Cooperation using a Robotic Ad Hoc Network made from Bluetooth, JXTA, OSGi and other commercial off the shelf (COTS) products

Kenneth Patrick Robinson
LLB (QUT), BInfoTech (QUT)
School of Information Technology
Faculty of Science and Technology
Master of Information Technology (Research)

2008

A very big thank you to Andy who set me out on this journey

Keywords

Robot soccer, robot communication, wireless networks, peer to peer protocols, software life-cycle management, messaging, skill transfer, code transfer, WiFi, Bluetooth, BlueZ, Linux, Java, OSGi, JXTA

Abstract

Abstract - Mobile devices in the near future will need to collaborate to fulfill their function. Collaboration will be done by communication. We use a real world example of robotic soccer to come up with the necessary structures required for robotic communication. A review of related work is done and it is found no examples come close to providing a RANET.

The robotic ad hoc network (RANET) we suggest uses existing structures pulled from the areas of wireless networks, peer to peer and software life-cycle management. Gaps are found in the existing structures so we describe how to extend some structures to satisfy the design.

The RANET design supports robot cooperation by exchanging messages, discovering needed skills that other robots on the network may possess and the transfer of these skills. The network is built on top of a Bluetooth wireless network and uses JXTA to communicate and transfer skills. OSGi bundles form the skills that can be transferred.

To test the final design a reference implementation is done. Deficiencies in some third party software is found, specifically JXTA and JamVM and GNU Classpath.

Lastly we look at how to fix the deficiencies by porting the JXTA C implementation to the target robotic platform and potentially eliminating the TCP/IP layer, using UDP instead of TCP or using an adaptive TCP/IP stack. We also propose a future areas of investigation; how to seed the configuration for the Personal area network (PAN) Bluetooth protocol extension so a Bluetooth TCP/IP link is more quickly formed and using the STP to allow multi-hop messaging and transfer of skills.

Contents

List of Figures

List of Tables

Acronyms

ARQ Automatic retransmission request

CERO Framework Windows CE Robot framework

COTS commercial off the shelf

FIRA Federation of International Robotsoccer Association

GPRS General packet radio service

GPS Global Positioning System

HTTP Hyper Text Transfer Protocol

IEEE Institute of Electrical and Electronics Engineers

IETF International Engineering Task Force

IP Internet Protocol

IrDA Infrared Data Association

J2EE Java Two Enterprise Edition

J2ME Java Two Micro Edition

JAUS Joint Architecture Unmanned Systems working group

JTRS Joint Tactical Radio System

JXME JXTA Micro Edition

JXSE JXTA Standard Edition

JXTA Juxtapose

JamVM open source Java Virtual Machine

MTU Maximum Transfer Unit

OMG Object Management Group

OSGi Open Services Gateway Initiative

P2P peer-to-peer

PAN Personal area network

PCMCIA Personal Computer Memory Card International Association

QoS Quality of Service

RANET robotic ad hoc network

RTC Robotic Technology Component

SAE the Society of Automotive Engineers

SIMPLE Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions

SIP Session Initiation Protocol

STP Spanning Tree Protocol

TCP Tranmission Control Protocol

UDP User Datagram Protocol

VOIP Voice over IP

WLAN Wireless local area network

XMPP Extensible Messaging and Presence Protcol

Statement of original authorship

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Signature

Date

Chapter 1

BASIC NEED FOR COOPERATION

Chapter Summary - The need for robotic cooperation is stated. Making a RANET from COTS products is put forward as the solution. Lastly, an outline of the thesis structure is given

1.1 Introduction

Cooperating mobile robots need to communicate. Mobility requires that such communication be wireless. Great advances have been made in wireless communication technology in the past two decades, motivated first by mobile telephony and later by wireless computer networking. Short range wireless communication has become ubiquitous and low cost, with several standardized protocols competing in the market. Likewise peer-to-peer (P2P) communication in fixed networks has an established software base.

