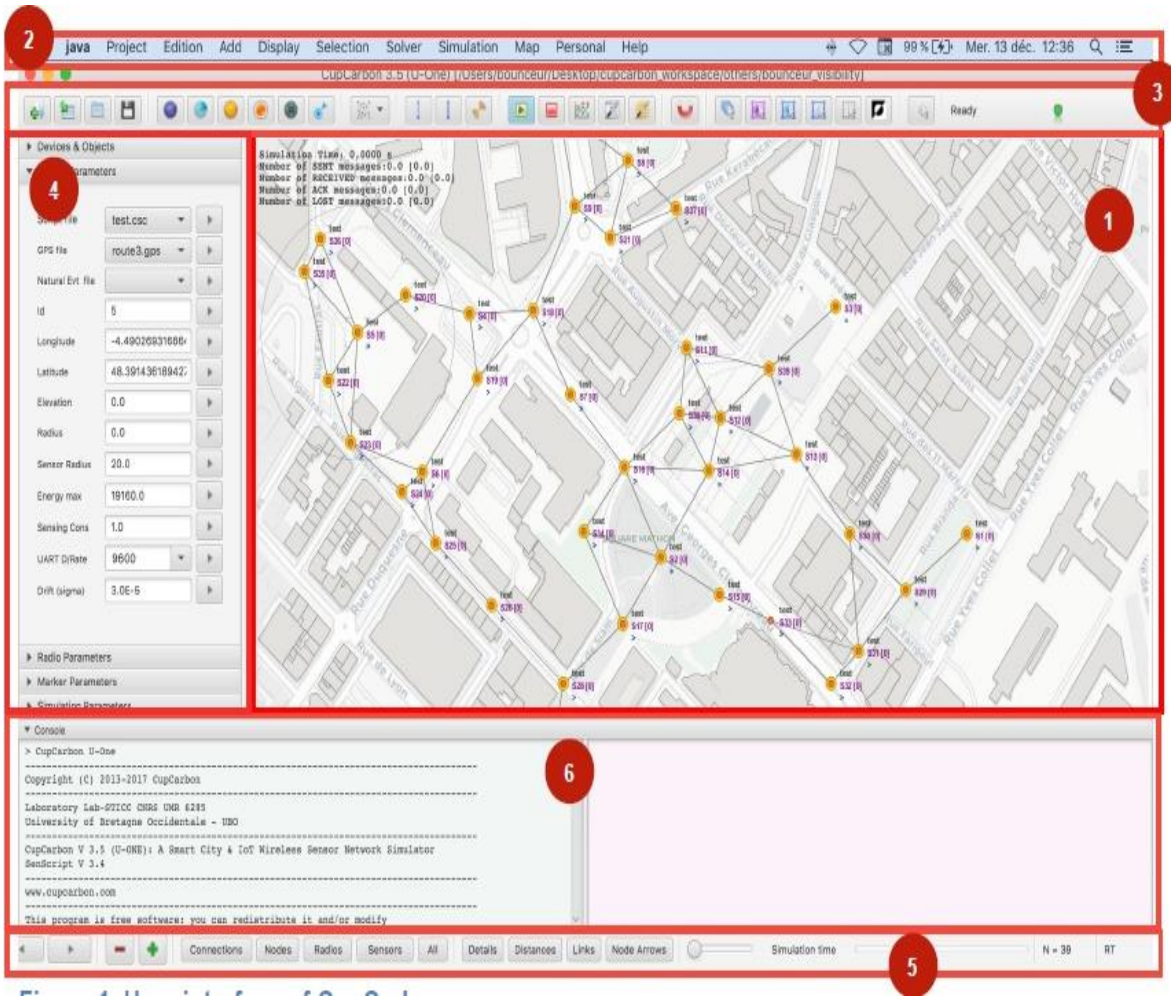



Рисунок 3.1 - Пользовательский интерфейс CupCarbon

Карта является основным объектом симулятора CupCarbon. Это та часть, где сеть и объекты проекта могут быть разработаны. Карта может быть изменена в соответствии с предпочтениями пользователя или для изменения способа представления информации. Пользователю можно выбрать следующие карты:

- светлый OpenStreetMap;
- темный OpenStreetMap;

- штриховой фон;
- мелкоклетчатый белый фон;
- мелкоклетчатый черный фон;
- черный фон;
- белый фон;
- серый фон;
- фон блокнота;
- синий фон;
- google карты;
- google карты (спутник).

Время моделирования отображается в левой верхней части карты. Во время моделирования это время отображается в красном цвете и вокруг карты рисуется дополнительный красный прямоугольник для обнаружения процесса моделирования в соответствии с рисунком 3.2. В этой части также отображается другая информация о сообщениях, т. е. Номер отправленных, полученных, и потерянных сообщений. Эту деталь можно скрыть и отобразить с помощью клавиш ALT+D.

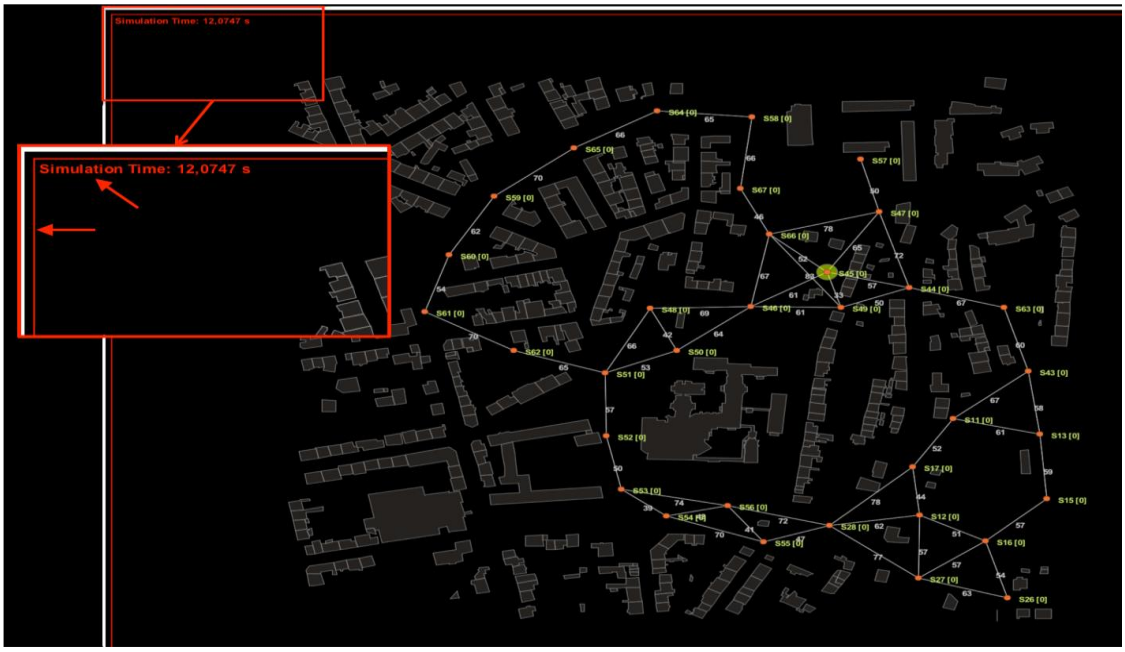



Рисунок 3.2 - Отображение информации на карте во время симуляции

Строка меню.

В соответствии с рисунком 3.3, строка меню состоит из 10 элементов:


Рисунок 3.3 - Строка меню CupCarbon

1. Проект;
2. Редактировать;
3. Добавить;
4. Показать;
5. Выбор;
6. Решающее устройство;
7. Моделирование;
8. Карта;
9. Личное;
10. Помощь.

Далее приводится подробная информация по каждому пункту:

Меню проекта.

Меню проект содержит следующие подпункты:

- Новый проект: он позволяет создать новый проект, начинающийся с полностью пустой карты. Будьте осторожны, если какой-либо проект был уже был запущен и разработан ранее, это удаление всей существующей работы. Чтобы избежать этого, необходимо выбрать элемент "Создать проект из текущего";
- Новый проект из текущего: он позволяет создание нового проекта с учетом существующие работы. Этот параметр можно использовать для дублирования проектов;
- Открыть проект: используется для открытия существующих проектов;
- Открыть последний проект: он позволяет открыть последний открытый проект;
- Последние проекты: он позволяет открыть один из последних 5 открытых проектов;
- Сброс: он сбрасывает весь проект;
- Если проект уже существует и не сохранен, этот параметр полностью удалит проект;
- Выход: он позволяет закрыть и выйти из проекта.

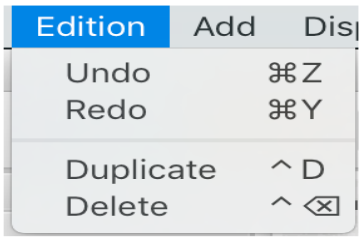



Рисунок 3.4 - Меню редактирования

Меню редактирования.

Меню «Редактирование» в соответствии с рисунком 3.4 содержит следующие подпункты:

1. Отменить: отменить действие

2. Повторить: пересмотреть отмененное действие
3. Дублировать: дублировать выбранные объекты на карте
4. Удалить: удалить любой выбранный объект на карте

*Меню «Добавить»*

Меню «Добавить» в соответствии с рисунком 3.5 позволяет добавлять объекты на карту.

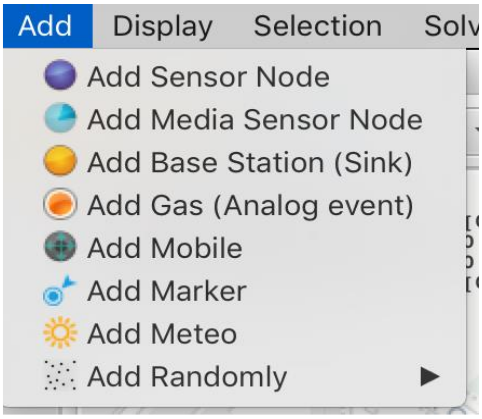



Рисунок 3.5 - Меню добавить

1. Добавить сенсорные узлы в соответствии с рисунком 3.6:

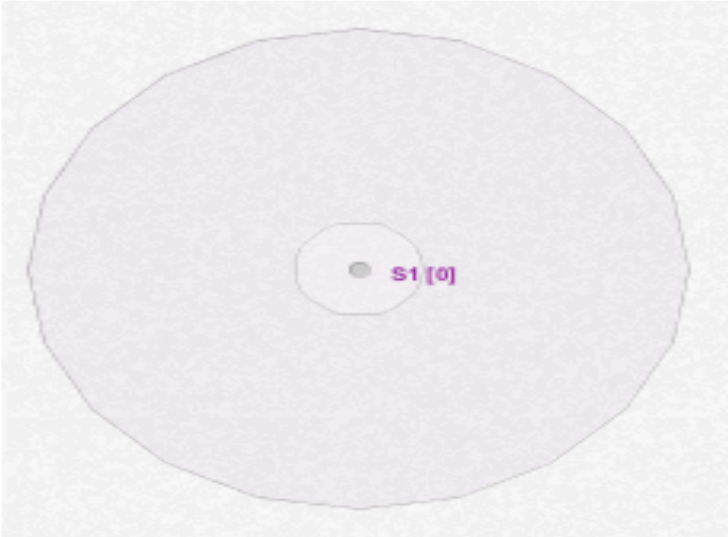



Рисунок 3.6 - Сенсорный узел

Сенсорный узел - это объект, который может обнаружить любое цифровое событие (событие движения, например, мобильные телефоны), отправлять и получать данные. Это может быть также мобильным. Видимые параметры сенсорного узла являются: диапазон радиосвязи, радио блока датчика и название. Сенсорный узел имеет много параметров; он может содержать много радио модулей, батарею и сенсорный блок.

2. Добавьте узлы датчика носителя в соответствии с рисунком 3.7.

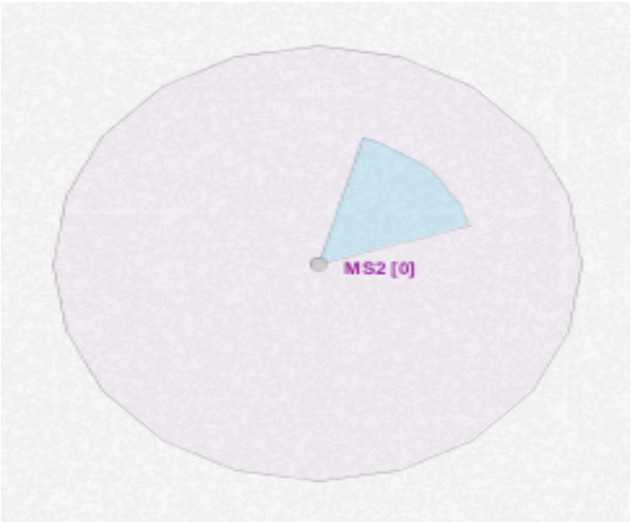



Рисунок 3.7 - Узел датчика носителя

3. Добавьте базовые станции в соответствии с рисунком 3.8.


Рисунок 3.8 - Базовая станция

4. Добавить газы в соответствии с рисунком 3.9

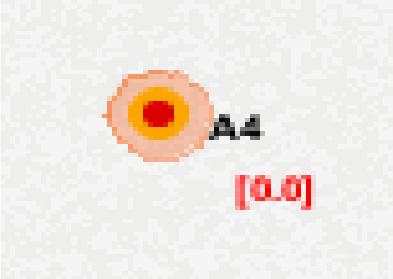



Рисунок 3.9 - Добавить газы

Он используется для генерации аналоговых событий для имитации параметров среды, таких как температура, влажность, газы и т. д. Этот объект требует использования генератора природных событий окно для генерации файлов с нужными значениями.

5. Добавить мобильные в соответствии с рисунком 3.10.

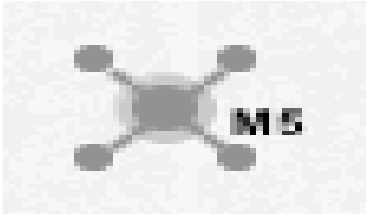



Рисунок 3.10 - Мобильная точка

Он используется для имитации мобильных телефонов. Маркеры также используются для создания маршрутов, сопровождаемых мобильных телефонов. У каждого мобильного телефона должен быть свой маршрут. Они также используются для генерации цифровых событий.

6. Добавить маркеры в соответствии с рисунком 3.11:

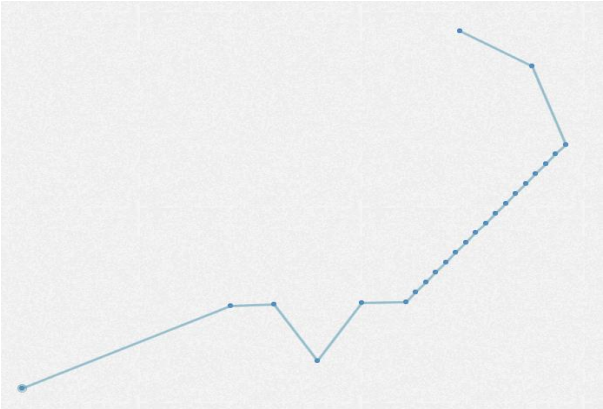


Рисунок 3.11 - Добавить маркеры

Маркер используется в основном для создания маршрутов для мобильных телефонов (или мобильных датчиков). Они также используются чтобы генерировать сенсорные узлы, создавать новые здания и указывать площадь генерирующих зданий или случайные сенсорные узлы.

7. Добавить Метеоузел в соответствии с рисунком 3.12.

Метеоузел (погода) используется для добавления переменной температуры, связанной с окружающей обстановкой.

Рисунок 3.12 - Метеоузел

Это полезно для моделей с потребление батареи, которые зависят от погоды. В проект можно добавить только один погодный узел.

8. Добавить случайно выбранные сенсорные узлы в соответствии с рисунком 3.13.

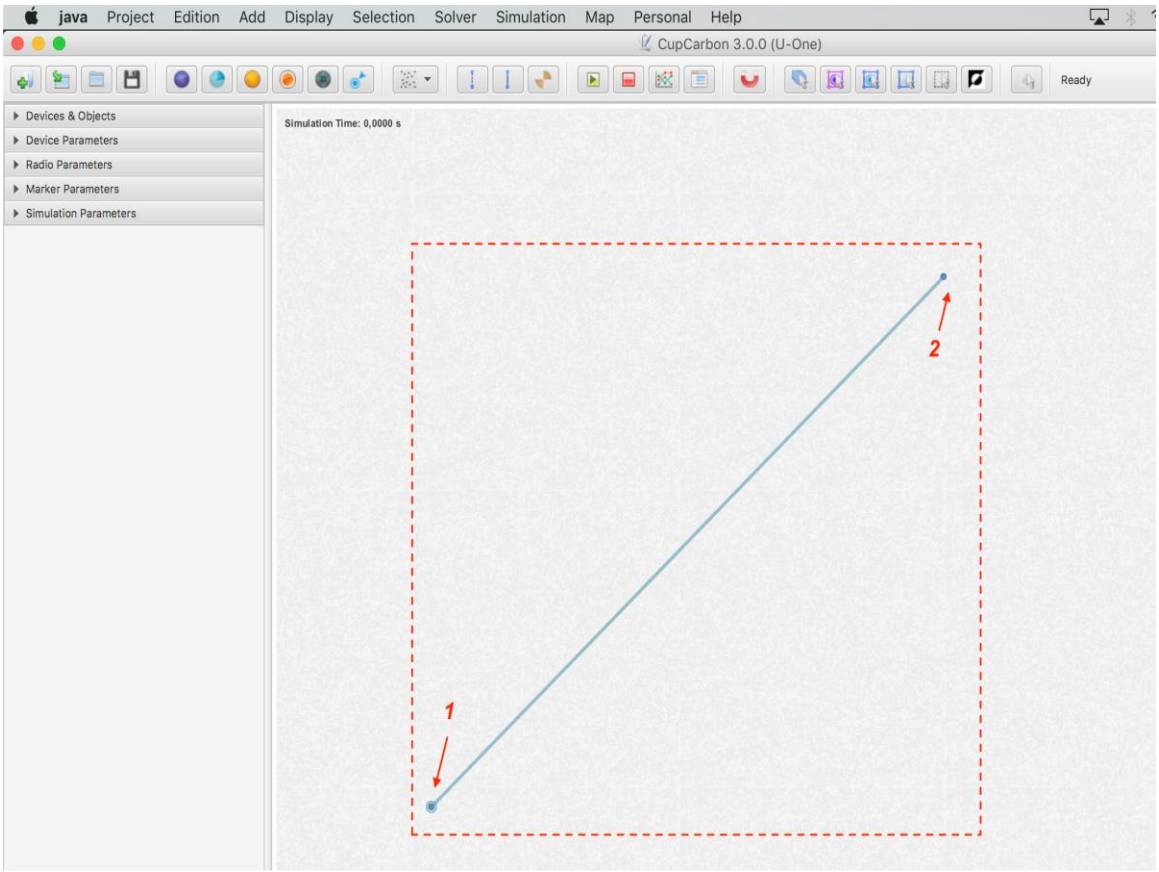



Рисунок 3.13 - Добавить случайно выбранные сенсорные узлы

Это позволяет случайным образом добавить определенное количество сенсорных узлов в выбранную область. Чтобы использовать эту опцию, сначала необходимо добавить два маркера. Первый маркер должен быть в нижнем левый углу карты и второй в верхнем левом углу карты, как показано выше. После добавления двух маркеров щелкните количество узлов датчика, которые необходимо добавить. В соответствии с рисунками 3.14 и 3.15 показаны примеры добавления случайным

образом 100 узлов датчиков.

