WPAN – BT SIMULATION PACKAGES

V. Atanasovski[1], L. Gavrilovska[1], L. Ilijoski[2]

[1]Faculty of Electrical Engineering, Ss. Cyril and Methodius University, Skopje, Macedonia

[2]Pexim Computers, Atinska 12, 1000 Skopje, Macedonia

{vlatko; liljana}@etf.ukim.edu.mk; lazo@pexim.com.mk

Abstract: WPANs (Wireless Personal Area Networks) are aimed at replacing wires between objects that are close to each other and then hook to the larger world when/if convenient. This wire replacement technology is intended as an embedded connection between a large variety of devices, many with limited capabilities. While the WLAN technologies are specifically designed for devices in and around the office or home, a WPAN device will travel from country to country, be used in cars, airplanes and boats and is truly designed for international use. At the moment, Bluetooth is the only existing radio technology for WPANs. The behavior of Bluetooth devices within WPANs can be evaluated through simulations. There are several Bluetooth simulators created particularly for Bluetooth environment. When considering a simulator it is very important that it matches the reality as close as possible. The performances of the most popular Bluetooth simulators are evaluated in this paper.

Keywords: WPAN, Bluetooth, simulator

1 Introduction

The popularity of wireless devices and applications in the last several years led to adapting solutions from the traditional wired networks into the wireless environment. Deploying and debugging applications on a real network can be rather tiresome, especially if large networks are considered. The process of deploying wireless applications and protocols often requires a step through a simulation phase. This is the reason why a computer based simulation can be very useful in designing, validating and improving applications and protocols. Simulators provide a toolkit that enables metrics collection and various wireless network diagnostics.

The widely accepted network simulator in the academia world is Ns-2 (Network Simulator 2), [1]. Ns-2 is a compromise between performance and ease of use. This is achieved by combining C++ as a core engine and OTcl (an object oriented version of TCL) for configuration and simulation scripts. This simulator is a powerful tool in terms of using it as a standalone simulator. Due to its object oriented nature, users can implement their own Bluetooth extensions that simulate a certain topic of interest. However, by using the Ns-2 simulator as a support for a Bluetooth environment and

installing some Bluetooth simulator over Ns-2, users get a full Bluetooth protocol stack within the Ns-2 environment.

There are several existing Bluetooth simulators that use the resources provided by the Ns-2 platform. They all intend to strictly follow the Bluetooth specifications v1.1, [2]. One of the most popular is *BlueHoc*, [3]. BlueHoc is available for download under IBM Public License (IPL) from IBM developerWorks. *Blueware*, [4], is also an extension to Ns-2. Blueware was developed after BlueHoc and addresses the known bugs in the BlueHoc simulator. It also allows users more Bluetooth functionalities. *UCBT*, [5], is the latest addition in Bluetooth modules for Ns-2. It is more tightly integrated in the Ns-2 environment because the Bluetooth module provided by UCBT appears as a new MAC/Phy layer in the Ns-2 protocol stack. However, UCBT is still under development and does not have all of its intended functionalities activated.

This paper is organized as follows. Section 2 gives a brief overview of the Bluetooth technology and its aspects. Section 3 analyzes the most popular Bluetooth simulation packages available, while section 4 shows some general comments on these packages. Section 5 introduces two simulation scenarios tested on the simulators elaborated in section 3. Section 6 concludes this paper.

2 Bluetooth technology overview

Bluetooth, [2], is a radio technology for Wireless Personal Area Networks (WPANs) operating in the 2.4 GHz ISM band. It is a short range (0m – 10m) wireless link technology aimed at replacing cables that connects phones, laptops, PDAs and other portable devices. The Bluetooth physical layer uses frequency hopping spread spectrum (FHSS) at a rate of 1600 hops/s and Gaussian frequency shift keying (GFSK) modulation. There is a set of 79 hop carriers at a 1 MHz spacing defined. The majority of Bluetooth devices transmit at a power level of about 1 mW (0 dBm) with a raw data rate of 1 Mbit/s.

Two or more units communicating on the same channel form a piconet, where one unit operates as a master and the others act as slaves. There can be up to seven active slaves at the same time. Multiple piconets overlapping in space and time form a scatternet. A device can be a master in one piconet only and a slave in several others at the same time.