Therefore, instead of developing a wireless communication framework from scratch, it seems more effective to combine and adapt commercial off the shelf (COTS) components to the specific needs of wireless communication between mobile robots. In this thesis we describe a wireless communication framework for mobile robots, as illustrated in Figure 1.1, built out of COTS components. This framework enables the robots to form mobile ad hoc networks, send and receive messages and discover, transfer, load and remember skills. Cooperation may involve two or more robots. Because of the dynamic nature of

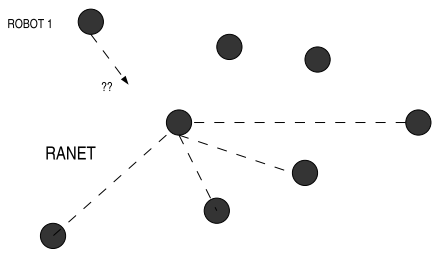


Figure 1.1: Robot wanting to join and communicate with RANET

cooperative work, new robots may join and robots of the group may head off for other tasks. Also, because of obstacles to wireless transmission, communication may drop out for short or long periods of time. A predefined network structure is unsuitable for such conditions; therefore an ad hoc network structure is required. We will call the desired network a RANET.

The objective and scope of this thesis is to design and test a reference implementation of a RANET. Success will be measured by the seeking and transfer of a skill from one robot to another. This will use separate physical devices which will mimic robot functionality. The skill will be the capability to navigate a two-dimensional room in a simulated environment.

This thesis is organized as follows: Chapter 2 covers the abstract structures required for robotic cooperation, Chapter 3 looks at existing structures which may satisfy robotic cooperation, Chapter 4 describes evolution of a design, including any protocols which satisfy robotic cooperation needs, Chapter 5 describes a reference implementation to test the design and some basic tests of that reference implementation and lastly, Chapter 6 gives a summary, highlighting the novel aspects of the design, suggesting some recommendations for improvement and provides observations on the development environment and JXTA.

Please note that the appendices contain non-obvious software documentation largely written by the author that enables others to reproduce the system. They should be read along with the main body of this work.

Chapter 2

STRUCTURES REQUIRED FOR ROBOTIC COOPERATION

Chapter Summary - A real world example of robotic soccer is used to come up with the necessary structures required for robotic cooperation.

2.1 A Real World Example

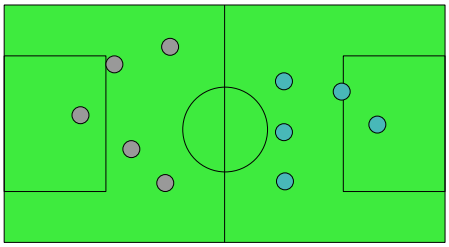


Figure 2.1: Soccer game

We have established a need for a RANET in order for robotic cooperation to occur. What structures are required to make a RANET and thus allow robotic cooperation? An example which requires a RANET must be found. From this example we can better look at what structures are truly required. Robotic soccer shows promise. There are presently two robotic soccer organizations, The RoboCup Federation [25] and Federation of International Robotsoccer Association (FIRA) [26]. Each organization pits robotic teams against each other in a game of soccer. They have found it advances and solves many issues in robotics.

We will analyze robotic soccer in the following sections and look at what robotic cooperation is required. After deriving the robotic cooperation required, we will decide upon the structures needed.

2.2 Structures Required

In teasing out what sort of robotic cooperation is required a general procedure which analyzes user demands is applied [65]. A spreadsheet is created having four columns; DEMANDS, FUNCTIONS, PROCESSES and STRUCTURES (see Table 18 and for an earlier version Appendix A). We look at the users that would be involved in a soccer game; robots, developers, maintainers and operators. From the DEMANDS of these users FUNCTIONS that each robot would require are found. Each FUNCTION is then carried out in a PROCESS using STRUCTURES. DEMANDS, FUNCTIONS, PROCESSES and STRUCTURES are linked to each other in a many to many relationship.