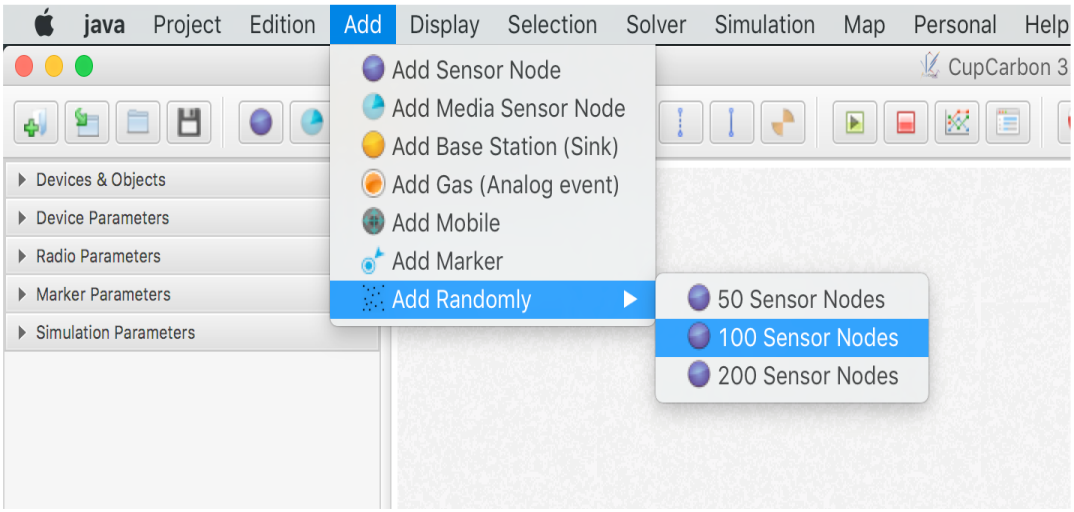



Рисунок 3.14 - Выбор количества случайных датчиков

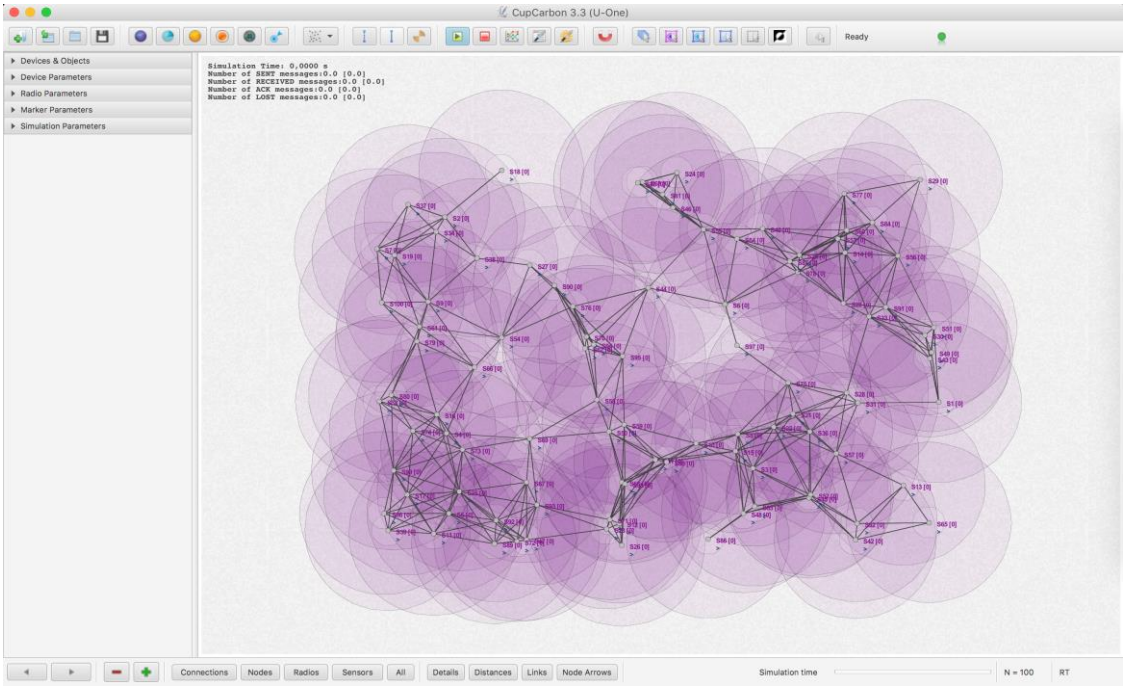



Рисунок 3.15 - Добавление случайных датчиков

Меню дисплея.

Меню дисплея в соответствии с рисунком 3.16 содержит следующие подпункты:

1. Показать/Скрыть детали: показать/скрыть имена выбранных узлов датчиков

2. Показать/Скрыть Имя файла: показать/скрыть назначенный SenScript файлов имя выбранного датчика узла

3. Показать/Скрыть уровни батареи/Buffer: показать/скрыть батарею и уровни буферных выбранные сенсорные узлы.

4. Показать/Скрыть Радио Расстояние: показать/скрыть расстояния между сообщающимся датчиками.

5. Показать/Скрыть радио Сообщения: показать/скрыть сообщения отправляются. Если ACK и ACK Show активизируются, эти сообщения будут предшествовать количество попыток передачи.

6. Показать/Скрыть Маркер Дистанция: показать/скрыть расстояния между маркерами.

2. Показать/Скрыть ссылки: показать/скрыть ссылки между взаимодействующими узлами датчиков.

3. Показать/Скрыть сеть Стрелка: показать/скрыть стрелку связи между узлами связи датчика.

4. Показать/Скрыть маркеры Стрелки: показать/скрыть стрелки между маркерами.

5. Показать/Скрыть здание: показать/скрыть загруженные/ созданные здания.

6. Следующей Ссылка Цвет и Предыдущее Ссылка Цвет: изменить цвет связей между узлами связи датчика.

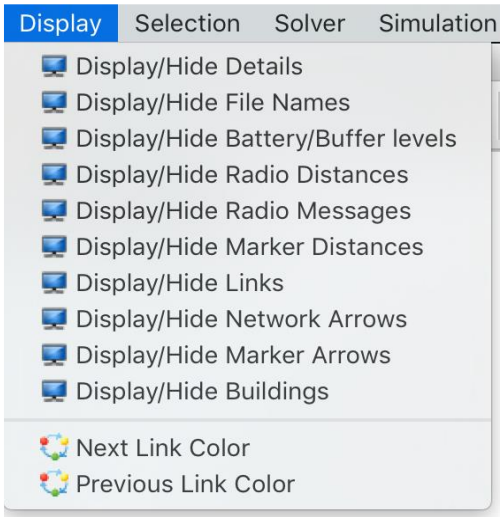



Рисунок 3.16 - Меню дисплея

Меню выбора.

Меню выбора в соответствии с рисунком 3.17 содержит следующие подпункты:

1. Выбрать все
2. Отменить все
3. Наоборот
4. Выберите датчики без скрипта
5. Выберите датчики без GPS
6. Выберите маркер датчики
7. Добавить выбор
8. Выбрать все ... и Отменить все ...

Другие варианты выбора доступны в левом меню в части устройств и объектов (вкладка: Выбор).

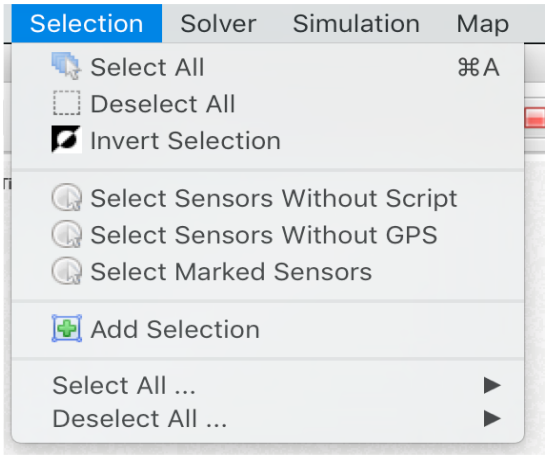



Рисунок 3.17 - Меню выбора

Кнопка «Добавить Выбор» позволяет добавить выбор к ранее выбранным объектам.

Меню решающего устройства.

Меню решающего устройства в соответствии с рисунком 3.18 содержит следующие подпункты:

1. Алгоритм покрытия датчика;
2. Алгоритм покрытия цели;
3. Целевой охват (м) алгоритма;
4. Планирование;
5. Сетевой центр;
6. Корпус.

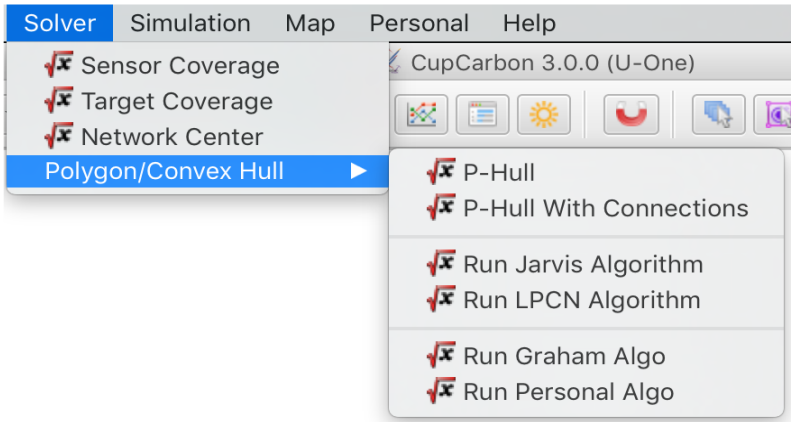



Рисунок 3.18 - Меню решающего устройства

Меню моделирования.

Меню моделирования в соответствии с рисунком 3.19 содержит следующие подразделы:

1. Запустить моделирование: для запуска моделирования.

2. Остановить моделирование: чтобы остановить моделирование.

3. Потребление энергии: для отображения графика потребления энергии. для выбранных узлов датчиков после завершения моделирования. Сначала нужно становить флажок - результаты перед запуском моделирования. Два вида графики возможны. Первый показывает состояние батареи по отношению к времени моделирования, а другой показывает потребление датчика по отношению к времени моделирования.

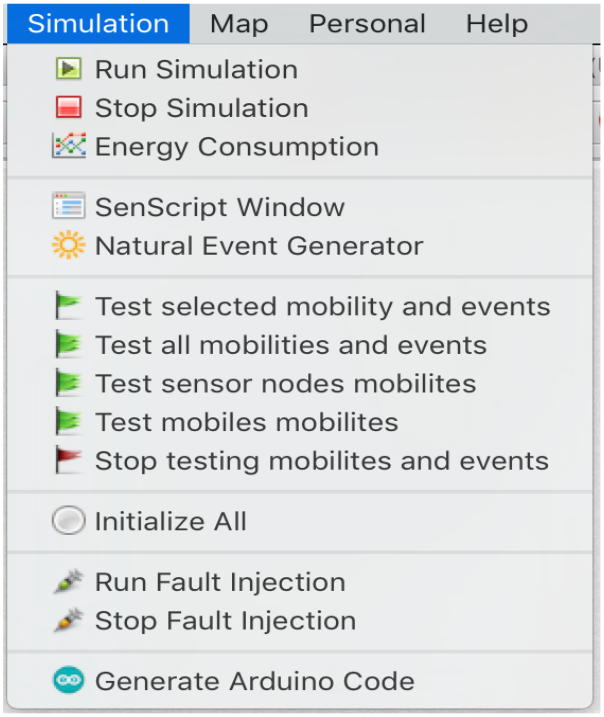



Рисунок 3.19 - Меню моделирования

4. Окно скрипта в соответствии с рисунком 3.20.

5. Генератор естественных событий: для генерации значений естественных событий. Этот генератор позволяет генерировать значения, которые могут быть обнаружены узлами датчика, например, температура, влажность, газ/загрязнение. Его можно использовать также для того чтобы произвести температуры погоды в соответствии с рисунком 3.21.

6. Можно скопировать непосредственно в текстовую область реальный файл данных или сгенерированный с помощью другого инструмента.

7. В первом столбце указано время в секундах, необходимое для создания следующего значения. Тестирование выбранной мобильности и событий: для имитации и проверки мобильности выбранного мобильного устройства, датчика узел и события (газ).

8. Проверка всех подвижностей и событий: для имитации подвижности и события всех устройств (мобильных, сенсорных узел,

газы).

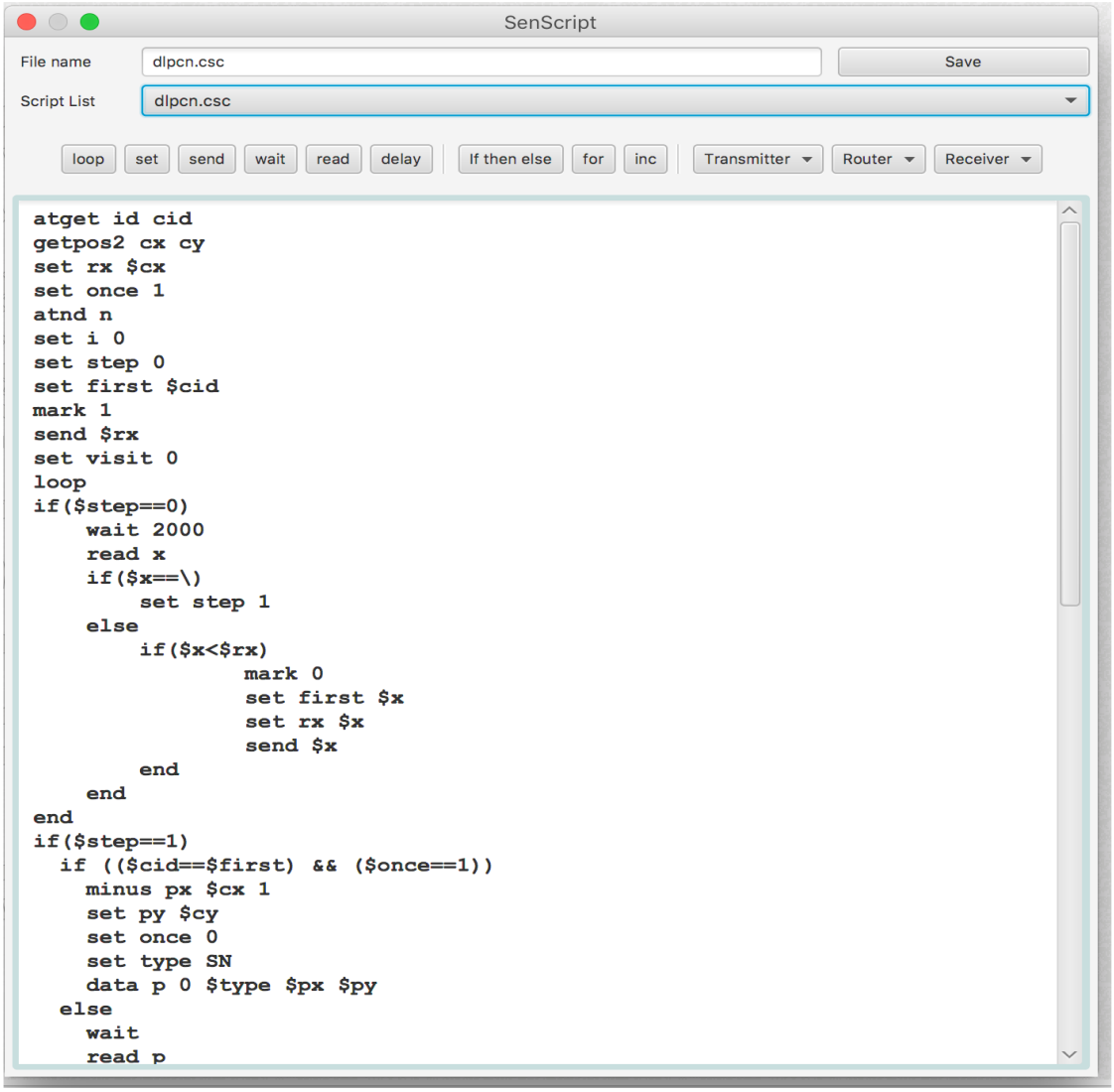



Рисунок 3.20 - Окно скрипта

1. Проверка подвижности узлов датчиков: для имитации подвижности всех датчиков.

2. Тест мобильных телефонов мобильность: для имитации мобильности всех мобильных телефонов.

3. Остановка тестирования подвижности и событий: остановка моделирования подвижности и событий.

4. Инициализировать все: для инициализации среды моделирования и информации, отображаемой во время моделирования, как и стрелки отправки сообщений.

5. Запуск проверки неисправностей в соответствии с рисунком 3.22: для случайного генерирования неисправных узлов датчиков (во время моделирования или нет). Дефектный датчики также можно активировать, выбрав датчик и нажав на клавишу "k".

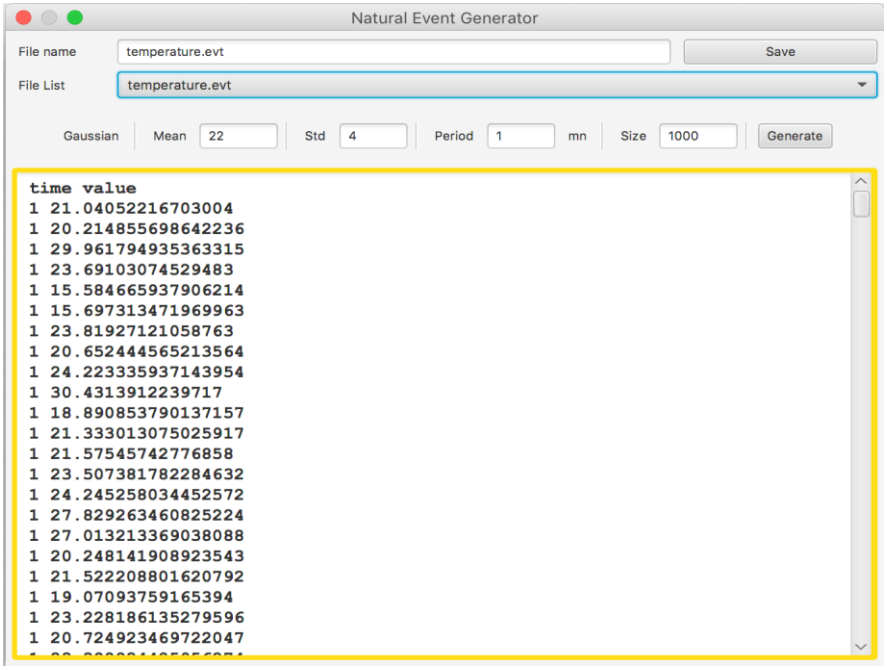



Рисунок 3.21 - Генератор естественных событий

Остановите впрыску недостатка: остановить процесс, производить небезупречные узлы датчика.

9. Генерировать код Arduino: для автоматического генерирования соответствующего кода Arduino каждого датчика узла из его кода скрипта. Файл параметров технологии должен быть добавлен в каталог технологий.