The link formation process specified in the Bluetooth baseband specification consists of two processes: *Inquiry* and *Page*. The goal of the inquiry process is for a master node to discover existence of neighboring devices and to collect enough information about the low-level state of those neighbors to allow it to establish a frequency hopping connection with a subset of those neighbors. The goal of the page process is to use the information gathered in during the inquiry process to establish a bi-directional frequency hopping communication channel. The Bluetooth link supports both synchronous services such as voice traffic and asynchronous services such as bursty data traffic. Two physical link types have been defined:

- The synchronous connection-oriented (SCO) link

- The asynchronous connectionless (ACL) link

In the Bluetooth design, special attention has been paid to reduction of power consumption. Therefore, a Bluetooth device can be either in an active mode of operation or in one of the low-power modes of operation (sniff, hold and park mode).

The Bluetooth protocol stack is shown in Figure 1.

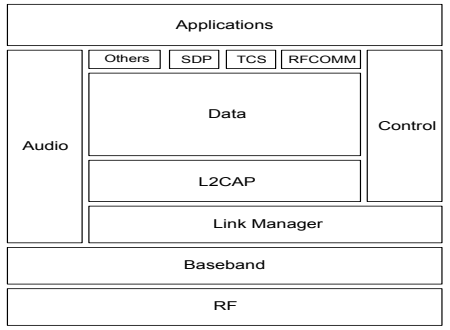

Figure 1: The Bluetooth protocol stack

The RF layer specifies the radio parameters. The baseband layer specifies the lower-level operations at the bit and packet levels (FEC, CRC, ARQ). The link manager layer is responsible for connection establishment and release, authentication, connection and release of SCO and ACL channels, traffic scheduling, link supervision and power management tasks. L2CAP layer is an interface between the standard data transport protocols and the Bluetooth protocol. RFCOMM and TCS (Telephone Control Specification) provide serial cable emulation and a cordless telephony protocol, respectively. SDP is a service discovery protocol which enables a Bluetooth unit to find the capabilities of other Bluetooth units in range. More information on the Bluetooth specifications can be found in [2].

3 WPAN-BT Simulation packages

This section gives an overview of the widely accepted and explored Bluetooth simulation packages and discusses some of their features.

3.1 Bluehoc

Bluehoc is an open-source Bluetooth simulator provided by IBM. There are several available versions of Bluehoc which require different versions of ns-2 to work with. BlueHoc implements the basic features of Bluetooth baseband, Logical Link Control and Adaptation protocol (L2CAP) and Link Manager Protocol (LMP) specifications. There is only support for piconet scenarios, whereas the scatternet scenarios require supplementary extension (bluescat) which has not been tested. A brief description of some of BlueHoc's main features is listed below.

Graphical interface. The main Bluehoc characteristic is its graphical interface which makes the simulator easy to use (requires no TCL knowledge). This interface helps the user create a piconet in a simple manner with just few mouse clicks and entering certain required values for each node such as: Starttime (time when a node is turned on), NumResponses (number of responses required in the inquiry process), Inqtimeout (the maximum time available for the inquiry process, typically 10.24s) and InqScanOffset (time between the activation of a node and the moment this node enters the inquiry procedure).

Device discovery and paging. The procedures of device discovery and paging in Bluehoc are simulated almost exactly as in the specifications. Using the values in the Starttime, NumResponses and Inqtimeout fields, the master carrries out the inquiry procedure. In the same time, using the Starttime and InqScanOffset fields, the slave performs the inquiry scan procedure. The inquiry scan procedure is fixed to 11.25ms (18 slots) every 2.56s. InqScanOffset field marks the time between the starting moment of the slave and the moment it enters the inquiry scan state. The device discovery procedures last until interrupted by the Bluetooth LMP. These situation occurs if one of two conditions are fulfilled, either the master has sufficient number of responses (as indicated in the NumResponses field) or the maximum allowed time for inquiry has expired (indicated in the InqTimeout field).

QoS. QoS mapping and negotiations are permitted on an ACL connection basis. Currently, Bluehoc allows the user to specify the application to be either FTP, Telnet or Voice.