At first when analyzing robotic soccer the DEMANDS, FUNCTIONS and PROCESSES were almost the same. For instance, "discover robotic group" was repeated across all three areas. Gradually the list of FUNCTIONS has been shortened, a number of them now contributing to each user demand. For example, "Passing ball to teammate" now involves "Calculate own position", "Calculate another position" (the teammate) and "Kick ball to calculated position", yet "Kick ball to calculated position" also meets the demand of "Kicking towards goal". PROCESSES have been reduced to making messages, sending messages, managing skills, finding one's own position (localization) and that of others and moving about. The classification of PROCESSES into these categories makes it easier to find satisfactory STRUCTURES. This analysis has benefits as localization was at first completely overlooked. It was only when examining the kicking of the ball that it was realized that calculating the position of oneself and others would be important.

The STRUCTURES derived are presently abstract. We will look at existing structures, narrowing them down to those that satisfy the needs of a RANET in Chapter 3. The design will then evolve through trying out and discarding of existing structures until we end up with the final design in Chapter 4. For the purposes of limiting the problem domain, we have eliminated the localization system and the robot base. Do the following remaining structures satisfy the needs of general purpose robotic communication?

- Single board computer - Most of the user demands require some computing device upon which to execute the PROCESSES. Without some sort of computer, messages could not be made and messages and skills sent or received. A single board computer is a computer built on one circuit board. This departs from the traditional structure of desktops where some of the functionality such as graphics, ethernet and signal processing reside in cards which plug into the motherboard. A single board computer is not strictly required but to satisfy general robotic communication needs size is a factor. Single board computers are the right size for a communications module. It could plug into an existing robot or potentially be also the main processing module.

- Wireless transceiver - A wireless transceiver allows data to be transmitted and received using electro-magnetic waves. This is needed for sending messages and transferring skills. Other means of sending messages, by saying the message out loud for example, are impractical for a robotic device compared to the the ease of sending it wirelessly.

- Operating system - An operating system is required for basic functionality such as a file system, control of wireless transceivers and non-volatile storage.

- Peer to peer protocol - A predefined network structure for mobile robots is unsuitable. To obtain an appropriate ad hoc networking structure for robotic communication, a peer to peer protocol is preferred. This is so robots can share the

responsibility; each robot providing skills to other robots as well as seeking them itself [48]. This gives robustness and redundancy to the RANET. Robots holding a skill can now leave without that skill disappearing. Members can request the skill from more than one place, the RANET sharing the load amongst its members.

- Skill packager - A skill packager is some mechanism by which a skill can be bundled together. Without some agreed way to bundle a skill, there would be no common way to transfer a skill and then load it.
- Non-volatile storage - This is typically flash or hard disk storage. Without such persistent storage the skills transferred could not be remembered.
- Dynamic skill loader - After skills are transferred they need to loaded into the robot. Without such loading no skills could be loaded and used.

It is submitted that the above structures are required. Together they satisfy the demands imposed by robotic soccer.

DEMANDS	FUNCTIONS	PROCESSES	STRUCTURES
Forming a soccer team	Group Functions	Make Join Group query message	robot base
Discover a robotic group	Discover other robots	Make Join Group response message	single board computer
Join a robotic group	Make group/Join group	Make discovery query message	wireless transceiver
Discover other robots skills	Discover skills	Make discovery response message	operating system
Sort skills in group	Sort skills	Make skill query message	peer to peer protocol
Transfer skills	Transfer skills	Make skill response message	skill packager
Put robot in soccer position	Give robot soccer position	Make other robot position message	non-volatile storage
Having a game	Individual Functions	Make own position message	dynamic skill loader
Kickoff	Calculate own position	Make soccer ball position message	localization system
Defending	State position to others	Make log message	
Dribbling	Calculate another position	Make group information message	
Passing ball to teammate	Move to calculated position	Sort skills	
Kicking towards goal	Kick ball to calculated position	Calculate own position	
Giving status to rest of team	Get ball	Calculate another position (ball, other robot)	
Collaborating	Log Actions	Move from present to new position	
Developing/Maintaining	Distribute information to group	Kick ball from present to new position	
Development		Get ball (involves trapping ball)	
Testing		Translate message	
Debugging		Send message	
Updating Capability		Receive message	
Logging Activity		Log Actions	
Obtain Diagnostics		Transfer skill	
		Load skill	
		Delete skill	

Table 2.1: Robotic Soccer

Chapter 3

EXISTING STRUCTURES

Chapter Summary - After finding what abstract structures are needed we search for concrete existing structures that would match. There are few examples of COTS that come close to a RANET except in the military domain. Robotic platforms, wireless networks, wired peer to peer implementations and software lifecycle management COTS exist. The RANET we suggest uses structures pulled from the areas of wireless networks, peer to peer networks and software lifecycle management.

