# High-Level Design Document for MANET Routing Protocols Benchmarking project

|  |  |
| --- | --- |
| **Author** | Assaf Israel & Eli Nazarov |
| **Version** | V0.5 |
| **Created** | 21-Sep-10 |
| **Last Updated** | 20-Oct-10 |

# Introduction Overview

This document aims to present the High level design as well as the predicted workflow of the application to be developed in the context of the MANET Routing Protocols Benchmarking project.

## Assumptions and Design Constraints List

Since this project aims to test existing MANET routing protocols implementation on windows operated mobile phones we assume the existence of at least 3 different implementations of the following protocols: OLSR, Dymo & ZRP. The implementations must be windows mobile "friendly" in the sense that they are executed as user level modules and do not require any changes to the existing windows network stack.

We also assume that the chosen benchmark (probably IPerf) is open source, since we predict a few modifications will be needed to re-route the benchmark to the routing modules.

Our design aims to be as modular as possible to enable farther testing of more MANET routing protocols and the extraction of different data type statistics.

However as mentioned in the functionality report, our main goal is the testing of the above 3 protocols (which represents different MANET routing protocols classes), with focus on Energy costs.

## Dependencies

Currently we predict the following dependencies:

1. OLSR, Dymo & ZRP implementations – some work will be needed to convert simulator implementations into real working ones. If curtain implementations will prove difficult or excessive time consuming to deal with, we will consider looking for alternative implementations. In extreme case other routing protocols, of the same classifications will be considered as viable alternative for those protocols with troubling implementations.

## Issues List

We’ve identified several issues which will need to be addressed in the development process:

1. Finding suitable JVM on that will work nicely on top of WM devices. Several alternatives are being considered at the moment: J2ME, IBM WebSphere Everyplace Micro Environment, Esmertec Jbed.  
   Some have operational difficulties, other lack in support & documentation. We’ll try to identify the one good for our goals as soon as possible.
2. The protocol implementation is obviously our biggest concern. We’ll need to handle this issue at the earliest stages of development, so we can be curtain we have implementations for all of our intended protocols and that they “work as advertised”.
3. At some point during development, we’ll obviously want to test our application on real devices. We hope the transition from the simulation environment to the actual platform will be smooth as possible.

## Tentative time-tables

|  |  |
| --- | --- |
| **Date** | **Development phase** |
| Oct-Jan 2010/11 | Implementation and testing. We plan to present a short non-formal E2E demo by late January. |
| Fab 2011 | Benchmarking + Final report/results. |

# Logical Architecture

The framework we provide consists of several components:

* Protocol support – The implementation of MANET Routing protocols and their adaptation for our framework.
* Controller (Windows Mobile Application) – The application for the mobile device through which the framework is controlled by the end-user.
* Ipref – Benchmark tool for protocol testing.
* Packet Forwarder – This component is responsible to forward the traffic from Ipref to the protocol (It can be viewed as an adaptor for the Ipref when using the MANET protocols).
* Statistics Collector – Using WM API, collects interesting information during benchmark process that later on can be retrieved and analyzed.

Additional information will be collected directly throw the Ipref.

Packet Forwarder will coordinate the traffic from Ipref to the specific protocol we will test.

We’ll configure the Ipref to work with some loopback port, meaning it will transmit and receive its traffic over a loopback port while the Packet Forwarder will listen on this port and forward Ipref packets to the selected MANET Protocol.

By doing this we hope to enhance the extendibility of the framework and make Ipref independent to the protocol implementation.

The relationship between the components is presented in Figure 2.1

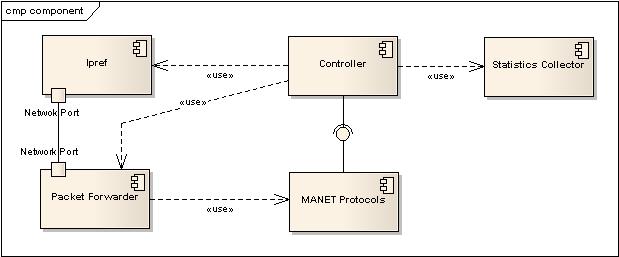


Figure ‎2.1 - Component Diagram

# Design

## Classes

### Class Diagram

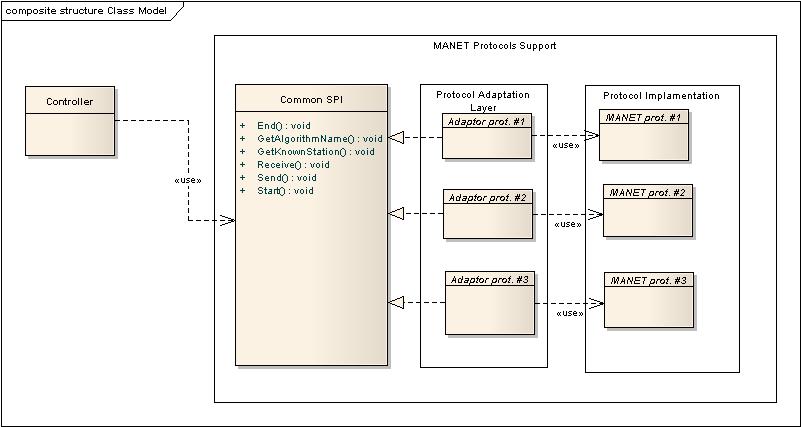


Figure ‎3.1 High level class

### Class details

Controller – Main application through which the framework is controlled.