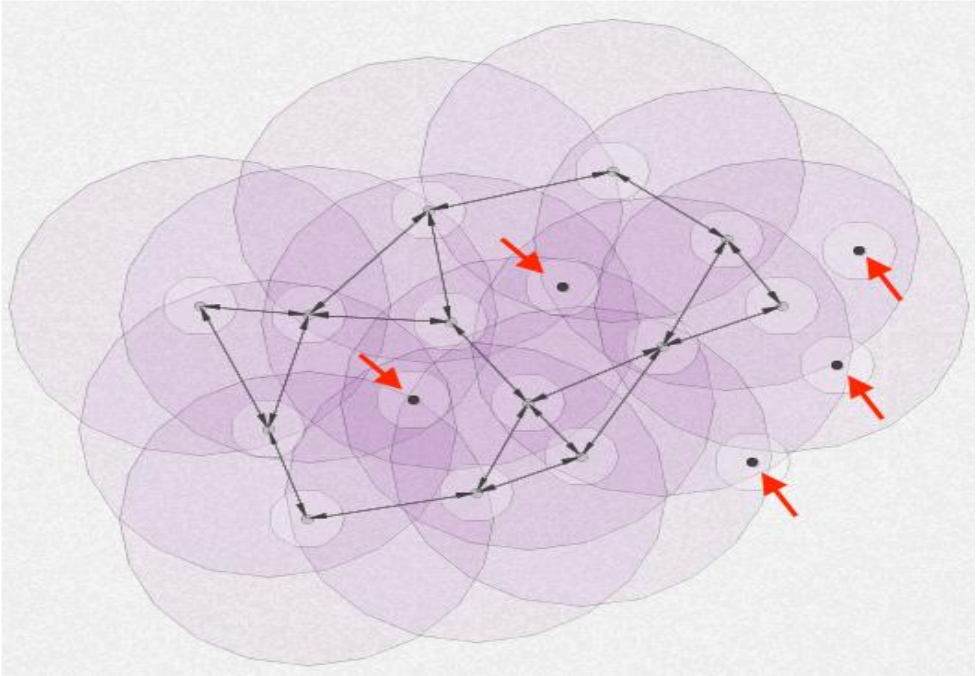


Рисунок 3.22 - Запуск неисправностей

Меню карты.

Меню карты позволяет выбрать предпочтительную карту и окружающую среду. Представлены все доступные карты начале этого раздела в соответствии с рисунком 3.23.

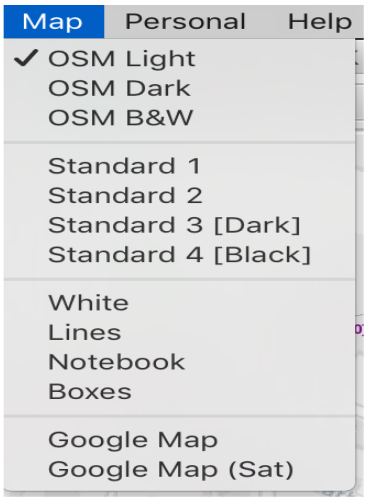



Рисунок 3.23 - Меню карты

Персональное меню.

Персональное меню используется для создания личных функций скрипта и для использования программирования VIZOR в соответствии с рисунком 3.24.

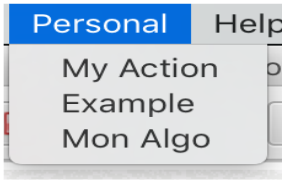



Рисунок 3.24 - Персональное меню

Меню справки.

Меню Справка содержит ссылку Справка и поле о программе в соответствии с рисунком 3.25.

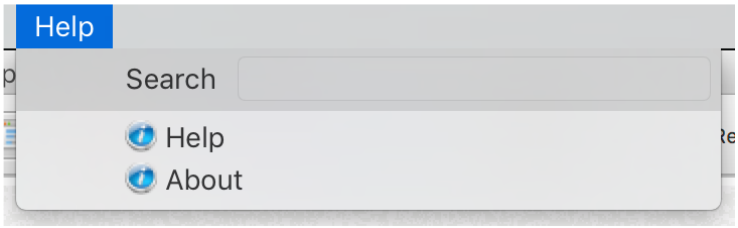



Рисунок 3.25 - Меню справка

Панель инструментов.

Панель инструментов используется для доступа к основным действиям CupCarbon в соответствии с рисунком 3.26.

Рисунок 3.26 - Панель инструментов

Он состоит из 7 частей, которые являются:

Часть проекта в соответствии с рисунком 3.27.

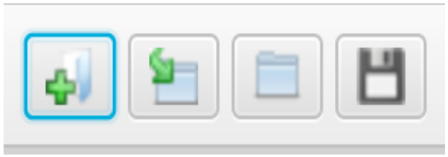


Рисунок 3.27 - Часть проекта

Он позволяет создать новый проект, открыть последний проект, открыть проект и сохранить проект (ср. Меню "Файл").

Часть добавления объектов в соответствии с рисунком 3.28.

Рисунок 3.28 - Часть добавления объектов

Он позволяет добавлять объекты (узел датчика, узел датчика направления, базовая станция, имеет, погода, мобильный, маркер, случайные сенсорные узлы) на карте.

Соединительная часть в соответствии с рисунком 3.29.

Рисунок 3.29 - Соединительная часть

Это позволяет рисовать нормальные или основанные на радиопередаче соединения между узлами датчика. Это позволяет также вычислить видимость радиуса сенсорного узла, рассматривая здания

города.

Часть моделирования в соответствии с рисунком 3.30.

Рисунок 3.30 - Часть моделирования

Он позволяет запустить моделирование, остановить моделирование, нарисовать функцию энергопотребления, открыть и увидеть окно скрипта и открыть генератор естественных событий. Эти параметры описаны выше в меню.

Часть магнетизма в соответствии с рисунком 3.31

Рисунок 3.31 - Часть магнетизма

В соответствии с рисунком 3.32 он позволяет добавлять объекты в невидимую сетку. Рекомендуется использовать карту (средний серый фон ячейки), когда используется этот параметр.

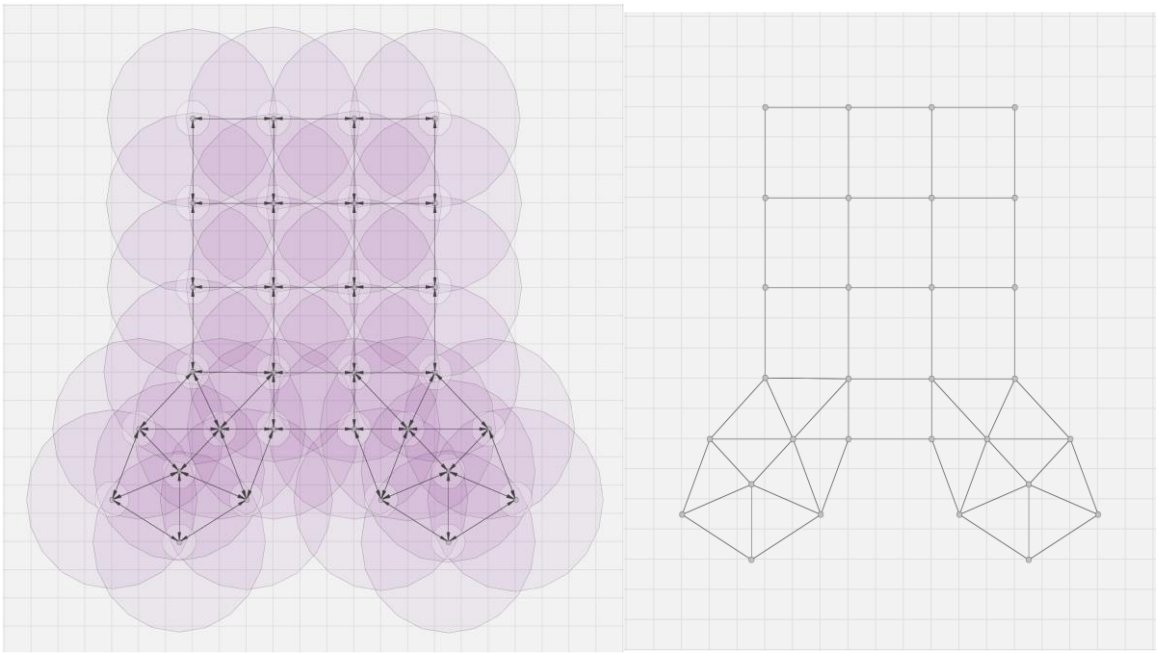


Рисунок 3.32 - Добавление объектов в невидимую сетку

Выборочная часть в соответствии с рисунком 3.33.

Рисунок 3.33 - Выборочная часть

Он позволяет выбрать все, выбрать все сенсорные узлы, выбрать все маркеры, выбрать сенсорные узлы/маркеры, отменить выбор всех и инвертировать выделение (ср. меню выбора.)

Строка состояния.

Строка состояния находится в нижней части главного окна CupCarbon. Это позволяет отображать различную информацию и содержит кнопки, чтобы сделать некоторые опции, как в случае с панелью инструментов в соответствии с рисунком 3.34.


Рисунок 3.34 - Строка состояния

Другая часть строки состояния будет представлена в следующем:

Отображение/скрытие панели параметров слева в соответствии с рисунком 3.35:

Рисунок 3.35 - Отображение/скрытие панели параметров

Масштабирование карты: номер увеличения указан внутри круга справа в соответствии с рисунком 3.36.


Рисунок 3.36 - Масштабирование карты

Опции визуализации сети в соответствии с рисунком 3.37:

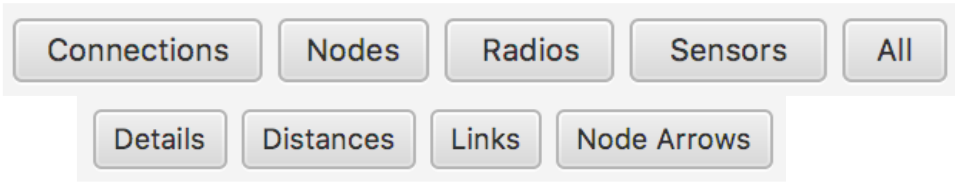



Рисунок 3.37 - Опции визуализации сети

- соединения: показывает только узлы датчиков с их соединениями;
- узлы: показывает только узлы датчика без соединений;
- радио: показывает только сенсорные узлы с их дальностью радиосвязи и без соединений;
- датчики: показывает только узлы датчиков с их сенсорными блоками и без соединений;
- все: показывает все части узлов датчика;
- детали: ср. показать/скрыть детали;
- расстояния: ср. показать/скрыть дистанции радио;
- час ссылки: ср. показать/скрыть дистанции радио;
- узлы стрелки: ср. показать/скрыть сетевые стрелки.

Эти параметры могут быть объединены между ними. Например, в случае, когда мы не хотим показывать сенсорные блоки, что означает, что мы хотим отображать сенсорные узлы только с их радиодиапазонами и подключений, затем мы нажимаем на кнопку радио, а затем на кнопку ссылки.

Контрастность карты в соответствии с рисунком 3.38:

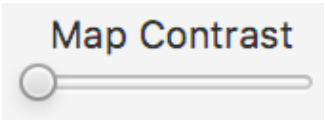



Рисунок 3.38 - Контрастность карты

Этот ползунок используется для изменения контрастности карты для улучшения видимость сети.

Симуляция прогрессии времени: следующая шкала прогрессирования позволяет визуализировать прогрессию. Время симуляции или прогрессирование других операций в соответствии с рисунком 3.39.

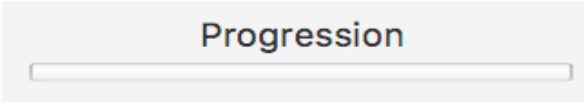



Рисунок 3.39 - Симуляция прогрессии времени

15. Количество сенсорных узлов на карте: следующий текст

показывает количество добавленных сенсорных узлов на карте в соответствии с рисунком 3.40.

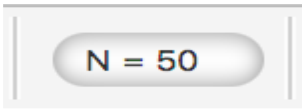



Рисунок 3.40 - Количество сенсорных узлов на карте

Реальное время симуляции (RT): после завершения симуляции в следующем текстовом представлении реальная продолжительность в секундах в соответствии с рисунком 3.41.


Рисунок 3.41 - Реальное время симуляции

Консоль.

Консоль используется симулятором для отображения некоторых сообщений, полезных для пользователя во время симуляции. Имеет две части. Первый слева используется для отображения сообщений о симуляции. Можно также отображать сообщения датчиков с помощью команды скрипта. Второй справа используется для отображения ошибок во время моделирования в соответствии с рисунком 3.42.

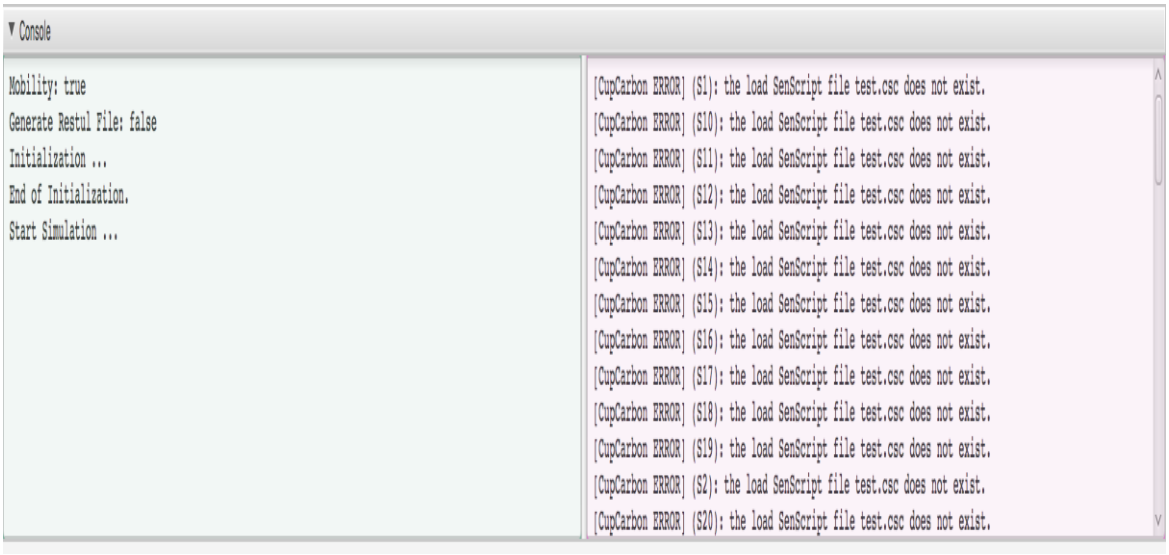



Рисунок 3.42 - Консоль

# 4 Управление и настройка возможных условий на CupCarbon

В этом разделе представлены все возможные объекты, которые могут быть добавлены на карту, и способы управления ими.

Сенсорный Узел.

Сенсорный узел в соответствии с рисунком 4.1 является главным объектом CupCarbon. Он содержит четыре основные части: радио модули, датчик, блок и батарея. Узел датчика можно добавить, щелкнув меню Add a AddSensorNode, а затем нажав на карту, в том месте, где должен быть добавлен узел датчика. Еще один последовательный щелчок мыши добавит еще один узел датчика, и так далее. Чтобы прекратить добавление сенсорных узлов, просто нажмите на правую кнопку мыши, или введя клавишу escape [esc]на клавиатуре. Мы также можем нажать на кнопку  в панели инструментов. Другой способ добавить сенсорные узлы - это ввести клавишу 1 на клавиатуре, а затем нажать на кнопку мыши, как объяснялось ранее. Мы также можем нажать на кнопку  в панели инструментов или использовать маркеры.

В центре сенсорного узла мы находим имя S, за которым следует его идентификатор. Например, если его идентификатор равен 4, тогда его имя будет S4. В правой части имени мы находим число, расположенное между скобками, которое является по умолчанию равно [0]. Этот номер представляет мой адрес этого узла датчика. Если скрипт является назначенный ему, то он будет отображаться в сером цвете над его именем. Отобразятся сообщения печати в синем под его названием.

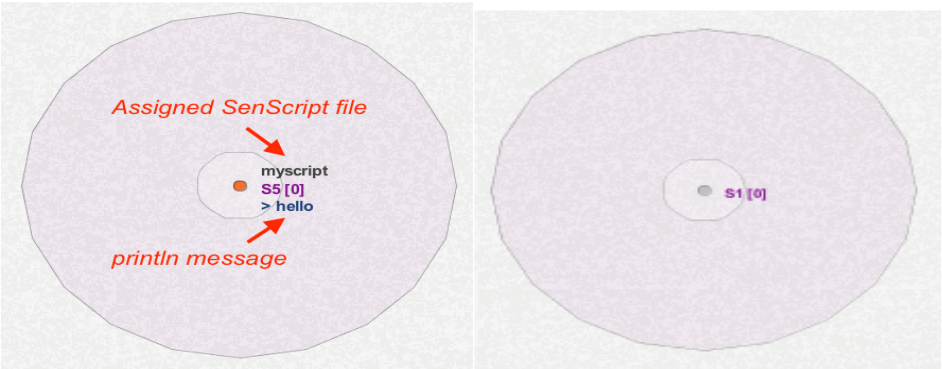



Рисунок 4.1 - Сенсорный узел

Узел датчика может содержать множество радио модулей с различными или одинаковыми стандартами (802.15.4, WiFi или Лора.) Однако для связи рассматривается только один радио модуль. Узел датчика в соответствии с рисунком 4.2 содержит чувствительный элемент, представленный прозрачным белым кругом. Радиус области может можно изменить с помощью кнопок’) ' для увеличения радиуса и ‘

(‘для уменьшения радиуса.

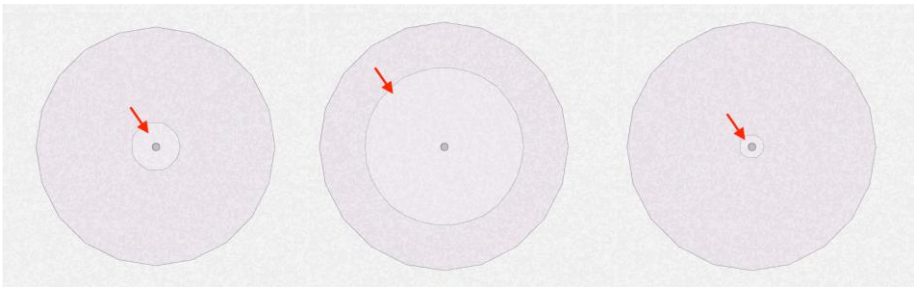


Рисунок 4.2 - Узел датчика

Другие клавиши клавиатуры могут использоваться для изменения некоторых параметров сенсорного узла, например, ‘d', чтобы показать/скрыть название датчика узла, ‘D’, чтобы показать/скрыть сообщение, отображаемое датчиком узел, ‘N’, чтобы Показать/Скрыть Назначенное имя файла Enscript. Клавиша 'b' показывает/скрывает уровни заряда батареи/буфера. Ключ "здесь" позволяет скрыть некоторые части сенсорного узла. Набрав много раз мы скрываем в каждый раз следующее части:

1. Первый раз: все части за исключением центра.
2. Второй раз: показывает только круг радиуса действия текущего радио модуля.
3. Третий раз: показывает только радиус действия текущего радио модуля и сенсорного блока.
4. Четвертый раз: показывает только чувствительный блок.
5. Пятый раз: показывает только область радиуса действия, текущего радио модуля.
6. Шестой раз: возвращаемся к первоначальному чертежу сенсорного узла (со всеми деталями).

Направленный узел датчика в соответствии с рисунком 4.3.


Рисунок 4.3 - Направленный узел датчика

Направленный узел датчика такой же, как и классический узел

датчика с другим типом чувствительного блока, что является направленным. Этот последний не является круглым, он имеет форму конуса, который может быть изменен с помощью SenScript и вручную с помощью следующих клавиш клавиатуры:

1. ‘)' и ‘(‘: увеличьте или уменьшите дальность действия датчика (расстояние). Используйте ‘]’ и ‘[‘ для большей точности.