Trace Support. Bluehoc supports two types of trace: Network Animator trace (nam) and Bluetooth Specific Trace (BTTrace). The nam trace contains information about the topology (nodes, links, queues, connectivity) as well as information for packet monitoring. There is no possibility of making interventions in the packet contents. BTTrace gives information about each packet sent during the inquiry and paging procedures and also information about LMP messages and L2CAP signalling. This is a very useful trace for monitoring the throughput and the delay distribution.

Mobility. Bluehoc does not support mobility.

Limitations. The Bluehoc simulator is very limited environment because it strictly follows the inquiry, paging and connection procedures in every simulation. Once the master enters the connection state it never repeats the inquiry procedure again. Therefore, it forces the user to create the piconet at the start of each simulation and latter interventions in the topology are not possible. Also, there is a support for simulating only ACL links and there is no support for multiplexing several L2CAP links over one ACL link.

3.2 Blueware

Blueware is an open-source Bluetooth simulator provided by Godfrey Tan from MIT Laboratory for Computer Sciences. It implements the features of the Bluetooth protocol stack according to the Bluetooth specifications 1.1. Blueware does not provide a graphical interface so it is necessary to write a TCL script for each simulation. Each

device in the Blueware simulator is equipped with several modules through which it discovers and communicates with other devices. Figure 2 shows a comparison between Bluetooth protocol stack and the Blueware modules.

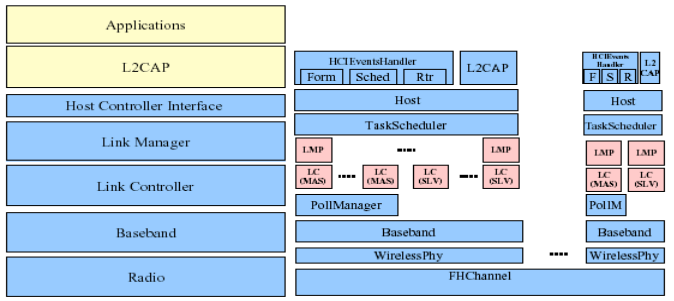

Figure 2: A comparison between Bluetooth stack and Blueware modules

The work on the Blueware simulator is a continuation of the BlueHoc simulator. Blueware authors indicate that their simulator fixes 10000 known bugs in BlueHoc. Moreover, Blueware provides some unique features which are not found in BlueHoc. These features are listed below.

Scatternet formation and Link scheduling. Blueware has a full support for scatternet scenarios. There are two aspects of the scatternet protocols that can be evaluated with Blueware. The user can specify the node arrival model in the scatternet (by using the parameters in the Sim field) and can evaluate the scatternet topology formation delay as well as the power consumption (in terms of how long each node is active). It is also possible to set a mixture of TCP and CBR traffic in order to monitor the throughput, the delay and the power consumption. Blueware currently implements TSF (Tree Scatternet Formation), efficient algorithm for construction of loop-free scatternet topologies, and LCS (Locally Coordinated Scheduling), efficient scheduling protocol for Bluetooth links.

Trace support. Blueware provides extensive trace support. There are 32 trace levels (which are manually set). The standard Bluehoc trace is also supported. The biggest advantage of Blueware is its ability to analyze the simulations which produces very useful information about the topology forming delay, throughput, packet delay between two nodes etc.

Limitations. Blueware does not implement interference model (so the results from the Blueware simulations are the best possible case) and there is no support for SCO links.

3.3 UCBT

Ucbt is an ns2 module, designed by Qihe Wang at the University of Cincinnati, which implements a full Bluetooth stack. It is tightly integrated within ns2 and supports the

latest ns2 versions. Figure 3 shows the functionalities in the protocol stack provided by UCBT.

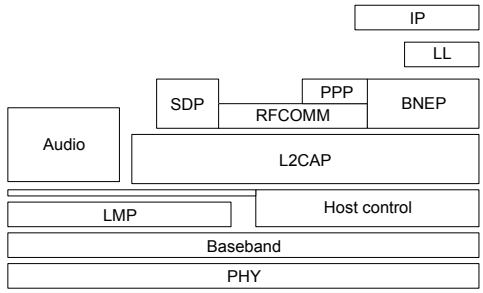

Figure 3: UCBT modules

UCBT has several advantages over the previously mentioned Bluetooth simulators. It supports scheduling algorithms for intra-piconets, supports almost all packet types, provides BNEP component including bridge capability and detects packet collisions between different piconets (BNEP stands for Bluetooth Network Encapsulation Protocol and is used for providing a broadcast segment across a scatternet which enables ordinary IP hosts to be interconnected in a Bluetooth ad-hoc environment). However, UCBT has its disadvantages such as the lack of scatternet scheduling algorithms, problems with the trace format and a need for fine tuning of the ARQ schemes.