3.1 Finding Concrete Existing Structures

In the previous chapter we decided upon some abstract structures which satisfied the needs of a Robotic Ad-hoc network. They were:

- a single board computer;
- a wireless transceiver;
- an operating system;
- a peer to peer protocol;
- a skill packager;
- a non-volatile storage; and
- a dynamic skill loader.

We now have to look at what concrete existing structures are available which match or closely match these abstract structures. We will attempt to find COTS products which satisfy more than one of these structures.

We will first look generally at robotic ad hoc networks and see if there exists a robotic system which has all these structures. Finding nothing, we will then look at robotic platforms to see if they satisfy our requirements. In the same way we also look at wireless communications, peer to peer protocols and ways of transfering skills and remembering them. Lastly we give a critique of each area, explaining how it meets or does not meet our needs.

3.2 Robotic ad hoc networks and their RANET-like kin

3.2.1 What exists

Numerous wireless robot communication systems have been built in the past, with the majority purpose built from the bottom up [30, page 210], or at the most, only using off-the-shelf wireless transceivers [70]. Sensor networks are the latest addition, with TinyOS being the most prevalent type of sensor network within the research community. TinyOS is an operating system designed for wireless sensor networks capable of running on a number of purpose built hardware platforms [71]. Some work has been done on incorporating the TinyOS platform into mobile sensor networks; essentially a type of robotic wireless network. CotsBots [9], Robomote [22] and MICAbot [44] involve connecting a hardware platform running TinyOS to a robot base. These mobile sensor networks capture information and send it back to a monitoring computer.

There is another class of research robots called flockbots [29], which are built from standard components and communicate using Bluetooth.

Recently Bug Labs has launched a product called BUG, essentially a base module and and modules which plug on to the base in the way Lego does. The base module has various I/O along with an ARM processor and WiFi. It and the other modules run instances of OSGi (see Section 3.6.1) in the Java Mobile Edition (ME) environment, a Java virtual machine and Java libraries using version 1.4.2 of the Java programming language. OSGi service bundles are registered and can be used by the other modules [16].

The Windows CE Robot framework (CERO Framework), has single board computers that control the Lego Mindstorm actuators [30]. It implements a Bluetooth scatternet protocol over 433MHz transceivers [30, page 210].

JXTA running over Bluetooth has been used in the context of a mobile phone doing a handover of a Hyper Text Transfer Protocol (HTTP) connection from Bluetooth to General packet radio service (GPRS) [1].

There is a proposal to have JXTA use Web Services, but there is no attempted implementation [4]. JMobiPeer is a proposed design which has been done on wired personal computers but not wirelessley on the target mobile hardware [11].

The Joint Tactical Radio System (JTRS) is a peer to peer short wave radio system promulgated by the United States military [67].

3.2.2 An evaluation

None of the mentioned robotic systems have all the structures required (See Table 3.1). CotsBots, Robomote and MICAbot do not have the peer to peer protocol, skill packager and dynamic skill loader required. They tend not to communicate with each other and do not form an ad hoc network. Further, skills are intended for the human observor and not other robots. The flockbots again do not have a peer to peer protocol, skill packager and dynamic skill loader. The BUG modules are connected using wires not wirelessley and do not have a peer to peer protocol. The CERO Framework, mobile handover of a HTTP connection and JTRS have no skill packager, dynamic skill loader or non-volatile storage.

3.3 Robot platforms

Having found no existing robotic ad hoc network which satisfies our need we look now to robot platforms which may meet our requirements.