2. ‘p " и "o": поверните (влево или вправо) диапазон датчиков. Нажмите на значок для большей точности.

3. ‘P " и "O": увеличение или уменьшение площади (угла) чувствительного элемента. Нажмите на значок для большей точности.

Базовая станция.

Базовая станция является точно таким же объектом, как и узел датчика, за исключением того, что она имеет бесперебойное питание в соответствии с рисунком 4.4.


Рисунок 4.4 - Базовая станция

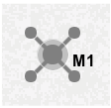
Газ. Газовое или природное событие позволяет генерировать аналоговые значения (значение отображается красным цветом) в соответствии с рисунком 4.5. Оно может быть мобильной.

Его цель-имитировать случайные или заданные значения, поступающие из окружающей среды. Используйте генератор естественных событий для генерации случайных величин на основе гауссовского распределения. Примите во внимание событие во время процесса моделирования, поле (мобильность/событие) должно быть активировано в поле Панель параметров моделирования.

Красный центр означает, что событие не является мобильным, иначе оно будет оранжевым цветом. Второй белый круг вокруг становится желтым, когда назначается файл естественного события. В течение моделирования, если событие является мобильным, то центр становится зеленым.


Рисунок 4.5 - Газ

``````Мобильный телефон.

Маршрут не назначен (серый центр) 


Назначенный маршрут (оранжевый центр) 


Мобильное моделирование в соответствии с рисунком 4.6 (Зеленый центр)

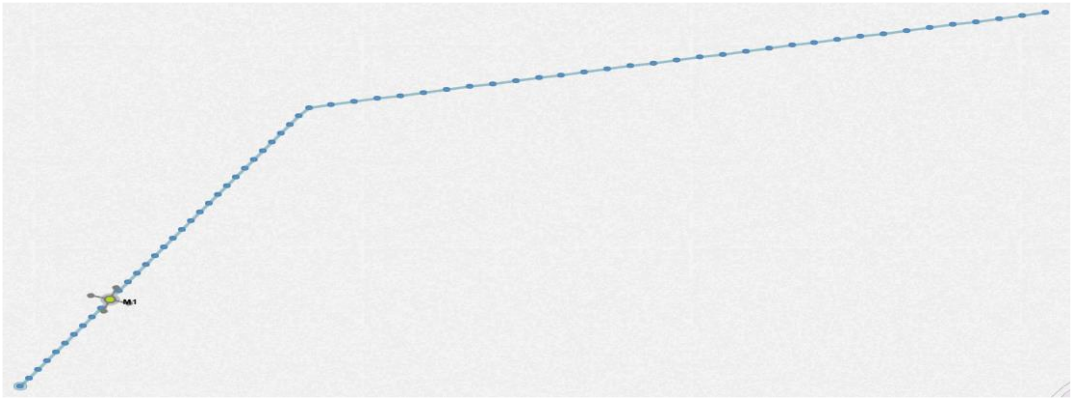


Рисунок 4.6 - Мобильное моделирование

Во время моделирования мобильный телефон переходит в каждую следующую точку (пункт назначения) через каждые 1 секунду. Принять во внимание учет мобильности в процессе моделирования необходимо активировать поле (мобильность/событие) в поле Просмотр параметров моделирования. Маркер в соответствии с рисунком 4.7.

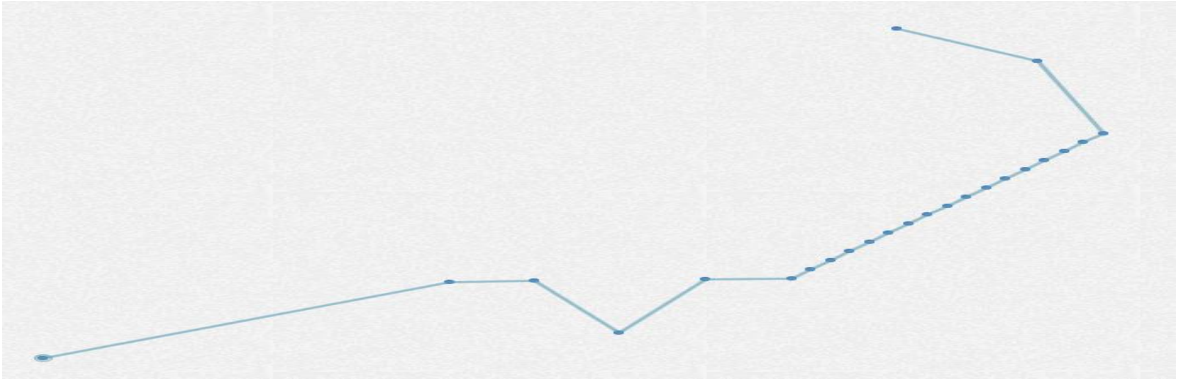


Рисунок 4.7 - Маркер
``````

Маркеры могут быть использованы для многих целей, которые:

— добавление случайных узлов датчиков;

— добавление узлов датчиков.

Возможно преобразование маркеров в сенсорные узлы. Это может помочь добавить узлы датчика, разделенные с помощью эквивалентные расстояния. Этот процесс объясняется следующим образом:

1. Добавьте 2 маркера. Затем выберите все эти маркеры (выберите все маркеры) и нажмите на клавишу 'u', чтобы вставить промежуточные маркеры. Это действие можно повторять много раз, пока не будет получено желаемое количество маркеров. Клавиша " U " может быть использована для выбора маркеров, расположенных после них. Для отображения расстояний между маркерами, нажмите клавишу ' X' (заглавная X);

2. Выберите все маркеры, снова и введите клавишу 't', чтобы преобразовать все выбранные маркеры в датчик узла;

3. Можно перемещать непосредственно маркеры, уже выбранные, чтобы добавить другие узлы датчика.

Генерация маршрутов. Можно создавать маршруты, либо рисуя маршрут вручную с помощью маркеров, либо просто создавая два маркера и нажмите на кнопку маршрут от маркеров в представлении параметров маркера. Затем каждый созданный маршрут может быть сохранен и добавлен в список маршрутов.

Добавление зданий. Список зданий может быть добавлен в область, разделенную двумя маркерами, как описано выше в маркере Раздел просмотра параметров.

Чертеж зданий в соответствии с рисунком 4.8. Можно нарисовать здание, нарисовав форму с помощью маркеров, а затем набрав на клавише ':'.

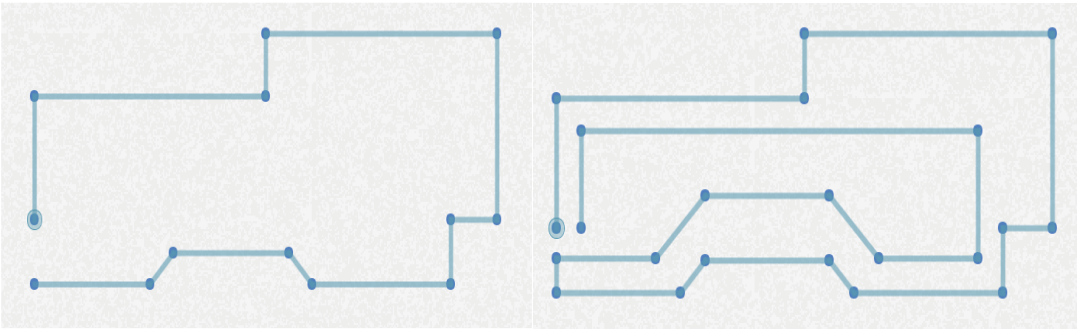


Рисунок 4.8 - Чертежи зданий

Погода. Модуль погоды используется для генерации любого параметра погоды и особенно температуры. Только один модуль погоды может быть добавлен в проект. В этой версии погода используется для моделирования изменений в соответствие с рисунком 4.9 температуры окружающей среды, которую можно учитывать в моделях

энергопотребления радиоприемника.

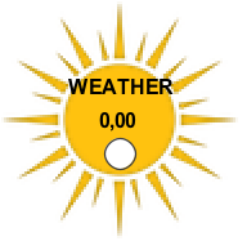



Рисунок 4.9 - Модуль погоды

Температуры могут быть получены с помощью генератора природных событий, представленного выше, и с использованием следующей кнопки в соответствии с рисунком 4.10:

Рисунок 4.10 - Кнопка погоды

Управление клавиатурой.

1. Клавиша 'a’: отображение/скрытие стрелок связей между соединяющимися сенсорными узлами;
2. Клавиша 'A’: Показать/Скрыть стрелки ссылок между маркерами;
3. Клавиша 'x’: Показать/Скрыть расстояния в соединениях между соединяющимися сенсорными узлами;
4. Клавиша 'X’: Показать/Скрыть расстояния в связях между маркерами;
5. Кнопка ‘W’: выделить все узлы датчика à выделить все маркеры à отменить все;
6. Клавиша 'l’: показать/скрыть связи между соединяющимися сенсорными узлами;
7. Клавиша 'v’: изменение цвета радиоканалов;
8. Клавиша 'd’: отображение/скрытие имени объектов и сообщений, отображаемых командой печати;
9. Клавиша 'n’: показать/скрыть имя файла объектов;
10. Клавиша 'r’: показать или скрыть некоторые параметры непосредственно на карте;
11. Клавиша 'R’: показать/скрыть все маршруты;
12. Клавиши направления: перемещение карты;
13. delete: удалить объекты.

# 5 Основы моделирования в CupCarbon

Рассмотрим несколько основных примеров, по использованию CupCarbon, для имитации различных сценариев.

Пример 1: HelloWorld

В этом примере показано, как датчик может отображать сообщение "HelloWorld". Следующие шаги показывают, как это сделать

*Шаг 1.*

Создать новый проект: это можно сделать, либо нажав на значок  "новый проект" на панели инструментов в соответствии с рисунком 5.1

панель инструментов или в меню проект → новый проект. Выберите имя (пример: hello world) и место, где вы хотите сохранить свой проект. Проект создаст новую папку с заданными данными проекта.

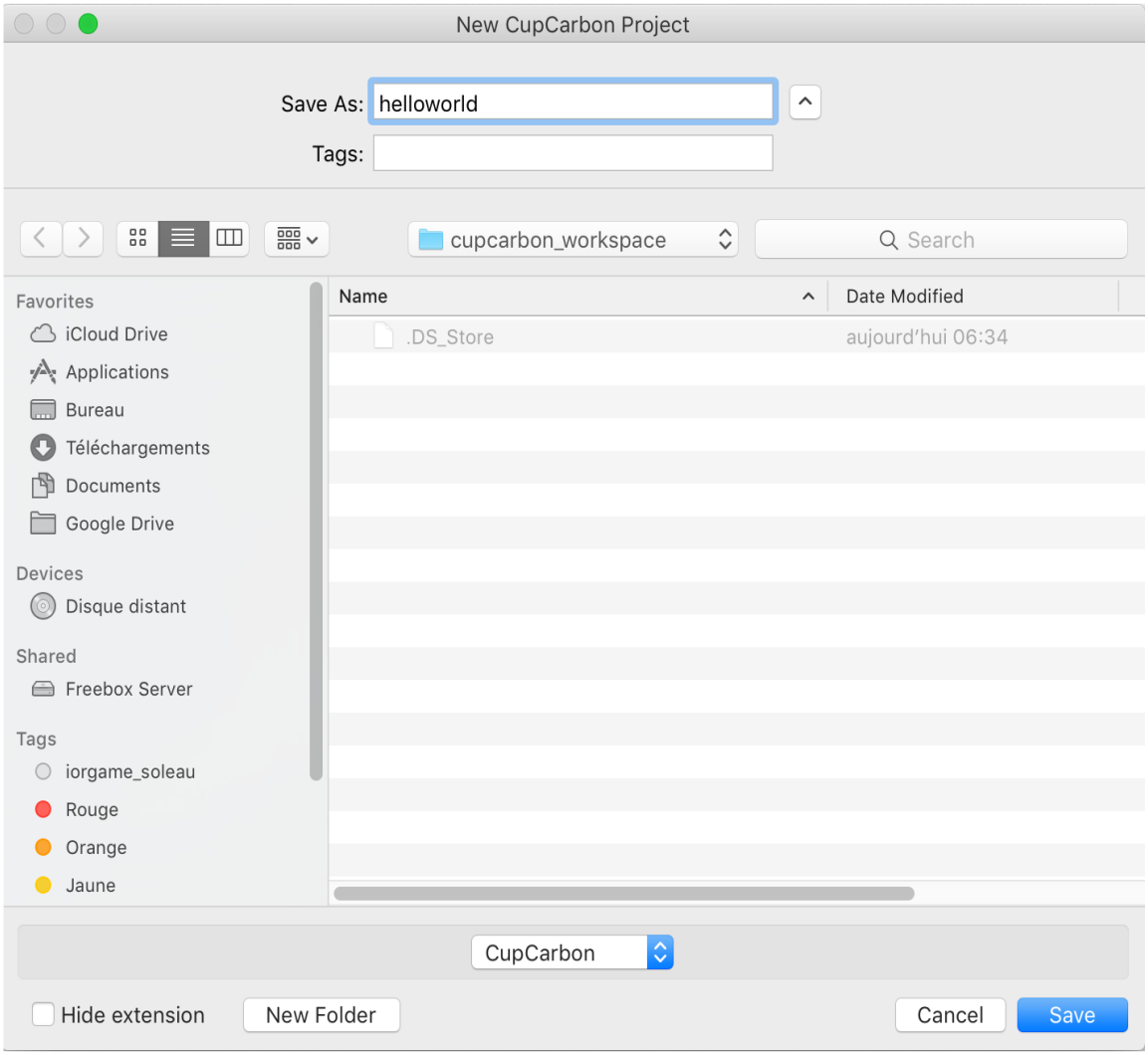



Рисунок 5.1 - Создание нового проекта

Внутри этой папки 1 файл (hello world.cup) и 8 других каталогов будут созданы. Содержание каждого из них в соответствии с рисунком 5.2 каталог приведен в следующем виде:

a. конфигурация: он содержит параметры моделирования файл, файл построение списка, маркер списка файлов и два других каталога (sensor и sensor_radios), которые содержат список узлов датчиков(один файл по узлу датчика) и список радиомодулей каждого датчика.

b. gps: он содержит список маршрутов;

c. журналы: он содержит файл журнала;

d. результаты: он содержит результаты моделирования (файл csv).

e. скрипты: он содержит файлы сценариев проекта;

f. netevents: содержит файлы естественных событий;

g. tmp, сеть: которая используется симулятором.

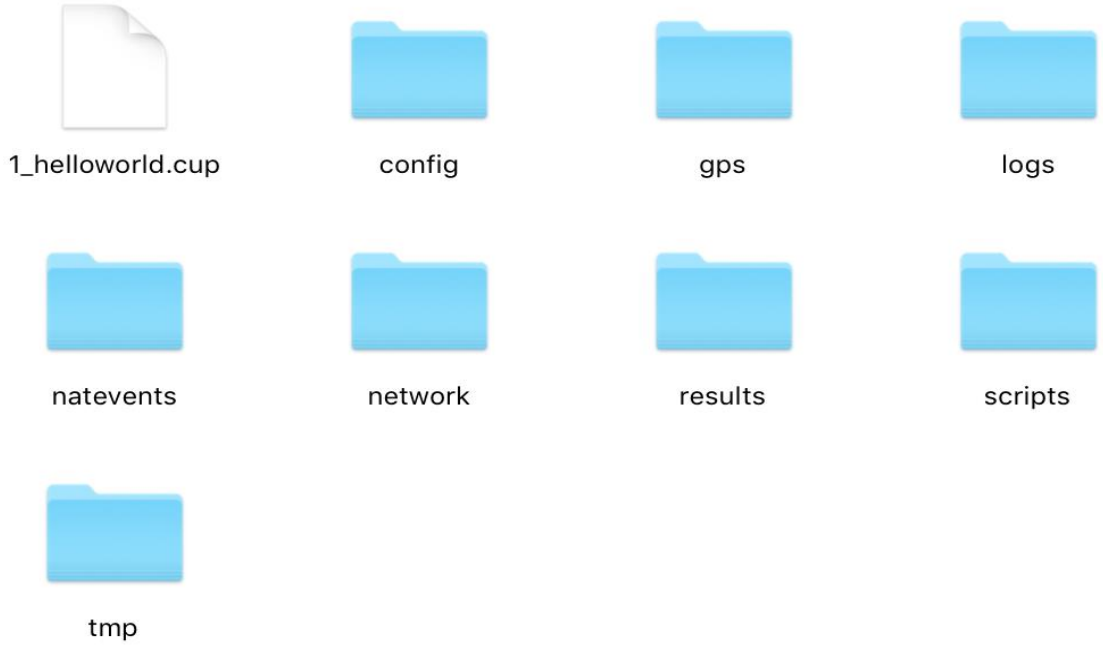



Рисунок 5.2 - Созданные каталоги

Шаг 2.

Добавление нового узла датчика на карте в соответствии с рисунком 5.3: щелкните значок добавить датчик на панели инструментов или в строке меню (Добавить → добавить узел датчика). Затем нажмите на карту, где вы хотите добавить сенсорный узел. Еще один щелчок приведет к другому новому узлу датчика и так далее. Чтобы остановить добавление датчика узлы, просто нажмите на правую кнопку мыши. Вы также можете нажать на значок панели инструментов, или набрав на клавиатуре клавишу escape [esc].

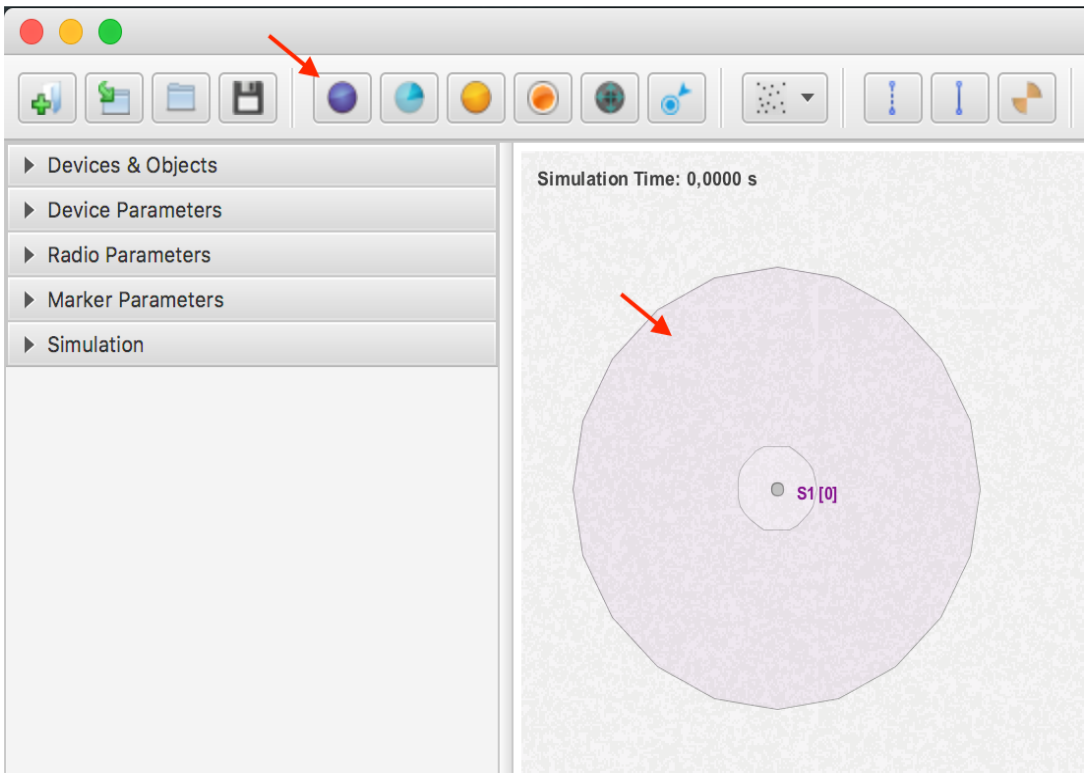



Рисунок 5.3 - Добавление нового узла датчика

Шаг 3.