UCBT seems to be promising in addressing advanced topics in Bluetooth environments, but is still under development. A complete version of the UCBT module is expected sometime during the next year (2004).

3.4 Other Bluetooth simulation packages

The above-mentioned simulators are not the only available tools for simulating Bluetooth environment. There are two other Bluetooth simulators worth mentioning, however none of them has been tested.

IVT Bluelet. IVT (International Validation and Testing Corporation) Bluetooth protocol stack, [6], is a full implementation of the Bluetooth protocol stack v1.1. It contains all protocols in the Bluetooth host protocol stack including TCS, SDP, RFCOMM, L2CAP and HCI. The code is very reliable, efficient and compact. It is ideal for both embedded applications and desktop applications. By using the formal design technique from Specification Description Language (SDL), IVT validated protocols to assure that all protocols were correct before writing C code. IVT Bluelet supports different Operating Systems (OSs), different Bluetooth chipsets and different HCI transport drivers. IVT Bluelet is a commercial product, however IVT offers the Bluetooth community free license to use Bluelet protocol stack object code in Bluetooth product development.

BlueSim. BlueSim simulator, [7], is used to seamlessly glue the BlueHoc simulator and the ns-based LAP (LAN Access Profile) simulator. LAP defines how Bluetooth

enabled devices can access the services of a LAN using PPP (Point-to-Point Protocol). Bluetooth LAP simulator was devised for exploring the end-to-end impact on IP applications caused by the underlying Bluetooth stack. Since the BlueHoc simulator addresses only the baseband, L2CAP and the LMP layers and the ns-based LAP simulator addresses higher layers, it is necessary to use BlueSim to evaluate the performances of LAP.

4 General remarks on the WPAN-BT simulation packages

All above-mentioned simulators were investigated running different scenarios. The following table presents the characteristics of the examined simulators:

Bluetooth simulator	BlueHoc2.0	Blueware1.0	UCBT*
ns2 version required	ns-2.1b7a	ns-2.1b7a	Supports all the latest versions
Scatternet support	Only with bluescat installed	Yes	Yes
BNEP	No	No	Yes
Trace support	Nam and BTTrace	Bluehoc trace format + analysis	Not fixed by now; there is a trace on all nodes or a trace on a specific node
Limitations	Strict following of the inquiry, paging and connection procedures	No interference model; two much oriented towards TSF and LCS	Still under development; bnep part not finished and scheduling problems at lmp

Table 1: Examined simulators

UCBT is still under development and some of the mentioned features may not be implemented yet. The choice of a particular simulation package depends on the intended simulation scenario.

5 Simulation scenarios

Different simulation scenarios have been tested with the examined simulators. The following subsections present some examples.

5.1 Simulation scenario under BlueHoc

The scenario under BlueHoc was evaluated with version 2.0 of the BlueHoc simulator. As already mentioned, the most powerful feature of the BlueHoc simulator is its graphical interface (Figure 4) which is very user-friendly and requires no TCL knowledge.