Product	Single board computer	Wireless transceiver	operating system	Peer to peer protocol	Skill packager	Non-volatile storage	Dynamic skill loader
CotsBots	✓	✓	✓			✓	
Robomote	✓	✓	✓			✓	
MICAbot	✓	✓	✓			✓	
flockbots	✓	✓	✓			✓	
BUG	✓		✓		✓	✓	✓
CERO Framework	✓	✓	✓	✓		✓	
mobile handover of a HTTP connection	✓	✓	✓	✓		✓	
JTRS	✓	✓	✓	✓			

Table 3.1: Existing COTS Products

3.3.1 What exists

Robot platforms can be further divided into the standards upon which robot platforms are based and implementations with only a few implementations of robotic standards.

Standards

There are a few bodies trying to define robotic standards; Joint Architecture Unmanned Systems working group (JAUS), the Society of Automotive Engineers (SAE) and Object Management Group (OMG) Robotics Domain Task Force.

JAUS has put out a standard by the same name. It has been expanded on by the SAE and has been given the name SAE AS-4. SAE has a slew of other standards regarding components and safety. OMG has the Robotic Technology Component (RTC) Specification. All of these standards call out Institute of Electrical and Electronics Engineers (IEEE) standards.

The way these standards inter-operate is the SAE and IEEE standards can be used for basic components and safety standards. JAUS (SAE AS-4) can be used for the message protocol between individual components and other robots The OMG robotic standard can be used along with JAUS for management of robotic systems [68]. JAUS seems to have more implementations than the OMG RTC specification.

For communciaton in Extensible Markup Language (XML) one could use SensorML which allows for discovery and control of sensors [5] or RoboML [58] (no new versions since 2003).

There is a need for standardization. Without standard interfaces and communication protocols, computer manufacturers will be forever rolling bespoke components [73]. We try in our architecture to use standards wherever we can to form the RANET design.

Implementations

There are many robotic software implementations [66] (see Table on page 23). As can be seen only a few implementations follow any standards (OpenJAUS and OpenRTM-aist).

3.3.2 An evaluation

All of the above platforms use a client server model, where the client needs prior knowledge of the method calls of the service provider with no dynamic discovery of services provided. They tend to be used between robot user and robot and not as a dynamic communications medium for robots to form groups, transfer skills and send and receive messages as in our proposed RANET. In the main they lack the peer to peer protocol, skill packager and dynamic skill loader required for a RANET.

3.4 Wireless Communication

Having found no existing robot platforms which satisfies our needs we will now look generally at products which may meet the functionality required of the individual structures. We will defer the choice of a single board computer, an operating system and a non-volatile storage as these are less important and may be dictated by our other choices. We look first at wireless transceivers.

3.4.1 What exists

Wireless communication was at first unrelated to computing. It began with broadcast analog radio, moving onto two way radio in closed groups, then to directed one way pagers. Data was then able to be transmitted over analog radio, satellite telephones were introduced and the rise of cellular telephones with data transmission capability occurred. In the military domain link 11 and link 16 protocols were used during the 1990s [52].

Wireless computer networks have come to the fore. Wireless local area networks (WLANs) are among the most popular. These are typically areas covering approximately 100m, with WiFi (IEEE 802.11x) predominantly used.

PANs have supplemented these WLANs. They are usually have a range of 10 metres, with Bluetooth (IEEE 802.15) prevalent. Bluetooth allows for PANs called piconets consisting of one master who controls up to seven slaves. These piconets can be connected together into a scatternet by either a slave or a master participating in two or more piconets [38]. Some studies of Bluetooth have centred on scatternet protocols and in particular the simulation of these scatternets [49]. The most relevant is a piconet only protocol. In this protocol, all devices in a potential piconet elect the most connected device to be master [35]. This protocol requires a period where all devices share information as to which is the most connected. Only recently have some scatternet implementations been acheived [10] [28].

Sensor networks are the latest popular wireless network, with Zigbee (IEEE 802.15.4) becoming the standard. Zigbee can cover vast areas, have low data rates, use little power and has the capacity for 65000 nodes in a network [6].