Откройте окно скрипт в соответствии с рисунком 5.4: окно SunScript можно открыть, нажав на значок на панели инструментов или из меню Simulation → SunscriptWindow.

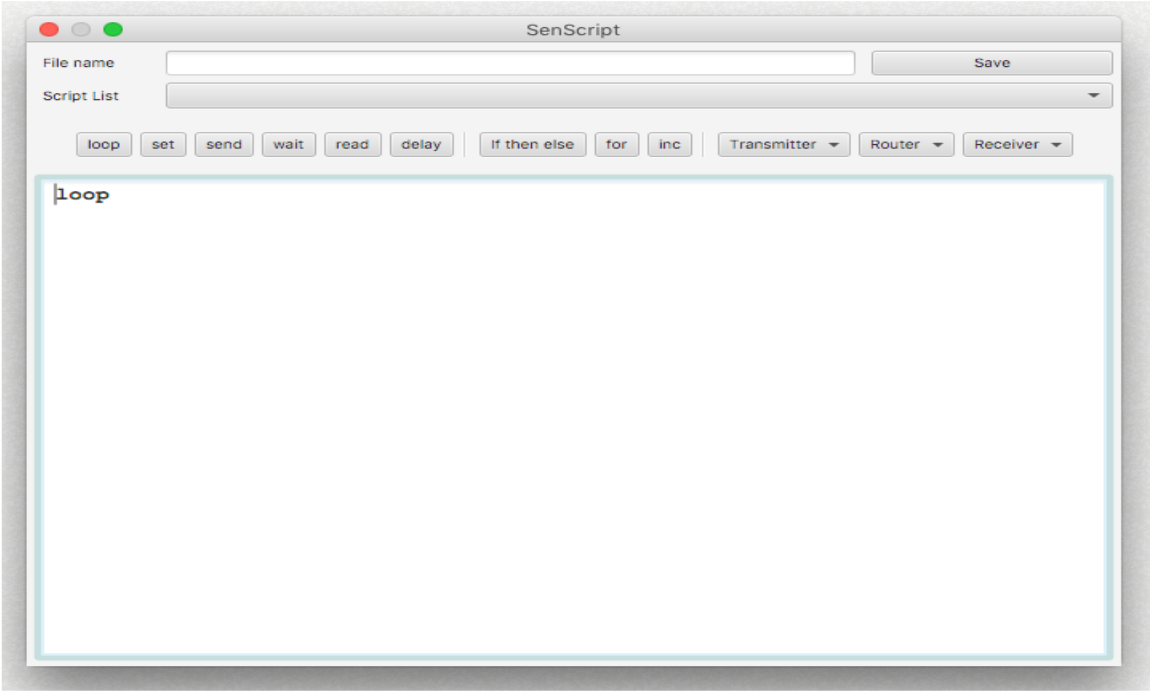



Рисунок 5.4 - Окно скрипта

Шаг 4.

Напишите скрипт: добавьте следующий скрипт в текстовую область окна SunScriptв соответствии с рисунком 5.5:

loop

printHelloWorld

stop

Добавьте Имя hello (1) этого скрипта в поле Имя файла (2), затем нажмите на кнопку Сохранить (3) только в поле Имя файла. Левая часть этого поля. Это создаст файл hello.csc в каталоге скриптов.

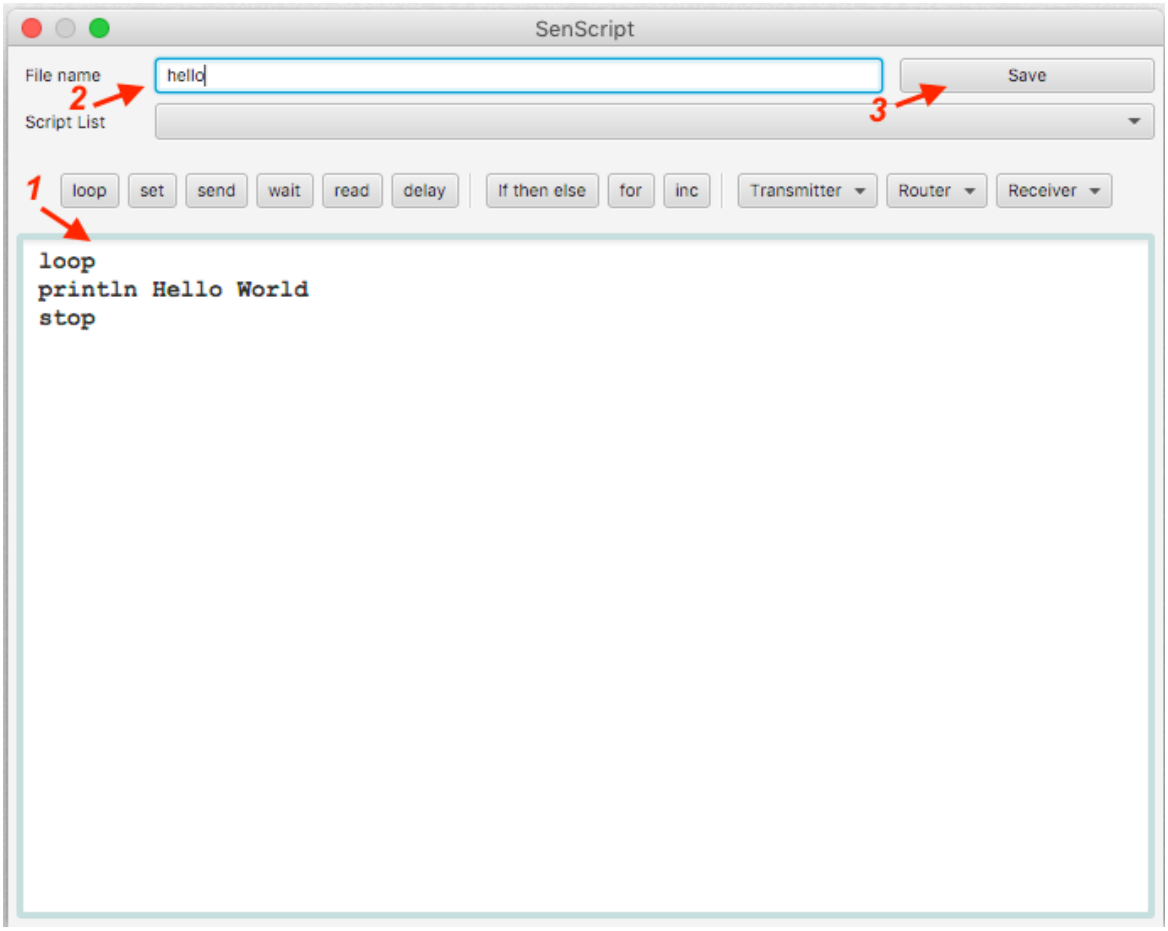



Рисунок 5.5 - Добавление скрипта

Шаг 5.

Назначение файла SenScript узлу датчика в соответствии с рисунком 5.6: выберите узел датчика на карте (1). перейдите к Параметры устройства в левой части главного окна (2). Затем выберите hello.csc файл в поле Scriptfile (3). И затем, нажмите на кнопку применить только в правом углу (4).

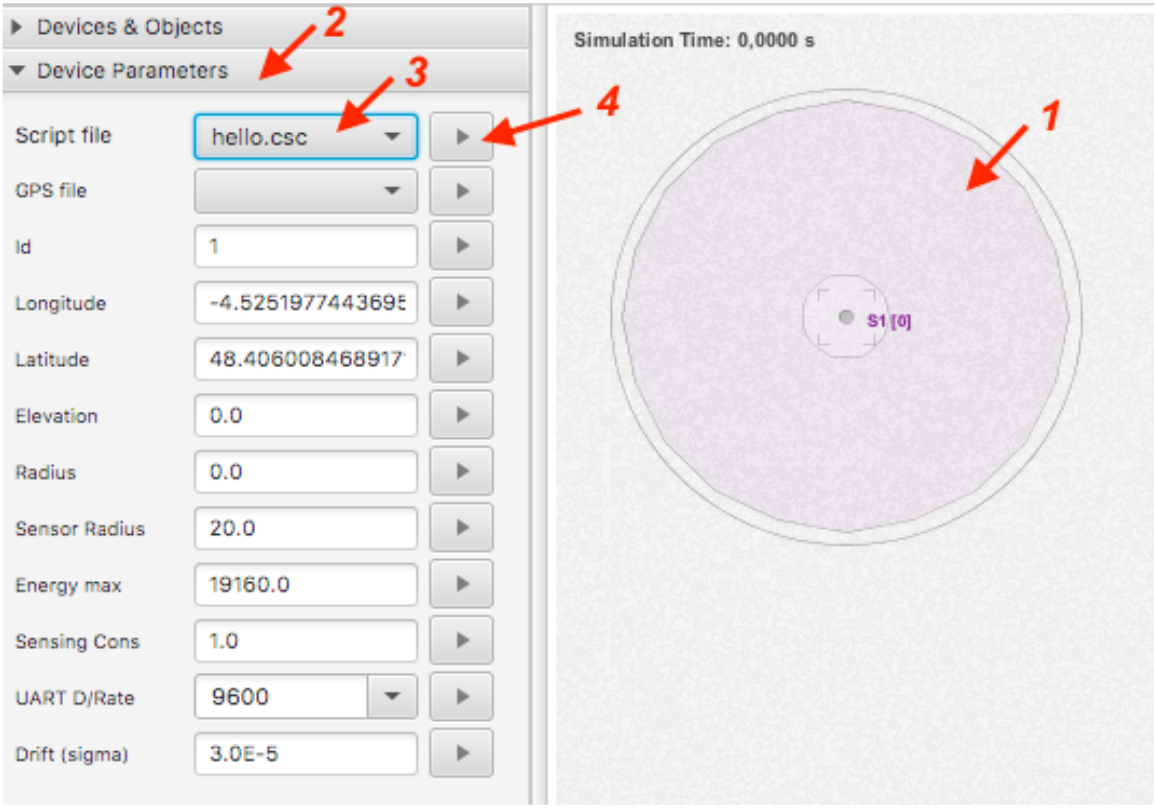



Рисунок 5.6 - Добавление скрипта для датчика

Обратите внимание, что как только скрипт будет назначен датчику, центр будет окрашен в оранжевый цвет. Это может помочь для графического обнаружения датчиков без скриптов соответствии с рисунком 5.7.


Рисунок 5.7 - Графическое обнаружение датчика

Шаг 6.

Запустите моделирование в соответствии с рисунком 5.8: в этом примере нет необходимости параметризовать моделирование, просто

нажмите кнопку  запустить моделирование на панели инструментов или в меню Параметры моделирования слева.

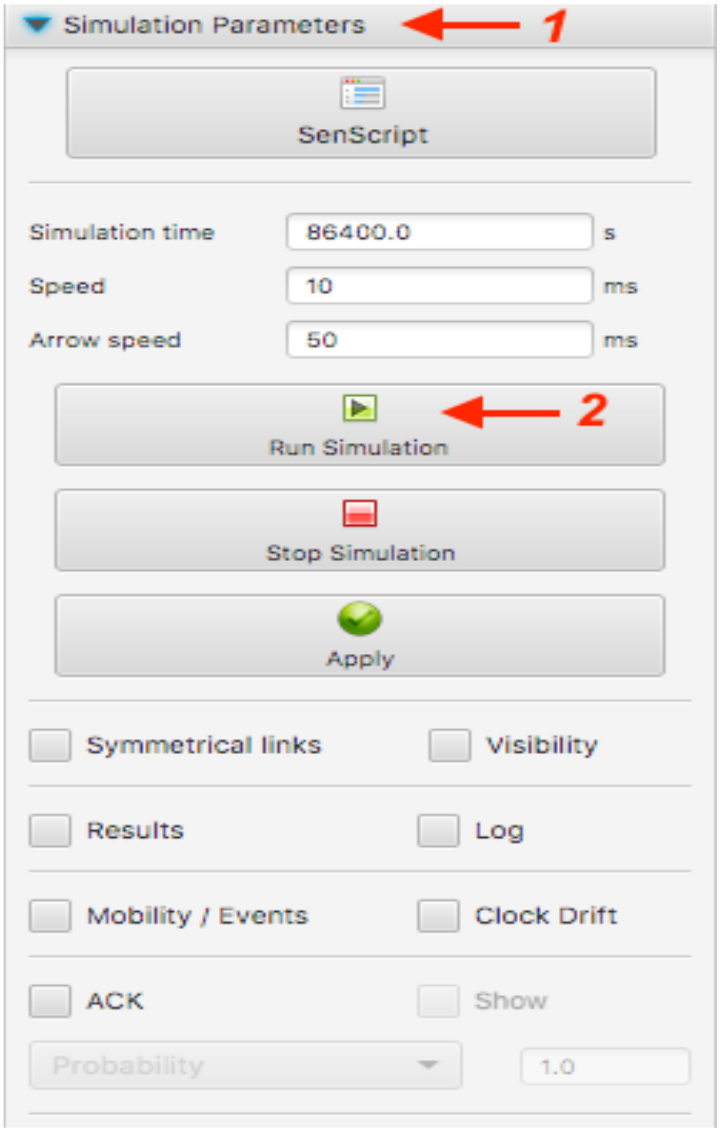



Рисунок 5.8 - Запуск моделирования

Пример 2: вычислить a+b

В этом примере показано, как вычислить сумму двух переменных a и b. Повторите все шаги примера 1, где только скрипт должен быть изменен следующим образом:

```
Loop
set a 7
set b 8
plus x $a $b
printa + b = $x
stop
```

Результат моделирования покажет в соответствии с рисунком 5.9: a + b = 15.

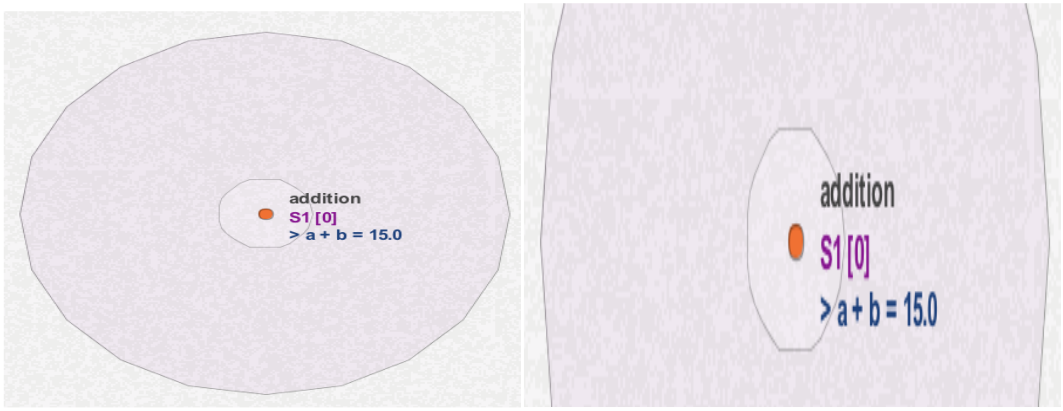



Рисунок 5.9 - Результаты моделирования

Если мы заменим команду печати следующей командой:
print $a + $b = $x.
Результат моделирования покажет соответствии с рисунком 5.10:
7 + 8 = 15.

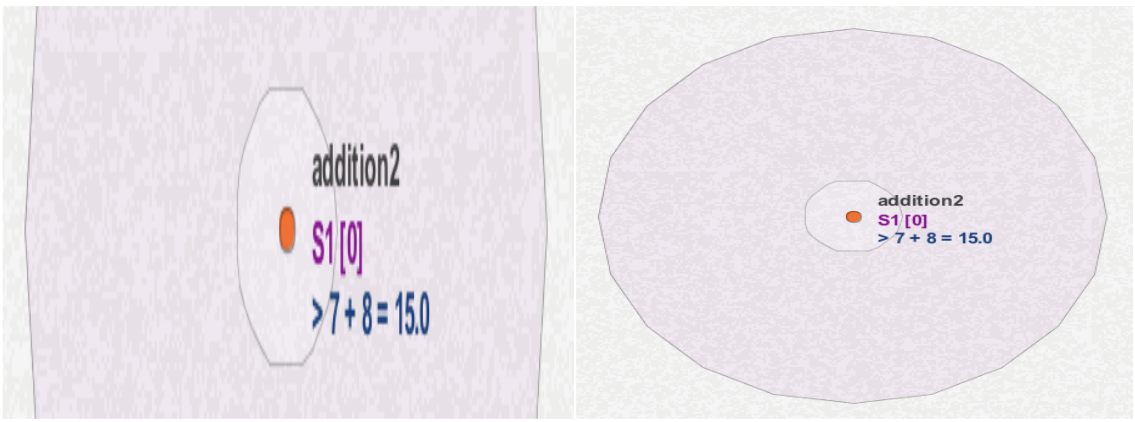



Рисунок 5.10 - Результат моделирования

Тест с помощью этого скрипта в соответствии с рисунком 5.11 (со скоростью моделирования 1000):

```
seta 7
loop
for b 0 11
mult x $a $b
print $a X $b = $x
delay 1000
end
```


Рисунок 5.11 - Тест с помощью скрипта

Пример 3: вычисление суммы вектора

В этом примере показано, как создавать таблицы и получать доступ к их элементам. В качестве примера мы постараемся создать таблицу с 5 строками и 1 столбцом. Затем мы выведем сумму его элементов.

Повторите все шаги примера 1, где только сценарий должен быть изменен следующим образом:

```
tab t 5 1
tset 3 t 0 0
tset 5 t 1 0
tset 2 t 2 0
tset 6 t 3 0
tset 4 t 4 0
set s 0
loop
fori 0 5
tget x t $i 0
plus s $s $x
end
print $s
stop
```

Результат моделирования будет отображаться 20 в соответствии с рисунком 5.12.

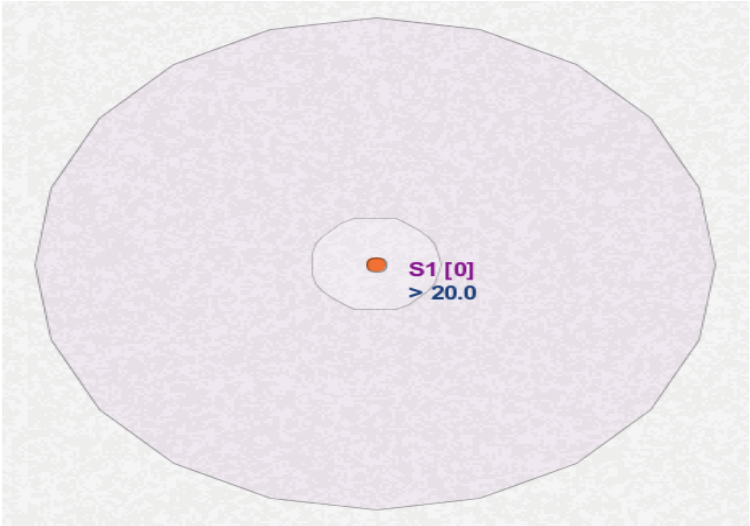



Рисунок 5.12 - Результат моделирования

Пример 4: маркировка узлов

Узел маркера представляет в реальности узел датчика с включенным светодиодом. Это помогает сделать видимое действие вместо того, чтобы отображать сообщения. Маркировка узла выполняется командой Enscriptmark 1. Пометки узел датчика выполняется командой mark 0.