Common SPI – Common SPI (Service Provider Interface) that all the protocols should implement and support. Using this SPI different protocols are triggered. This should provide high extendibility of the framework. We hope that any new protocol can be used in the framework if it implements the provided SPI.

Adaptor prot.# - Adaptors for each MANET protocol that using the API provided by each protocol implementation that we use will implement the given SPI. When some function from the SPI is triggered the adaptor will translate it to the API of the protocol.

Manet prot. # - The implementation of MANET protocols.

## Flows

Main flow of the framework operation is presented in Figure 3.2.

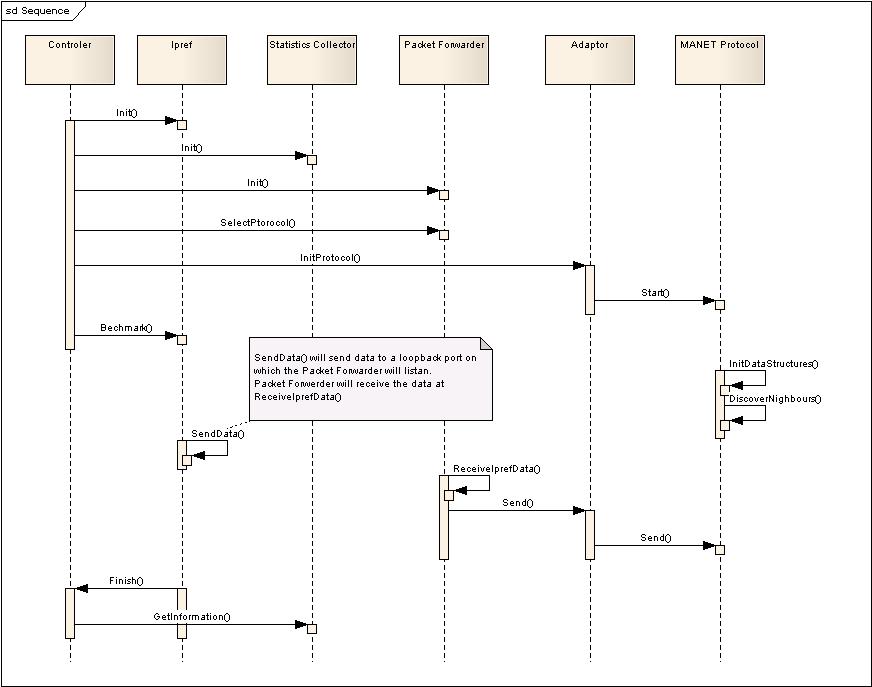


Figure ‎3.2 Framework Flow

## User interface

Main screen will list all of the supported protocols from which the user will select the one he wants to activate. Following selection the system will invoke the selected routing protocols And display statistical information (protocol dependant), e.g. number of neighbors found. From here the user will be able to run the benchmark and view the results at the final screen.

# Use cases

Describe main use cases for your project. For the desired format, refer to slides on Intro to Software Engineering website. You are free, however, to choose any other acceptable format.

|  |  |
| --- | --- |
| Pre-conditions | System is installed on device, and at least one other device has activated the system within a WiFi reception radius of the main device. |
| Trigger | Application invocation by the user |
| Main Success Scenario | 1. The list of supported MANET routing protocols is presented to the user. 2. The user selects its desired protocol. 3. The application starts running the desired protocol. An adaptation period, in which the protocols will stabilized might be impose on the system; during that period the application may look busy (hour glass icon). 4. An option is presented to the user, either become a server side for a benchmark, or a client side[[1]](#footnote-1).    1. In case the user selects to be a server side, the application will start IPerf in server mode. The user may click to end the benchmark at anytime.    2. In case the user selects to run in client mode:       1. The user will be prompt for an IPerf server IP address (that is the server side mobile device).       2. A quick verification of matching protocols invocations, as well as connectivity will be invoked.       3. If successful, the application will start IPerf in client mode with the given IP and Energy saving statistics will start being collected until the benchmark has finished.       4. Benchmark results as well as Energy saving statistics will be presented to the user at the final screen. |
| Alternatives | [\*] – In any step, the user is presented with the option to go back one step, or exit the application.  [4.a] – In case the user clicks to exit the server mode, the application goes back to step 4.  [4.b.ii] – In case the compatibility/connectivity test fails, a notice is presented to the user (going back to 4.a.i) |
| Exceptions | [3] – In case of a protocol error, a notification will be presented to the user, and he may try again (going back to 1).  [4.b.iii] – In case of a benchmark failure, the IPerf error notification is presented to the user (going back to 1). |

# References – to external papers/packages

|  |  |  |
| --- | --- | --- |
| **Title** | **Reference** | **Usage description** |
| IPerf | <http://www.noc.ucf.edu/Tools/Iperf/> | Used as a network benchmark to stress test the routing protocols. |
| Windows Buttery driver support | <http://msdn.microsoft.com/en-us/library/ms923711.aspx> | Used to collect battery statistics |
| Windows Power Management support | <http://msdn.microsoft.com/en-us/library/aa447554.aspx> | Used to collect general power state statistics |
| Comparison of MANET routing protocols using a scaled indoor wireless grid | <http://portal.acm.org/citation.cfm?id=1413939.1413946> | Related Paper |
| Performance Evaluation AODV routing protocol on ad hoc hybrid network testbed using PDAs | <http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?reload=true&arnumber=1635480> | Related Paper |

1. This option might be selected as a command-line parameter to Ipref and not as GUI screen. This will be decided later on. [↑](#footnote-ref-1)