Lastly there is infrared with technologies like Infrared Data Association (IrDA) with data rates between 2.4Kbps and 14Mbps [31].

Platform	Description
ERSP	From Evolution Robotics. It is a proprietary offering, having a visual pattern recognition system, a simultaneous localization and mapping system and can use a graphical development kit for describing behaviours.
Microsoft Robotics Studio	It has a graphical development kit (it calls this a visual programming language) and a simulation environment.
OROCOS	It is written in C++ and provides extensive libraries for motion and control but no simulation libraries.
URBI	This is a robot special purpose scripting languague with parallelism. It has voice recognition, voice synthesis, face detection, face recognition, SLAM, color blob detection and SIFT based object recognition modules.
Player/Stage/Gazebo	This offers control of hardware through a Tranmission Control Protocol (TCP)/Internet Protocol (IP) server using Player in 2D with Stage and 3D with Gazebo.
OpenJAUS	This is an open source version of the Joint Architecture for Unmanned Systems JAUS.
CLARAty	CLARAty is a software platform developed by Jet Propulsion Laboratories with various algorithms on position estimation, navigation, path planning, localization, etc. It suffers from a licence allowing use of its software only for non-commercial activities.
OpenRTM-aist	This is an implementation of parts of the OMG RTC specification [42].

Table 3.2: Robotic Platforms

3.4.2 An evaluation

Zigbee is not as freely available and as low cost as Bluetooth and WiFi. For that reason it has been eliminated from further consideration. Because IrDA requires line of sight to communicate with another device it is also not suitable for a RANET. Bluetooth and WiFi remain the only two contenders.

3.5 Peer to Peer Protocols

Next we look generally at peer to peer protocols; covering first the history of peer to peer communication before looking at some peer to peer standards and some implementations. An evaluaton is then done of those implementations.

3.5.1 What exists

Historically, peer to peer implementations have occurred on wired networks. Wired communication topologies originally were very centralized, but have become more distributed.

Centralized communication occurred in the 1960s between mainframes and dumb terminals; communication being at the discretion of the mainframe [62]. With the rise of the personal computer (PC) in the late 1970s and early 1980s a shift in communication occurred, going to a client-server relationship. Communication was at the discretion of the client PC. If no requests were made of a server no communication occurred. From about 2000 onwards we have seen the rise of networks where peer to peer relationships exists. As a PC is both a client and a server, all the PCs in a peer to peer network have the same discretion to communicate.

It is not true to say that no peer to peer implementations existed prior to the late 1990s. One of the earliest peer to peer implementations was Usenet, launched in 1979, which allowed for posting of articles to servers which would then push out the updates to other Usenet servers [76]. It is true to say that peer to peer has become more prevalent, with many P2P implemenations and some standards.

Standards

There is presently no working group set up for peer to peer implementations apart from one for Session Initiation Protocol (SIP) (used predominantly for Voice over IP (VOIP)) [15]. This group does separate the SIP layer from the P2P layer. Further they have published an internet draft for this layer [8], refering to P2P implementations of Bamboo, Chord, Pastry, Kademlia, Gnutella, and Gia.

The International Engineering Task Force (IETF) has a section called Internet Research Task Force. This has research groups covering various areas, including the Peer to Peer Research Group. It has published various preliminary documents, including P2P in mobile environments [2] and internet chat protocols.

BEEP is a framework to build any sort of protocol over TCP/IP, using XML in its channel initiation profile. It has a published standard (RFC 3080) and implementations done in C, C++, Python and Java. BEEP is not a peer to peer protocol but has all the necessary tools to make a peer to peer protocol [13].

Implementations

P2P implementations can be classified in two different ways. The first is upon their degree of structure. The second is upon what they do; file transfer, messaging or distributed

computing like distributed databases [47, page 26].

When a node joins a structured network, typically they are slotted into some predefined structure like a binary tree or a hash table. This has some advantages; for instance, it allows for access to a node in a limited number of hops by another node within the network. It has disadvantages in that it is not as fluid as unstructured networks.