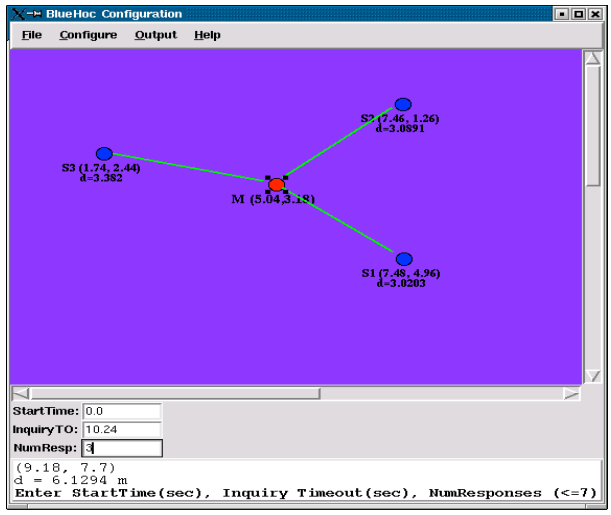

Figure 4: BlueHoc's graphical interface

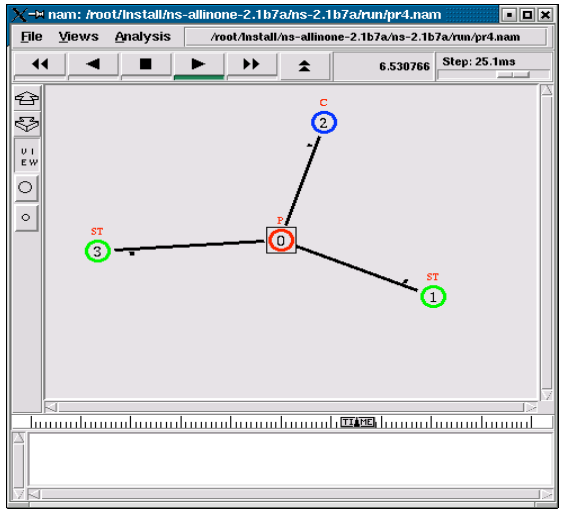

Figure 5: Network Animator

The graphical interface allows the user to create a Bluetooth piconet (one master and up to seven slaves), set the required values for the inquiry procedure and choose the application type on each link. The evaluating scenario in this paper consists of one master and 3 slaves. The trace files supported by BlueHoc are: Nam trace files and BTTrace files. The nam trace files allow a visual representation of the packet flow between the network nodes as well as monitoring of a node's state condition (Figure 5).