Чтобы пометить узел, выполните те же действия, что и в Примере 1, со следующим сценарием:

loop
mark 1
stop

В результате моделирования будет показан датчик, отмеченный желто-зеленым центром в соответствии с рисунком 5.13.

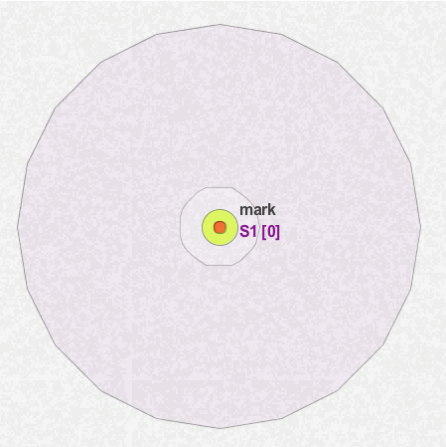



Рисунок 5.13 - Смоделированный датчик

## 6 Практические задания

Задание 1: Прислать свои координаты.

В этом примере указанный узел датчика отправит сообщение, чтобы запросить у определенного узла датчика в сети его координаты. В этом примере мы увидим, что нет необходимости в процессе обнаружения соседей и локальном сохранении таблицы соседей. Это всего лишь пример, а не рекомендуемый протокол маршрутизации. Потому что в реальной сети выполнение этого процесса для каждого действия (запрос координат) будет действительно очень энергозатратным. Обратите внимание, что моделирование CupCarbon и сценарий Sun выполняются на уровне приложения.

Создайте новый проект и добавьте 100 сенсорных узлов случайным образом.

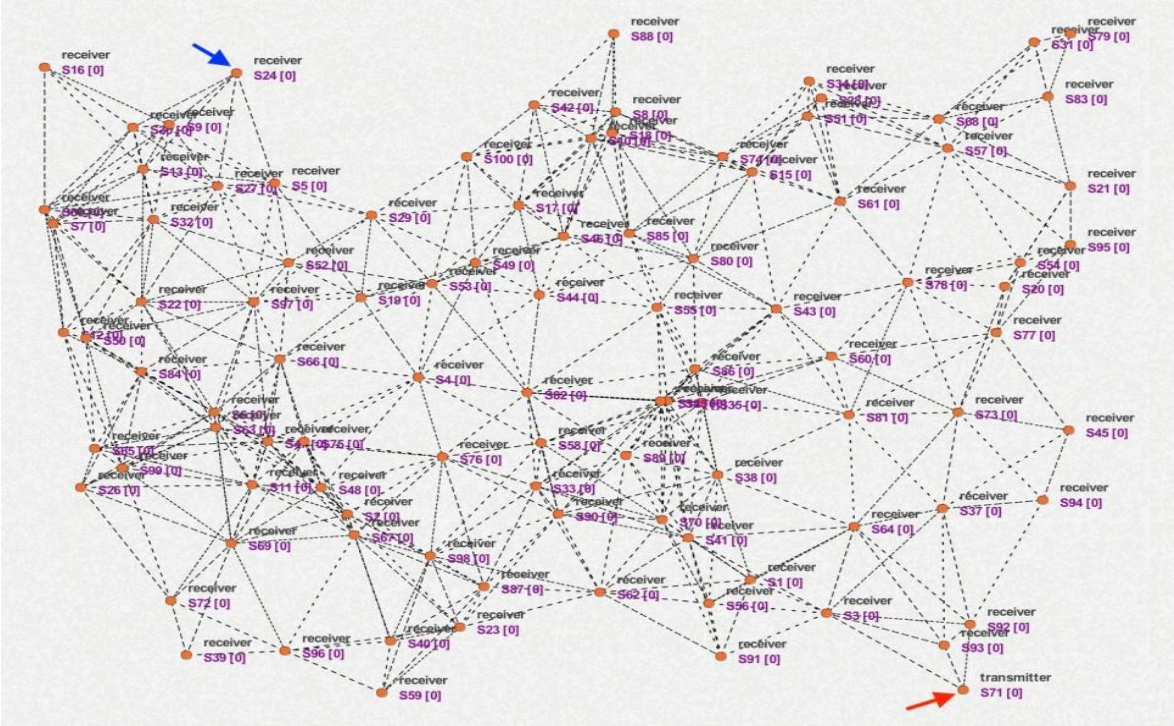



Рисунок 6.1- Визуализация

Чтобы получить эту визуализацию (рисунок 6.1), нажмите на кнопку Соединения в строке состояния. Возьмем любой узел датчика, например, S71, начальный узел, расположенный в крайнем правом нижнем углу сети, как показано красной стрелкой на рисунке 6.1 выше. Этот сенсорный узел запросит координату сенсорного узла S24, расположенного в левом верхнем углу сети, как показано синей стрелкой на рисунке выше. Эти идентификаторы (24 и 71) должны быть адаптированы и изменены в вашем примере. Затем замените эти значения в вашем скрипте значениями из вашего примера.

Во-первых, мы начнем со скрипта, который позволяет просто обнаружить и отметить соответствующий узел датчика (например, S24). Для этого S71 отправит в широковещательную передачу сообщение, сформированное A и 24, которое означает: “Я ищу узел датчика с идентификатором 24”. Другие сенсорные узлы, которые получат это сообщение, сделают то же самое один раз, если они не являются сенсорным узлом S24. В противном случае они будут помечены.

Сценарий стартового узла (S71) задается следующим образом (рисунок 6.2):

```
atgetididdata
d $id A
24send$d

lo
op
st
```

Рисунок 6.2 - Сценарий стартового узла

Сценарий работы приемников (других сенсорных узлов) задается следующим образом (рисунок 6.3):

```
atget id
idset
recA
0loop

wai
tre
adm

rdata $m rid type
infoif(($type==A)&&($recA=
=0))

    setrecA
    1if($info==$id)



        mark
    1else

        data d $id A
        $infosend$d

    end
```

Рисунок 6.3 - Сценарий работы приемников

Переменная id представляет собой текущий идентификатор датчика. Переменная RecA означает “уже полученное сообщение A” для того, чтобы отправить сообщение A один раз или быть помеченным один раз, если его идентификатор является тем, который исследуется (здесь, 24). Информация представлена здесь 24, что и

было сделано исследованием. Мы называем это информацией, потому что на следующем шаге эта информация станет координатами сенсорного узла 24, который будет отправлен в сенсорный узел S71. Имитация со скоростью моделирования = 0 и скоростью стрелки моделирования = 500. Моделирование даст следующий результат (рисунок 6.4):

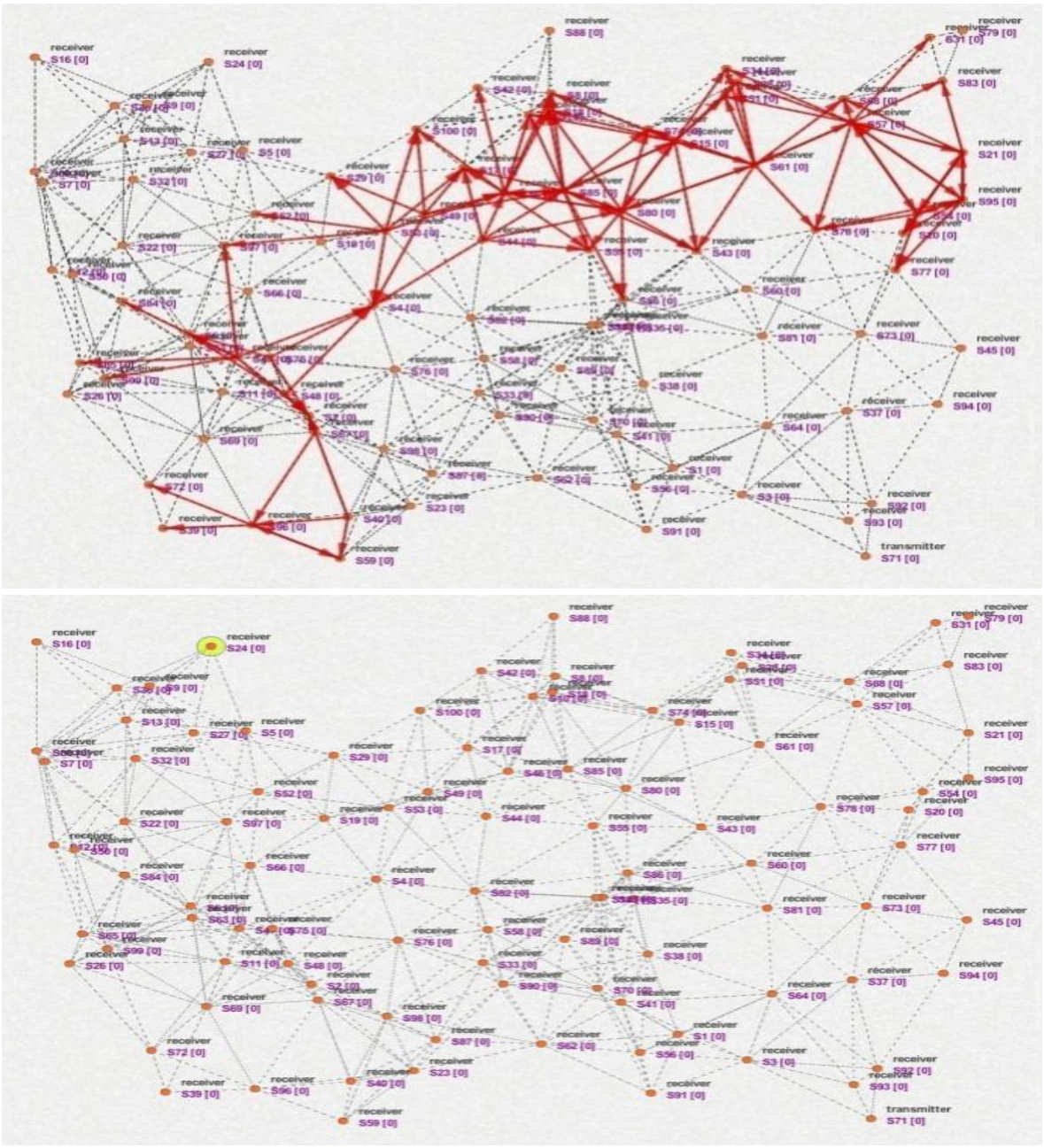



Рисунок 6.4 - Результаты моделирования

Теперь отмеченный узел датчика отобразит свои координаты и отправит их в виде сообщения B к предыдущему узлу датчика (здесь S5), который отправил ему сообщение A. Этот предыдущий узел датчика (S5) снова отправит к B своему предыдущему узлу датчика и

так далее, пока не достигнет начального узла датчика S71. Этот последний будет помечен и отобразит полученные координаты. Предыдущие сценарии будут завершены следующим образом. Сценарий стартового узла (S71) будет выглядеть следующим образом рисунок 6.5):

```
atgetididdata
d $id A
24send$d

loopwa
itread
m

rdata $m rid type x
yif($type==B)

   mark1print
   $x $ystop

end
```

6.5 - Сценарий стартового узла

```
getpos
patget
id
idset
recA
0loop
wa
it
re
ad
m
rdata$mridtypeinfoinfo2
if(($type==A) &&
    ($recA==0))setrecA
    1if($info==$id)
        mark1
        print$p
        data d $id
        B
        $psend$d$r
        id
    else
        setprev$rid
        data d $id A
        $infosend$d
    end
end
if($type=
    =B)mar
    k1
    data d $id B $info
    $info2send$d$prev
end
```

Рисунок 6.6 - Сценарий сенсорных узлов

Моделирование даст следующий результат (рисунок 6.7), где имитация со скоростью моделирования = 0 и скоростью стрелки моделирования = 500.

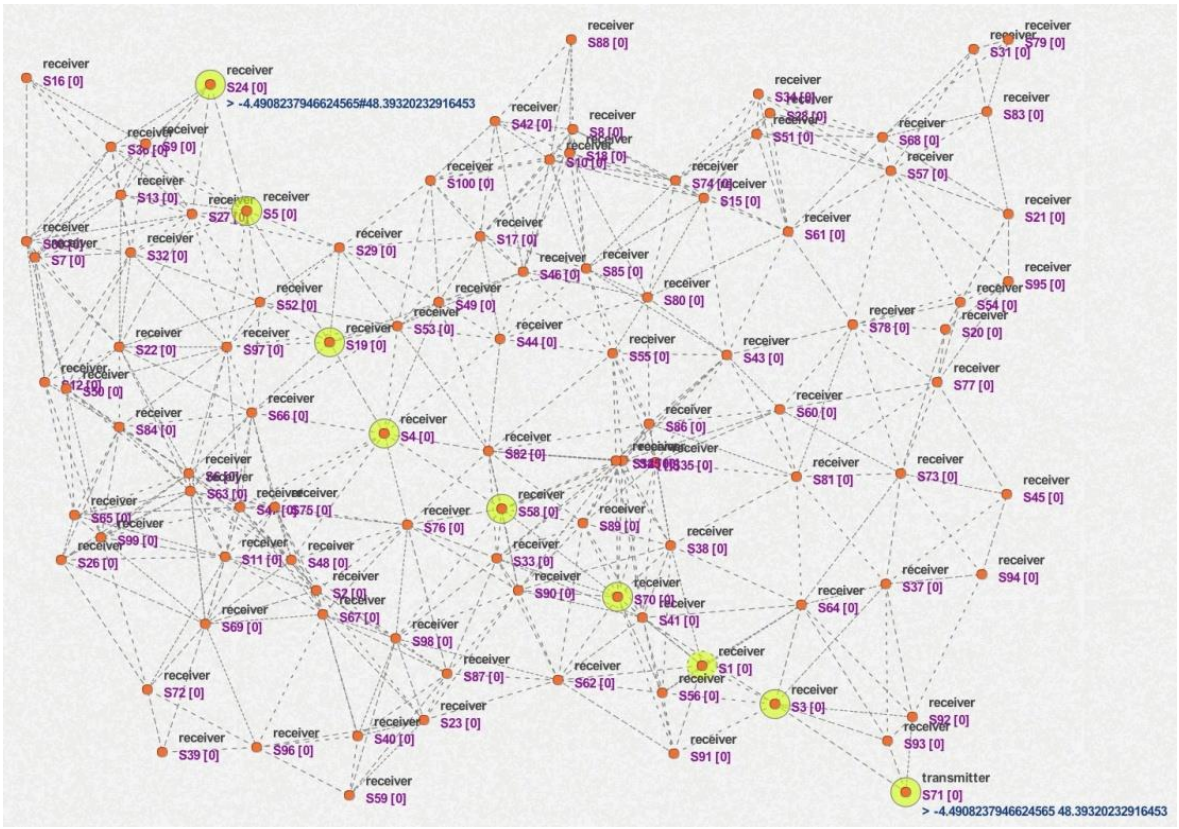



Рисунок 6.7 - Результат моделирования

Задание 2: Найти крайний левый узел

Этот пример вдохновлен примером поиска лидера в сети. Сначала каждый узел датчика генерирует случайное значение от 0 до 1000, а затем помечает узел датчика, имеющий максимальное сгенерированное значение.

Сценарий работы сенсорных узлов один и тот же, и он задается следующим образом (рисунок 6.8):

```
randb x 0
1000print$x

mark1

setvmax
$xsend
$vmaxloop

wait
```

```
readv

if ($v >
    $vmax)mark
    0

    setvmax
    $vsend$v

end
```

Рисунок 6.8 - Сценарий работы сенсорных узлов

Результат моделирования может быть следующим (Рисунок 6.9). Тест со скоростью моделирования=0 и для стрелок со скоростью =0, 100 и 500. Если существует много сенсорных узлов с одинаковым максимумом, то все они будут помечены.

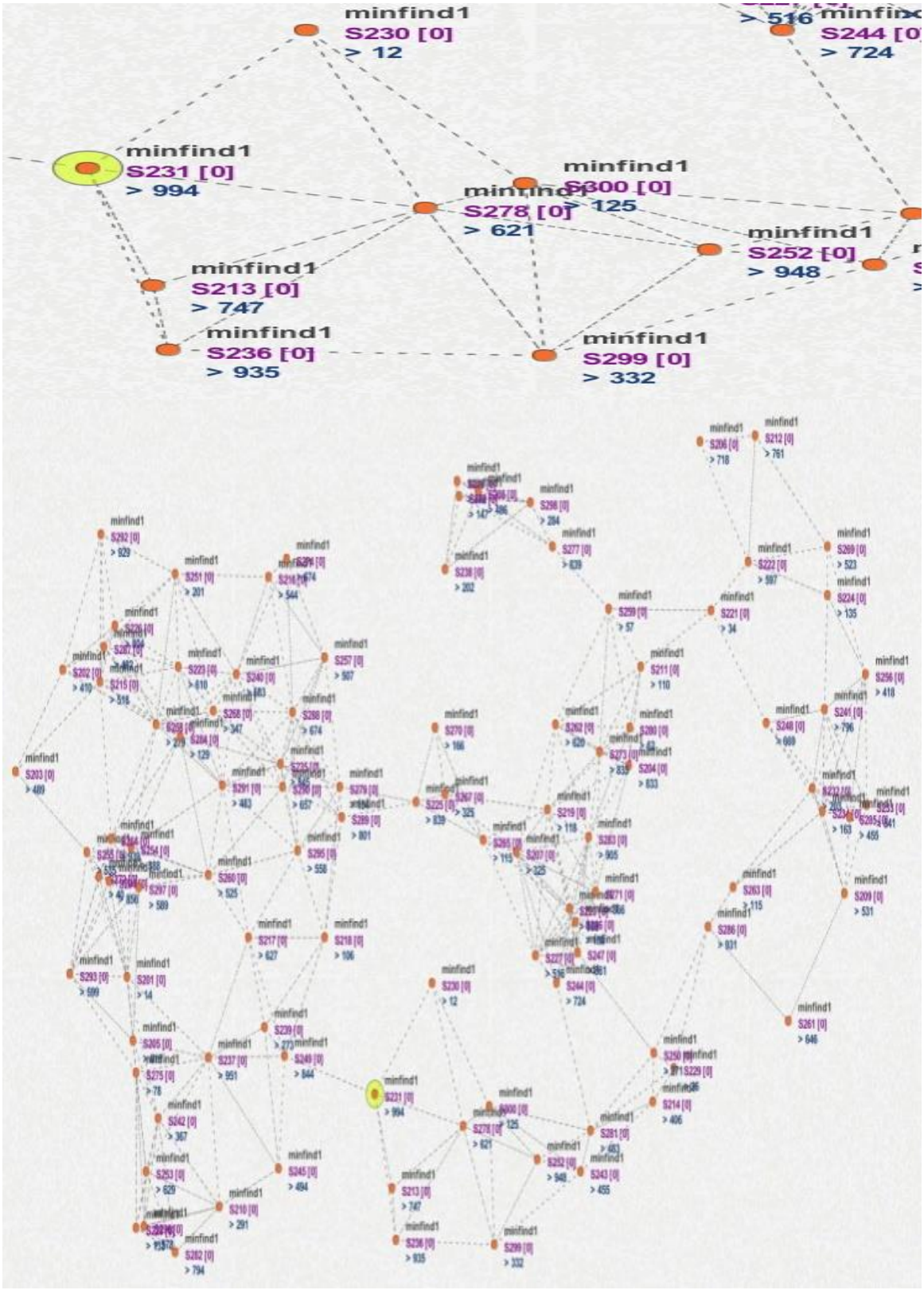



Рисунок 6.9 - Результаты моделирования

Этот алгоритм может быть использован также для поиска сенсорного узла с минимальным идентификатором. В этом случае будет отмечен только один узел датчика. Сценарий почти такой же, как и предыдущий, только переменная x будет заменена идентификатором сенсорного узла вместо случайного значения, и нам не нужно его печатать, так как он представляет собой сам идентификатор, который уже отображается на карте для каждого сенсорного узла. Кроме того, поскольку мы ищем min вместо max, мы изменим условие if на < (Рисунок 6.10).

```
atget id
vminmark1
send
$vminlo
op
wai
tre
adv
if ($v
    <$vmin)m
    ark0
    setvmin
    $vsend$v
end
```

Рисунок 6.10 - Сценарий

Результат моделирования даст (здесь idmax равен 201):

```
getpos2vmin
ymark1

send
$vminloop

waitre
adv

if ($v <
    $vmin)mark0

    setvmin
    $vsend$v

end
```

Рисунок 6.11 - Сценарий

Этот же сценарий теперь можно использовать для поиска сенсорного узла, который находится в крайнем левом углу сети (Рисунок 6.11). Это просто узел датчика, имеющий минимальную координату x.