The BTTrace file (Figure 6) gives information about each packet sent and received during the inquiry and the paging procedures and also shows LMP and L2CAP signalling messages.

```
appname[0] = Application/FTP
appname[1] = Application/Telnet
appname[2] = Application/Traffic/Exponential
INQ_MSG ****-->1        CLKN: 4113    clock: 1.285580e+00
INQ_MSG AFTER BO *-->1  CLKN: 5073    clock: 1.585580e+00
FHS_PKT 1-->0           CLKN: 5076    clock: 1.586522e+00
INQ_MSG ****-->1        CLKN: 5108    clock: 1.596517e+00
INQ_MSG AFTER BO *-->1  CLKN: 6516    clock: 2.036517e+00
FHS_PKT 1-->0           CLKN: 6519    clock: 2.037460e+00
INQ_MSG ****-->1        CLKN: 6549    clock: 2.046830e+00
INQ_MSG ****-->2        CLKN: 4113    clock: 2.285580e+00
INQ_MSG AFTER BO *-->2  CLKN: 4209    clock: 2.315580e+00
FHS_PKT 2-->0           CLKN: 7412    clock: 2.316523e+00
INQ_MSG ****-->2        CLKN: 4244    clock: 2.326518e+00
INQ_MSG AFTER BO *-->2  CLKN: 4276    clock: 2.336518e+00
FHS_PKT 2-->0           CLKN: 7479    clock: 2.337460e+00
INQ_MSG ****-->2        CLKN: 4309    clock: 2.346830e+00
INQ_MSG AFTER BO *-->1  CLKN: 7637    clock: 2.386830e+00
FHS_PKT 1-->0           CLKN: 7640    clock: 2.387773e+00
INQ_MSG ****-->1        CLKN: 7672    clock: 2.397768e+00
INQ_MSG ****-->3        CLKN: 12341   clock: 5.856830e+00
INQ_MSG ****-->2        CLKN: 15605   clock: 5.876830e+00
INQ_MSG AFTER BO *-->3  CLKN: 13685   clock: 6.276830e+00
FHS_PKT 3-->0           CLKN: 20088   clock: 6.277772e+00 //end inquir.proc.
PAGE_MSG ****-->1       CLKN: 20092   clock: 6.279017e+00
PAGE_ACK 1-->0          CLKN: 20094   clock: 6.279647e+00
MASTER FROZEN           CLKE: 20090
FHS_PKT 0-->1           CLKN: 20097   clock: 6.280580e+00
FHS_ACK 1-->0           CLKN: 20098   clock: 6.280892e+00
POLL_ACK 1-->0          CLK: 20103    clock: 6.282455e+00
LMP_HOST_CONNECT_REQ    AM_ADDR: 1
LMP_ACCEPTED            AM_ADDR: 1    ACCEPTED PDU: LMP_HOST_CONNECTION_REQ
LMP_QOS_REQ             AM_ADDR: 1
LMP_ACCEPTED            AM_ADDR: 1    ACCEPTED PDU: LMP_QOS_REQ
PAGE_MSG ****-->2       CLKN: 28256   clock: 9.830267e+00
PAGE_ACK 2-->0          CLKN: 31458   clock: 9.830897e+00
MASTER FROZEN           CLKE: 28254
FHS_PKT 0-->2           CLKN: 28261   clock: 9.831830e+00
FHS_ACK 2-->0           CLKN: 31462   clock: 9.832142e+00
POLL_ACK 2-->0          CLK: 31467    clock: 9.833705e+00
LMP_HOST_CONNECT_REQ    AM_ADDR: 2
LMP_ACCEPTED            AM_ADDR: 2    ACCEPTED PDU: LMP_HOST_CONNECTION_REQ
```

```
LMP_QOS_REQ             AM_ADDR: 2
LMP_ACCEPTED            AM_ADDR: 2    ACCEPTED PDU: LMP_QOS_REQ
PAGE_MSG ****-->3       CLKN: 32417   clock: 1.213058e+01
PAGE_ACK 3-->0          CLKN: 38819   clock: 1.213121e+01
MASTER FROZEN           CLKE: 32415
FHS_PKT 0-->3           CLKN: 32421   clock: 1.213183e+01
FHS_ACK 3-->0           CLKN: 38822   clock: 1.213214e+01
POLL_ACK 3-->0          CLK: 38827    clock: 1.213370e+01 //end paging proc.
LMP_HOST_CONNECT_REQ    AM_ADDR: 3
LMP_ACCEPTED            AM_ADDR: 3    ACCEPTED PDU: LMP_HOST_CONNECTION_REQ
LMP_QOS_REQ             AM_ADDR: 3
LMP_ACCEPTED            AM_ADDR: 3    ACCEPTED PDU: LMP_QOS_REQ
RECV L2CAP_CONNECT_REQ  CH: 2
RECV L2CAP_CONNECT_RSP  CH: 2
RECV L2CAP_CONNECT_REQ  CH: 0
RECV L2CAP_CONNECT_RSP  CH: 0
L2CA_DATA_WRITE        SIZE: 1000     CID: 2  DEST_IP 1 clock 1.214120e+01
L2CA_DATA_READ         SIZE: 1000     CID: 2
BD_ADDR 1 DELAY 1.062500e-02 SIZE 1000 clock 1.215183e+01
L2CA_DATA_READ         SIZE: 40       CID: 2
BD_ADDR 0 DELAY 1.875000e-03 SIZE 40 clock 1.215370e+01
L2CA_DATA_WRITE        SIZE: 1000     CID: 2  DEST_IP 1 clock 1.215370e+01
RECV L2CAP_CONNECT_REQ  CH: 1
RECV L2CAP_CONNECT_RSP  CH: 1
L2CA_DATA_READ         SIZE: 1000     CID: 2
BD_ADDR 1 DELAY 1.312500e-02 SIZE 1000 clock 1.216683e+01
L2CA_DATA_READ         SIZE: 40       CID: 2
BD_ADDR 0 DELAY 1.875000e-03 SIZE 40 clock 1.216870e+01
L2CA_DATA_WRITE        SIZE: 1000     CID: 2  DEST_IP 1 clock 1.216870e+01
```

Figure 6: BTTrace file

The trace file in Figure 6 gives information about the time spent for the inquiry and the paging procedures. It is evident that the inquiry procedure ends when the LMP in the master gathers the required number of responses from the slaves. The paging procedure is shorter than the inquiry procedure since the master uses the information from the inquiry procedure when paging the slaves. The trace file in Figure 6 also shows that there is another connection definition (logical) at the L2CAP layer because the concept of master-slave communication is no longer valid at the higher layers. The LMP and L2CAP signalling packets are monitored and their size and sending time (global simulation time) is shown in the trace file.

5.2 Simulation scenario under Blueware

When simulating a certain scenario under Blueware it is important to point out the two main aspects of the scatternet protocols that can be evaluated. First, the users can

specify the node arrivals model to evaluate the delays associated with forming a connected scatternet topology and energy consumed (in terms of time nodes spend in the busy mode). Second, a mixture of TCP and CBR flows can be setup to evaluate the throughput, delay and energy consumption. However, unlike BlueHoc, Blueware does not provide a graphical interface for configuring the simulation scenario, so it is necessary to write a TCL script.

A very useful feature of Blueware is the ability to analyze the trace files and, thus, provide aggregated results for the simulation scenario. Figure 7 shows an analysis script of a communication scenario that consists of 20 connected nodes.

```
Node     PktDelay ByteRecv Goodput  ConDelay
0        0.00e+00 0        0.00e+00 2.51e-01
1        0.00e+00 0        0.00e+00 3.03e-01
2        0.00e+00 0        0.00e+00 3.95e-02
…
15       0.00e+00 0        0.00e+00 1.20e+00
16       0.00e+00 0        0.00e+00 4.02e-02
17       0.00e+00 0        0.00e+00 1.25e+00
18       0.00e+00 0        0.00e+00 2.20e-01
19       0.00e+00 0        0.00e+00 2.51e-01
Avg 4.75e-01 StdDev   5.38e-01
SuccRate 100.00 LastLink 10.52 FromLastNode 10.52
…

Node      Life       INQ        INQ_SC     INQ_RE     PGE        PGE_SC     PG_MAR
PG_SLR    STB       CONN
0       2.71e+02   0.00e+00   1.73e-01   1.98e+00   0.00e+00   1.66e-02   0.00e+00
6.60e-03  1.76e+02 (0 ) 1.26e-02 (12) 2.46e+00 (19) 8.98e+01
1       2.71e+02   5.13e+00   1.43e-01   2.76e-02   2.00e-03   2.66e-02   2.40e-03
6.60e-03  2.63e+02 (1 ) 1.76e-02 (8 ) 2.93e+00
2       2.71e+02   9.79e+00   5.32e-02   1.82e-02   4.60e-03   1.46e-02   2.60e-03
7.20e-03  2.56e+02 (2 ) 9.00e-03 (14) 3.16e+00 (16) 2.15e+00
3       2.71e+02   1.72e+00   2.07e-01   5.60e-03   9.60e-03   3.20e-03   2.40e-03
3.60e-03  2.65e+02 (3 ) 8.60e-03 (9 ) 3.58e+00
…
AVG                3.07e+00   1.49e-01   9.66e-01   1.03e-02   8.24e-03   2.79e-03
4.05e-03  2.44e+02  2.28e+01
Avg disc time(%):          1.79     BefScatFormed         100.00   AftScatFormed
-3.01
…
Traffic Stats
src->dst Thruput   Delay    NPkts   Dur   Proto NHops StdDevDelay           // the
throughput between nodes
0->14   2.26e+05   4.61e-01   11025   2.00e+02 tcp 2 1.42e-01            //0 and 14
in a tree topology is measured
Total 2.26e+05   4.61e-01   AvgHops 2.00e+00   NFlows 1
```

Figure 7: An analysis script in Blueware

The script in Figure 7 gives information about the time it takes for a node to establish its first communication link as well as the time used for creating a complete loop-free scatternet topology (with the use of the TSF algorithm). It also shows the time each node spends in every baseband mode and traffic statistics for a certain application.

6 Conclusion

The modelling of WPANs is becoming easier with the simulation packages available for this purpose. Furthermore, the simulation phase is much cheaper than testing on a real network. Also, the widely accepted simulation packages have open-source nature, which makes them very flexible. The mentioned limitations of the elaborated simulation packages can be easily overcome by modifying the source code.

7 References

1. The Network Simulator 2, UCB/LBNL, http://www.isi.edu/nsnam/ns/
2. Bluetooth Specifications Version 1.1, http://www.bluetooth.com
3. BlueHoc Simulator, http://oss.software.ibm.com/developerworks/opensource/bluehoc/
4. Blueware Simulator, http://nms.lcs.mit.edu/projects/blueware/software/
5. UCBT, qwang@ececs.uc.edu
6. IVT Bluelet, http://www.ivtcorporation.com/products/stack/newstack.htm
7. BlueSim Simulator, http://www.harris.cise.ufl.edu/seminars/bluetoothsim.html