Результат моделирования дает следующее (Рисунок 6.12):

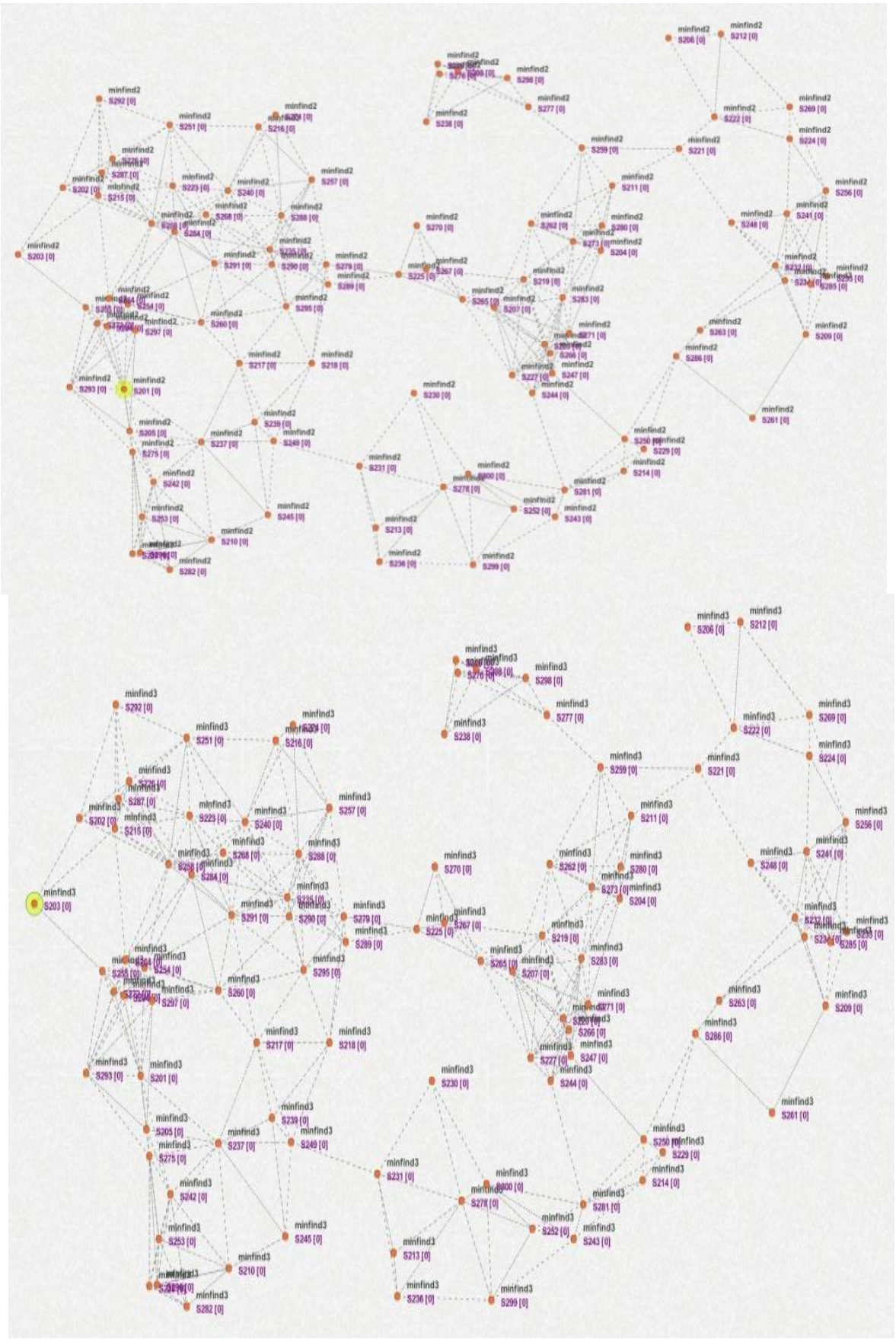


Рисунок 6.12 - Результаты моделирования

Задание 3: Имитация алгоритма D-LPCN (версия 1).

Алгоритм D-LPCN позволяет найти граничные сенсорные узлы сети. Он начинается с любого граничного сенсорного узла (например, крайнего левого). Затем каждый узел вычисляет углы, образованные ранее найденным граничным узлом и каждым из его соседей; а затем он выбирает узел, который образует наименьший угол, в качестве следующего граничного узла.

Для написания этого алгоритма мы будем использовать 3 типа сообщений. Сообщение AC, чтобы попросить соседей отправить свои координаты, сообщение CS для отправки координат и сообщение SN, чтобы сообщить датчику, что это пограничный узел. Сценарий алгоритма DLPCN приведен ниже. Во-первых, (строки 1-3), каждый узел датчика назначит переменной cid свой текущий идентификатор и вычислит свои текущие координаты cx и cy. Поскольку этот сценарий выполняется всеми сенсорными узлами сети, мы сначала добавим переменную (здесь она равна 49, которая представляет крайний левый сенсорный узел) для обозначения начального узла. Некоторые части скрипта будут выполняться только первым узлом. Затем (строки 5-10) только первый узел выполнит эту часть. Он выполнит его один раз. Вот почему в строке 6 переменная сначала устанавливается в 0. Вычислить углы, образованные предыдущим граничным узлом и его соседями. В строках 5-8 мы будем считать, что существует виртуальный сенсорный узел, расположенный в левой части стартового узла, который имеет координаты (px=cx-1 и py=cy). Затем будет рассмотрено сообщение SN, так как начальный узел также является граничным узлом. За строкой 10 (сообщение SN) непосредственно следует строка 22. Блок 23-26 представляет собой условие остановки, которое представляет собой двукратное посещение начального узла. В первый раз переменная first равна 0, затем условие остановки не проверяется, и переменная first будет установлена в значение -1 (строка 26). Однако если мы вернемся к этому блоку во второй раз, условие (first == -1) будет проверено, то будет вызвана команда stop. Если условие остановки не проверено, то, как и рассматриваемый блок, выполняется после получения сообщения SN, что означает, что приемник является граничным датчиком узла. Линия 27 будет отмечать этот датчик, а затем, от линии 28 до 45 будет запущен процесс запроса и получения координат соседей. Для каждого полученного сообщения (координаты) угол будет вычислен и сравнен с ранее вычисленным углом, чтобы определить наименьший. Строка 28 позволяет получить значение координат px и py предыдущего граничного узла. Линия 29 образует сообщение с углом, равным 10 радианам. Строка 30 используется для получения количества соседей и их идентификаторов. Линии 31-34 используются для отправки каждому соседнему узлу сообщения переменного тока с просьбой отправить свои координаты. Линии 35-45 используются для ожидания ответа соседей.

Если получено сообщение CS, то координаты передаются и могут быть использованы для расчета угла (строка 40) с целью выбора минимального (строка 42). Функция smin позволяет найти минимальное значение между двумя значениями, извлеченными из сообщений, сформированных идентификатором сенсорного узла и соответствующей ему переменной. Как только будет найден минимальный угол, а также соответствующий ему сенсорный узел (id), то будет отправлено сообщение SN (строки 46 и 47), чтобы сообщить полученному сенсорному узлу, что он является граничным сенсорным узлом. Обратите внимание, что линии 17-20 выполняются сенсорными узлами, которые получают сообщение AC для отправки своих координат (с сообщением CS: строка 18). Для простоты мы рассматриваем как условие остановки, когда первый начальный узел посещается дважды, что не является реальным условием остановки. Выполнение приведенного сценария (Рисунок 6.13) приведет к следующей ситуации (Рисунок 6.14):

```
11    1:    atgetidcid 2:     getpos2 cxcy
12    3:    setfirst 49

13    4:    loop
14    5:    if ($cid==$first)
15    6:    setfirst 0
16    7:    minuspx $cx 1
17    8:    setpy $cy
18    9:    data p 0 $type $px $py
19    10:   set type SN
20    11: else
21    12:   wait
22    13:   read p
23    14:   rdata $p id type
24    15: end
25    16:
26    17: if ($type==AC)
27    18:   data p $cid CS $cx $cy
28    19:   send $p $id
29    20: end
30    21:
31    22: if ($type==SN)
32    23:   if ($first==-1)
33    24:   stop
34    25:   end
35    26:   set first -1
36    27:   mark 1
37    28:   rdata $p id type pxpy
38    29:   data m $cid 10
39    30:   atnd n tneg
40    31:   for i 0 $n
41    32:   vgetnegtneg $i
42    33:   data p $cid AC
43    34:   send $p $neg
44    35:   wait
45    36:   read p
46    37:   rdata $p id type
47    38:   if ($type==CS)
48    39:   rdata $p id type x y
49    40:   angle2 a $px $py $cx $cy $x $y
```

```
50      41:   data p $id $a
51      42:   smin m $m $p
52      43:   rdata $m id a
53      44:   end
54      45:   end
55      46:   data p $cid SN $cx $cy
56      47:   send $p $id
57      48: end
58      1:    atgetidcid 2:     getpos2 cxcy
59      3:    setfirst 49
60      4:    loop
61      5:    if ($cid==$first)
62      6:    setfirst 0
63      7:    minuspx $cx 1
64      8:    setpy $cy
65      9:    data p 0 $type $px $py
66      10:   set type SN
67      11: else
68      12:   wait
69      13:   read p
70      14:   rdata $p id type
71      15: end
72      16:
73      17: if ($type==AC)
74      18:   data p $cid CS $cx $cy
75      19:   send $p $id
76      20: end
77      21:
78      22: if ($type==SN)
79      23:   if ($first==-1)
80      24:   stop
81      25:   end
82      26:   set first -1
83      27:   mark 1
84      28:   rdata $p id type pxpy
85      29:   data m $cid 10
86      30:   atnd n tneg
87      31:   for i 0 $n
88      32:   vgetnegtneg $i
89      33:   data p $cid AC
90      34:   send $p $neg
91      35:   wait
92      36:   read p
93      37:   rdata $p id type
94      38:   if ($type==CS)
95      39:   rdata $p id type x y
96      40:   angle2 a $px $py $cx $cy $x $y
97      41:   data p $id $a
98      42:   smin m $m $p
99      43:   rdata $m id a
100     44:   end
101     45:   end
102     46:   data p $cid SN $cx $cy
103     47:   send $p $id
104     48: end
```

Рисунок 6.13 - Сценарий

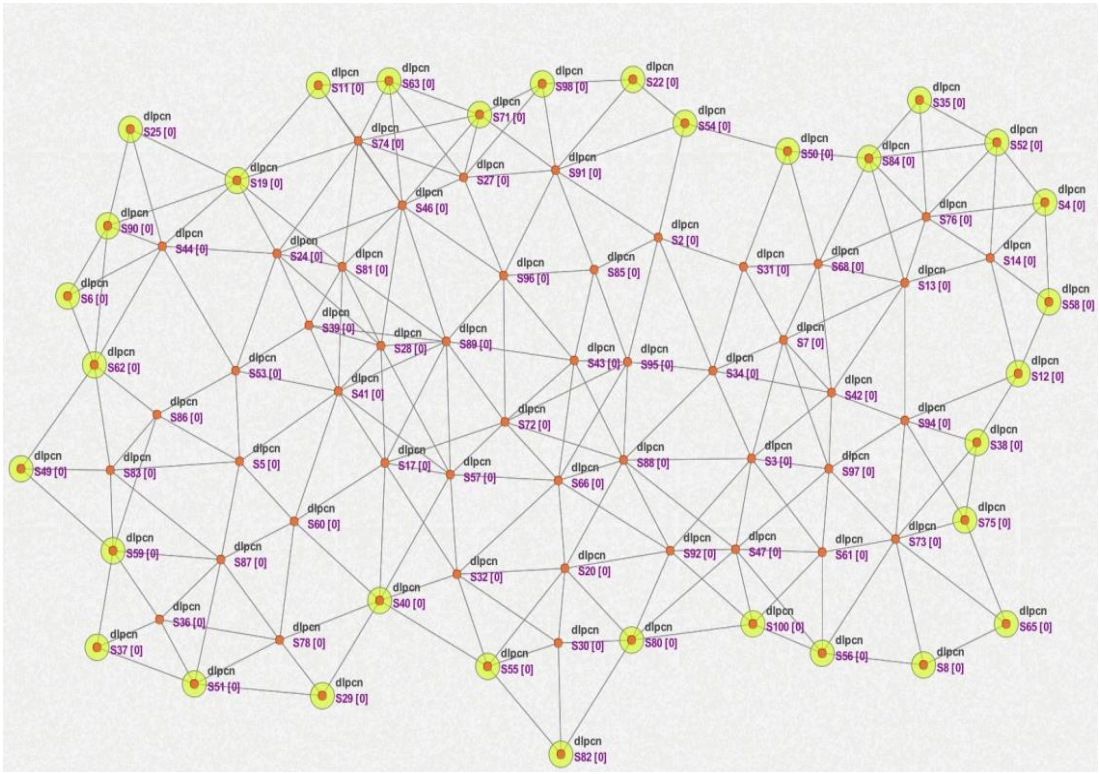



Рисунок 6.14 - Результат моделирования

Этот сценарий можно записать следующим образом, где мы рассматриваем широковещательные сообщения соседям (Рисунок 6.15).

```
1  :atgetidcid
2  :getpos2cxcy
3  :setfirst49
4  :loop
5  :if($cid==$first)
6  :    setfirst0
7  :    minuspx$cx1s
8  :    etpy$cy
9  :    data p 0 $type $px
   :    $pysettypeSN
 10:

11 :else
12 :    wait
13 :    readp
14 :    rdata$pidtype
15 :end
```

```
16 :
: if ($type==AC)
:       data p $cid CS $cx $cy
:       send $p $id
: end
: if ($type==CS)
:       rdata $p id type x y
:       angle2 a $px $py $cx $cy $x $y
:       data p $id $a
:       smin m $m $p
:       rdata $m id a
:       inci
28 :    if($i==$n)
:       data p $cid SN $cx $cy
:       send $p $id
:       end
: end
: if ($type==SN)
34 :    if($first==-1)
:       stop
:       end
:       set first -1
:       rdata $p id type pxpy
:       mark 1
:       data m $cid 10
:       atnd n
:       set i 0
:       data p $cid AC
:       send $p
: end
```

Рисунок 6.15 - Сценарий

Задание 4: Моделирование алгоритма D-LPCN (версия 2)

В этом примере мы покажем, как мы можем объединить много видимых сценариев в один сценарий. Например, сценарий, приведенный ранее в примере 3, запускается с заданного сенсорного узла. Это узел датчика, который находится в крайнем левом углу сети. Пример 2 показывает другой сценарий, позволяющий найти этот крайний левый узел датчика. Поэтому ниже мы покажем, как написать код, который сначала находит крайний левый узел датчика, используя сценарий примера 2, а затем запускает алгоритм D-LPCN, используя пример 3. Для этого мы создадим новую переменную step. Эта переменная будет равна 1, а затем 2, чтобы определить, какой алгоритм будет выполняться. Назовем код Примера 2 Кодом 1, а код примера 3 Кодом 2. Тогда окончательный сценарий будет иметь почти следующую структуру (Рисунок 6.16):

Код 1 позволит определить узел датчика, который находится в крайнем левом углу. Идентификатором этого сенсорного узла будет значение переменной first, которая используется в коде 2 для начального узла. В этой новой ситуации в сценарии, который находит крайний левый узел датчика, команда wait приведет к ситуации блокировки.

```
set step 1
if($step == 1)
if (condition)
set step 2
end Code1
end
if($step == 2)
Code2
end
```

Рисунок 6.16 - Сценарий

Ни один узел датчика не может продолжать выполнение. Тогда невозможно перейти ко второму шагу. Чтобы преодолеть это ограничение, мы будем использовать команду wait, которая будет ждать получения сообщений в течение определенного времени t и продолжать выполнение после этого времени. В нашем случае это время представляет собой необходимое время для завершения первого процесса поиска крайнего левого узла. Если по истечении этого времени нет прочитанных сообщений (пустое сообщение), то приступаем ко второму этапу определения граничных узлов с помощью алгоритма D-LPCN.

Окончательный сценарий дается следующим образом (Рисунок 6.17).

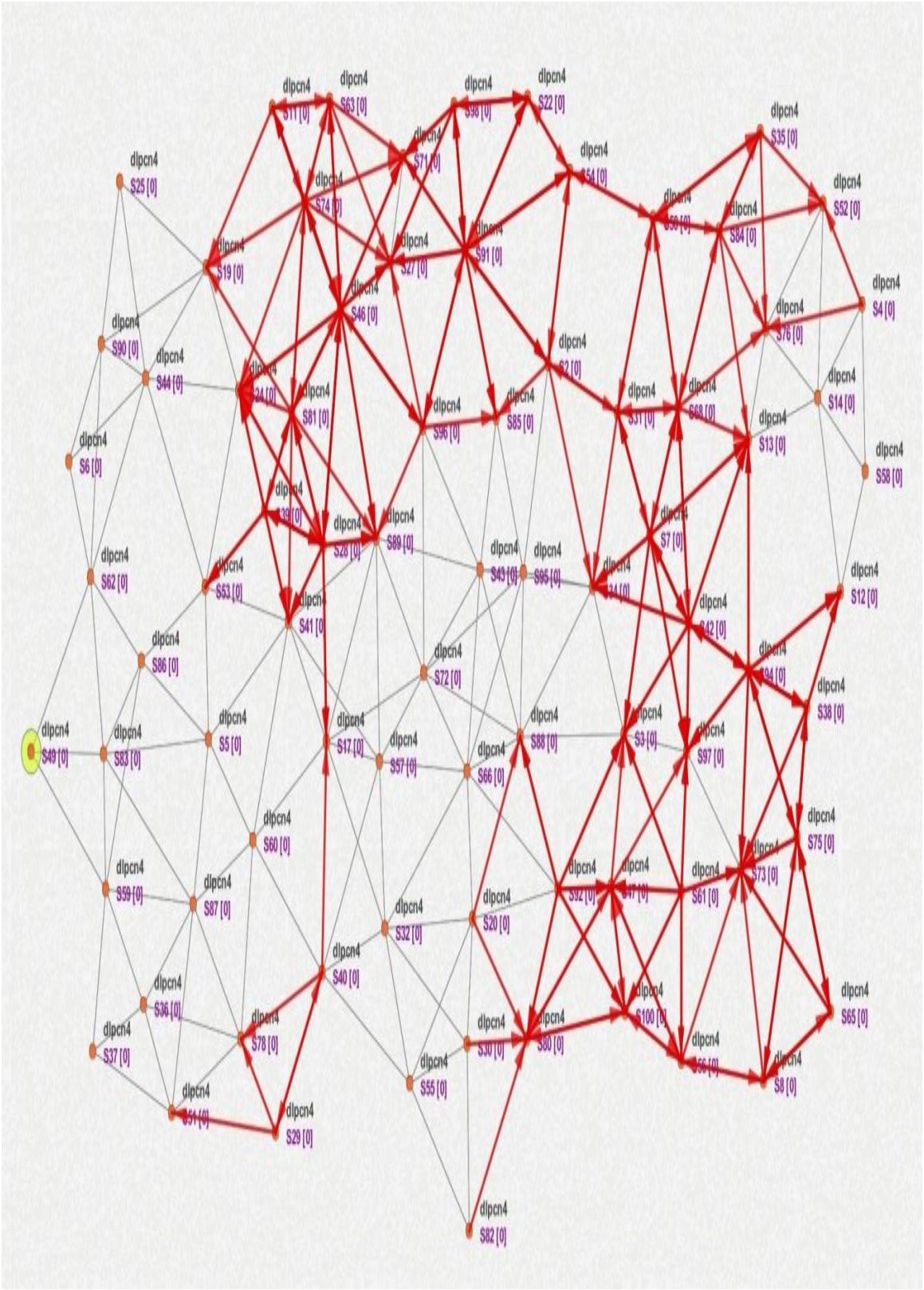
Имитация со скоростью моделирования = 0 и скоростью стрелки = 100. Для визуализации, поскольку скорость моделирования для первого шага более быстрая, чем для второго, рекомендуется изменить параметр скорости стрелки, например, с 5 для шага 1 до 200 для шага 2. Эту процедуру трудно выполнить вручную.

```
: set step 1
: set first -2
: getpos2 vmin y
: set marked 1
: send $vmin
: loop
7 : if($step == 1) 8 :  wait 2000
9 :   read v
10 :  if($v == \)
:     atget id cid
:     getpos2 cx cy
:     set step 2
:     if ($marked == 1)
:     set first $cid
:     end
:     else
:     if ($v < $vmin)
:     set marked 0
:     set vmin $v
:     send $v
:     end
: end
25 : if($step == 2)
```

```
:       if ($cid==$first)
:       set first 0
:       minus px $cx 1
:       set py $cy
:       data p 0 $type $px $py
:       set type SN
:       else
:       wait
:read p
:       rdata $p id type
:       end
:       if ($type==AC)
:       data p $cid CS $cx $cy
:       send $p $id
:       end
:       if ($type==CS)
:       rdata $p id type x y
:       angle2 a $px $py $cx $cy $x $y
:       data p $id $a
:       smin m $m $p
:       rdata $m id a
:       inci
48 :    if($i==$n)
49 :    data p $cid SN $cx $cy
:       send $p $id
:       end
:       end
:       if ($type==SN)
54 :    if($first==-1)
:       stop
:       end
:       if($first==0)
:       set first -1
:       end
:       set first -1
:       rdata $p id type pxpy
:       mark 1
:       data m $cid 10
:       atnd n
:       set i 0
:       data p $cid AC
:       send $p
:       end
: end
```

Рисунок 6.17 – Сценарий

Именно поэтому мы рекомендуем использовать команду simulation speed для форсирования различных значений для каждого шага (Рисунок 6.18).



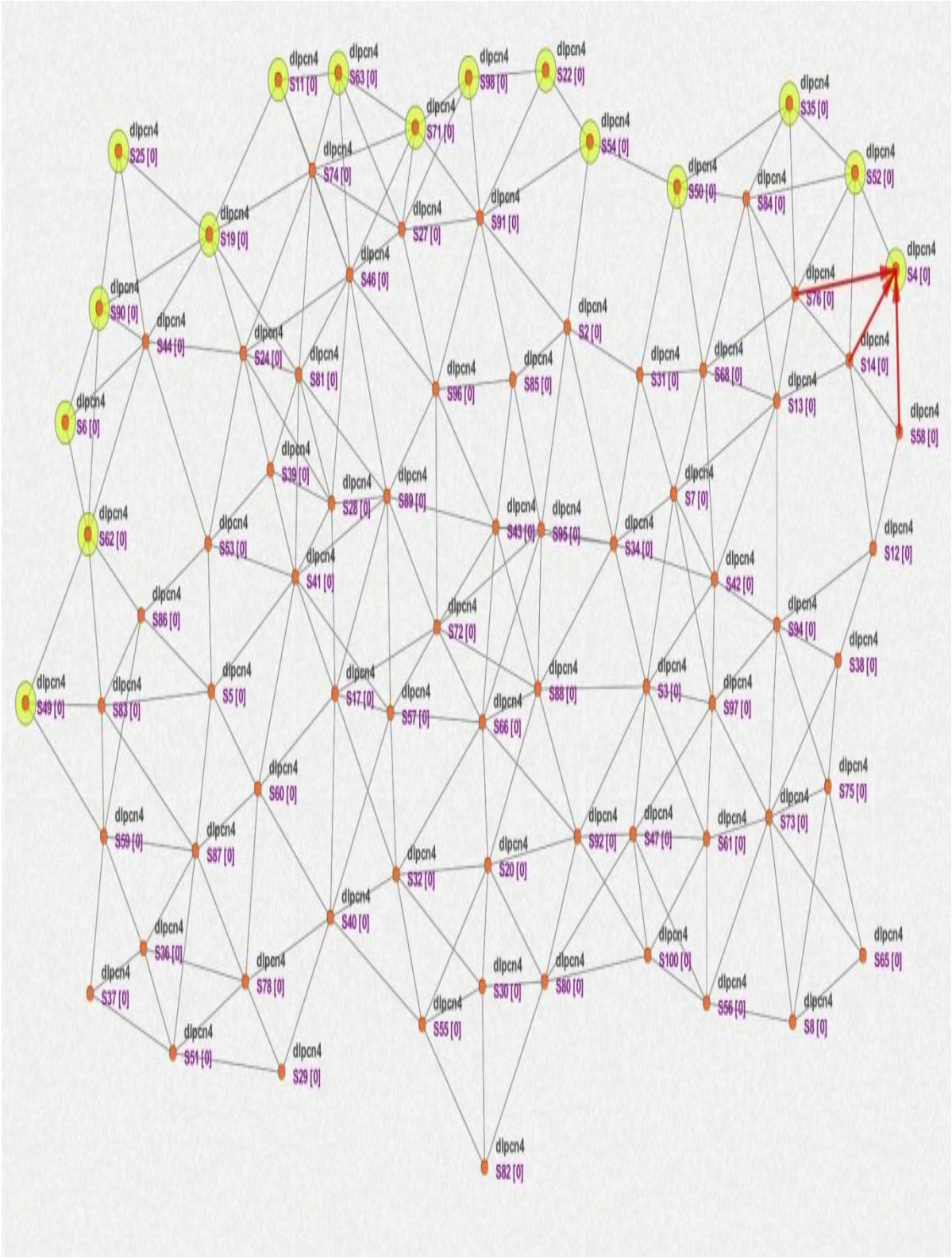



Рисунок 6.18 - Результаты моделирования

Если мы добавим инструкцию “edge $id” между строками 48 и 49 последнего скрипта, то получим следующий результат (с отмеченными граничными краями) (Рисунок 6.19).

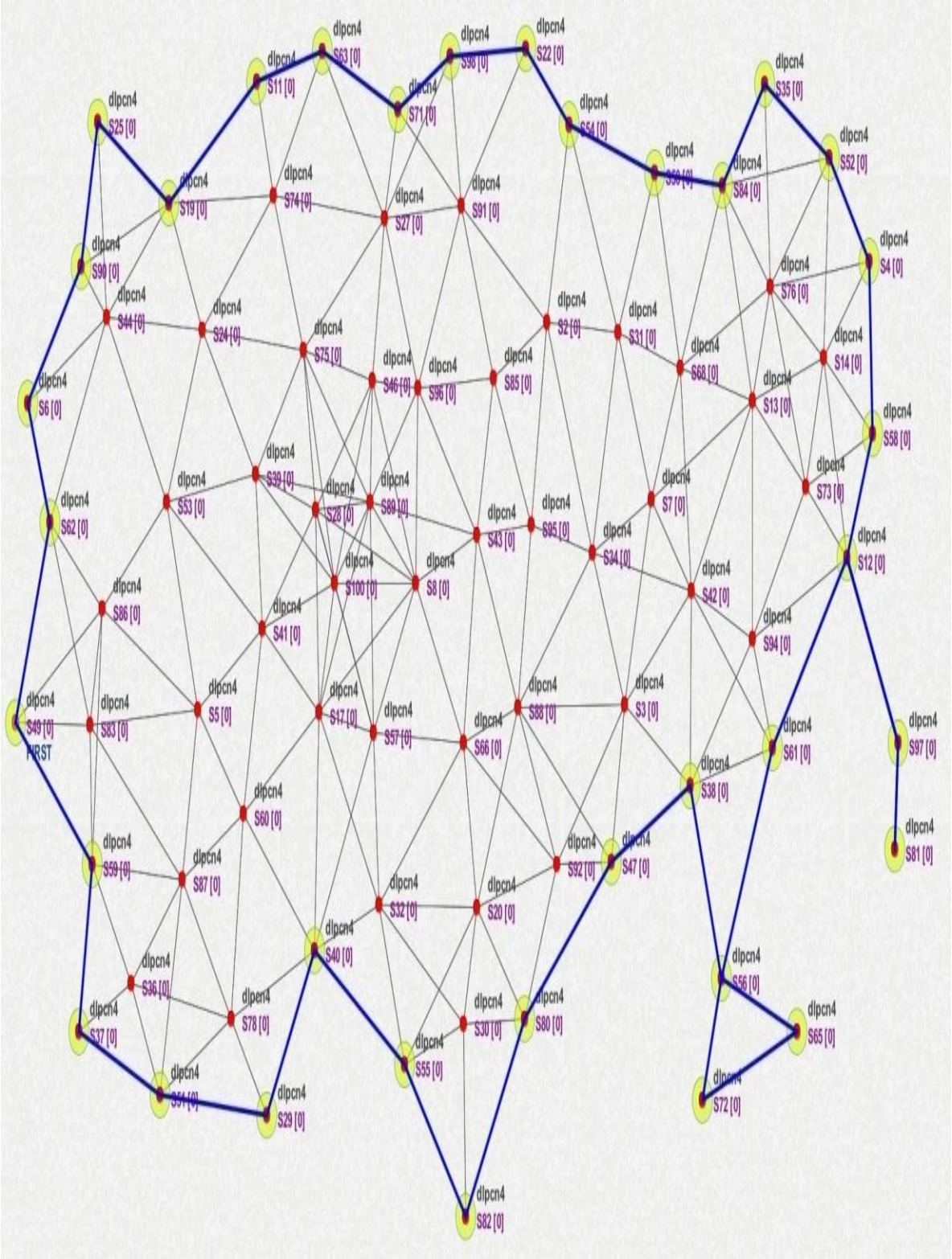



Рисунок 6.19 - Результат моделирования

Задание 5: Имитация алгоритма D-LPCN (версия 3).

В этом задании мы будем использовать командный скрипт. Во-первых, мы можем рассмотреть сценарий алгоритма D-LPC без указания значения переменной, сначала представляющей идентификатор начального узла (Рисунок 6.20). Этот скрипт должен быть сохранен с именем dlpc.csc. Чтобы вызвать его из другого скрипта, мы используем командный скрипт dlpc.

```
1  :atgetidcid
2  :getpos2cxcy
3  :setfirst49

4  :loop

5  :if($cid==$first)
   :
6
   :setfirst0
7
   :minuspx$cx1setp
    y$cy
   :
    data p 0 $type $px
    $pysettypeSN

11 :else
12 :    wait
13 :    readp
14 :    rdata$pidtype
15 :
end16:
17 :if($type==AC)
18 :    datap$cidCS$cx$cy
19 :    send$p$id
20 :end
21 :if($type==CS)
22 :    rdata$pidtypexy
23 :    angle2a$px$py$cx$cy$x$y
24 :    datap$id$a
25 :    sminm$m$p
26 :    rdata$mida
27 :    inci
28:     if($i==$n)

29 :        datap$cidSN$cx$cy
30 :        send$p$id
31 :    end
32 :end
33 :if($type==SN)
34:     if($first==-1)

35 :        stop
36 :    end
37:     if($first==-1)

38 :        setfirst-1
39 :    end
40 :    rdata$pidtypepxpy
41 :    mark1
42 :    datam$cid10
43 :    atndn
44 :    seti0
45 :    datap$cidAC
46 :    send$p
47 :end
```

Рисунок 6.20 – Сценарий

Затем в алгоритме майнинга мы вызовем предыдущий скрипт LPCN после завершения работы с помощью командного скрипта (строка 16). Обратите внимание, что мы добавим время задержки, как только будет найден первый узел (строка 12).

Это позволит дождаться, пока другие узлы закончат свой алгоритм MinFind и будут готовы запустить алгоритм LPC. Алгоритм майнинга в этом случае можно переписать следующим образом (рисунок 6.21). В этом случае, только алгоритм интеллектуального анализа данных присваивается узлам датчиков. Алгоритм LPC будет вызван автоматически из этого (строка 16).

```
1:atgetidcid
 1  :getpos2vminy
 2  :mark1
 3  :send$vmin
 4  :loop
 5  :
    mark$marke
    d
 6

 7:wait2000

 8:readv

 9:if($v==\)

10 :   if($marked==1)
11 :      setfirst$cid
12 :      delay1000
13 :   else
14 :      setfirst-3
15 :   end
16:    scriptdlpcn


    :else
18 :   if($v<$vmin)
19 :      mark0
20 :      setvmin$v
21 :      send$v
22 :   end
23 :end
```

Рисунок 6.21 - Сценарий

Таким образом, представленный нами симулятор CupCarbon способен удовлетворить потребности к имитационному моделированию современных телекоммуникационных систем. Он способен обеспечивать моделирование сетей следующего поколения, что обеспечивает возможность его использования в ближайшем будущем.

## Список использованной литературы


1 Соболь Б.В. Сети и телекоммуникации: учебное пособие / Б.В. Соболь. - РнД: Феникс, 2015. - 191 с.

2 Соболь Б.В. Сети и телекоммуникации: учебное пособие / Б.В. Соболь. - Рн/Д: Феникс, 2015. - 522 с.

3 Основные принципы и технические средства измерений параметров передачи для сетей PDH, SDH, IP, Ethernet и ATM [Электронный ресурс]: учебное пособие/ И.И. Власов [и др.].- Электрон. текстовые данные.- М.: Горячая линия - Телеком, 2014.- 480 с. - Режим доступа: http://www.iprbookshop.ru/12051.- ЭБС «IPRbooks».

4 Строганов М.П. Информационные сети и телекоммуникации. / М.П. Строганов, М.А. Щербаков. - М.: Высшая школа, 2008. - 151 с.

5 Гребешков А.Ю. Вычислительная техника, сети и телекоммуникации: учебное пособие для вузов. /А.Ю. Гребешков. - М.: ГЛТ, 2016. - 190 с.

6 Пескова С.А. Сети и телекоммуникации: учебник / С.А. Пескова. - М.: Academia, 2017. - 416 с.

7 Гусева А.И. Вычислительные системы, сети и телекоммуникации: учебник / А.И. Гусева. - М.:

8 Academia, 2016. - 640 с. Замятина, О.М. Вычислительные системы, сети и телекоммуникации. Моделирование сетей.: учебное пособие для магистратуры / О.М. Замятина. - Люберцы: Юрайт, 2016. - 159 с.

9 Проектирование, развитие и содействие умным городам: городской дизайн для IoT Solutions 1-е изд. 2017 г.Автор издания: Вангелис Анджелакис(редактор), Элиас Трагос (редактор), Генрих С. Пельс (редактор), Адам Каповиц(редактор), Alessandro Bassis.

10 Venkatesh Upadrista. IoT Business Strategy// IoT Standards with Blockchain. - Berkeley, CA: Apress, 2021. - С. 25-41.

11 Charith Perera, Chi Harold Liu, Srimal Jayawardena. The Emerging Internet of Things Marketplace From an Industrial Perspective: A Survey // IEEE Transactions on Emerging Topics in Computing. - 2015-12. - Т. 3, вып. 4. - С. 585-598. - ISSN 2168-6750. - doi:10.1109/tetc.2015.2390034

12 Хамзина Б. Е., Мендыбаев С. А., Наурзбаев К. К., Сагындык А.Б. Моделирование беспроводной сети в Cup carbon [Электронный ресурс] http://rmebrk.kz/journals/6143/71136.pdf. ВЕСТНИК ПГУ. – 2020. - №2 - 468 с.

13 Сети IoT/M2M: технологии, архитектура и приложения. IoT/M2M: Networks: technologies, architecture and applications / В. О. Тихвинский, В. А. Коваль, Г. С. Бочечка, А. И. Бабин. - Москва: Медиа Паблишер, 2017. - 319 с.

14 Администрирование сетей. Программа Cisco "CCNA Routing and Switching": учебное пособие и лабораторный практикум для студентов Инженерно-экономического института/А.Ю. Невский, О.Р. Баронов,

А.Ю. Модорский; Инженерно-экономический институт Национального исследовательского университета "МЭИ". - Москва: ВНИИГ систем, 2018. - 95 с.

15 Проектирование и анализ вычислительных сетей в программном продукте CiscoPacketTracer [Электронный ресурс]: электронное учебное издание: методические указания к лабораторным работам по дисциплине "Основы телекоммуникаций"/Аксенов А.Н. - Москва: МГТУ им. Н.Э. Баумана.

16 Официальное руководство по подготовке к сертификационным экзаменам CCENT/CCNA ICND1 Official Exam Certification Guide. - 2-изд. - М.: Вильямс, 2008. - 572 с.

17 Massinissa Saoudi, Ahcène Bounceur, Farid Lalem, Reinhardt Euler, M-Tahar Kechadi, Abdelkader Laouid, Madani Bezoui, Marc Sevaux, D-LPCN: A Distributed Least Polar-angle Connected Node Algorithm for Finding the Boundary of a Wireless Sensor Network, Ad Hoc Networks Journal, Elsevier, Volume 56, 1 March 2017, Pages 56-71.

18 UldisDzerkals. EVE-NG Professional Cookbook. Emulated Virtual Environment. Editors: Michael Doe Christopher Lim. Pages 9-10.

Хамзина Ботагоз Еркеновна, Толегенова Арай Сарсенкалиевна

# МОДЕЛИРОВАНИЕ СЕТЕЙ В ПРОГРАММНОМ ЭМУЛЯТОРЕ CUPCARBON

ISBN 978-601-257-407-4

Сдано в набор 20.11.2023 Подписано в печать 26.01.2024
Формат 60x84 $^{1}/_{16}$ Заказ №2421
Усл.печ.л. 6, 25 Тираж 20 экз.

Типография Казахского агротехнического исследовательского университета им. С.Сейфуллина, 2024
010011, г. Астана, пр. Жеңіс 62а, тел. 39 39 17