With an unstructured network their is no predefined structure. Their is no guarantee on the number of hops for a node to reach another node in the network. There is constrained flooding of the network and the topology is haphazard.

What we call "inbetweeners" exhibit some structure. They typically have a start topology with some super peers exchanging information and lesser peers attached to these super peers (see Figure 3.1).

We explore structured networks, unstructured networks and inbetweeners below.

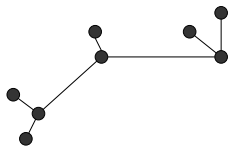


Figure 3.1: Star topology exhibited in "inbetweener" networks

Structured Networks

Strucutured Networks include Bamboo, Chord, Pastry [59], P-Grid [60], Kademlia, Gia and P2PS (running on Mozart/Oz) [45]. Many of these use distributed hash tables which mainly index and retrieve data on a series of peers; essentially distributed databases. Each peer pushes the hash tables to other peers so they can accurately search the database. These peer to peer protocols have mostly been used for file transfer and data storage. They tend to have each node only knowing about some other nodes at the time of insertion into the network.

Unstructured Networks

A file transfer protocol, GNUtella has a published protocol specification and has been used in research before [7]. Wired implementations of GNUtella include LimeWire (using GNUtella). Ad hoc networks like GNUtella flood the network with a query and have directed responses and file transfers.

Inbetweeners

Peer to peer protocols that sit between structured and unstructured networks can be further divided in to those that deal with file transfers, those that deal with messaging and those that deal with both.

For file transfer one can look at Bittorrent; a file transfer protocol implemented in many programming languagues and having an official standard [19].

For messaging internet chat protcols are in the main. They include Jabber and Simple.

Jabber or Extensible Messaging and Presence Protcol (XMPP) is used for internet chat. It has the advantage of being formalised as standards RFC 3920 and RFC 3921 by the IETF. It operates by sending an XML message to its Jabber server, who forwards

it on to a remote Jabber server, who lastly hands it onto the intended recipient. The interesting thing is it has many extensions to the basic protocol including file transfer and remote procedure calls. The disadvantage is most of these are in the draft, experimental or proposed stage and there is no reference implementation covering all these extensions [77].

Another instant messaging protocol is Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions (SIMPLE), an extension to the session intiation protocol. It allows for instant messaging and file transfer. It has known implementations in Microsoft Office Communicator (the client) and Microsoft Office Communications Server (the server which does the peer to peer communication) [46].

For protocols that do both there is JXTA and P2PS.

JXTA comes from Project JXTA, an open source project which provides a platform for peer to peer implementations and runs on a number of platforms (Windows, Linux and Unix). Using super peers, JXTA builds up a picture of nodes in a network over time, both flooding the network with a query and asking known super peers the query. Directed responses, file transfers and messaging occur.

P2PS is a refactoring of the paradigms in JXTA to present a lightweight alternative to JXTA [75]. It is similar in functionality in all respects to JXTA.

3.5.2 An evaluaton

Only really JXTA, P2PS and GNUtella satisfy the needs of robots in a RANET. They allow nodes to join at any time, allow for nodes to leave the network ungracefully without breaking the structure and are readily available as libraries to incorporate into an application. Further they allow for messaging between nodes and tranmission of skills between nodes.

None of the peer to peer protocols have been designed with wireless transmission in mind [47, p. 2]. All have assumed wired connections. None of them are particularly robust. This is in part due to the fact that they do not need to be. Most of the protocols are transmitted via the application layer of the the TCP/IP stack[1]. The TCP protocol ensures that all packets are delivered for P2P protocols. However certain Quality of Service issues may prevent TCP being used for wireless P2P networks (see page 36).

3.6 Skill packager and skill loader

Lastly we look at skill packagers and loaders before again giving an evaluation. For a RANET to work there must be a way of packaging skills and then loading them. A robot must also persist those skills. In effect one must figure out how to package a skill in a loadable lump, send it in some way and then load and remember that skill.

[1]The TCP/IP stack is divided up into the application, transport, internet, network and hardware layers. The application layer is where protocols and applications access the transport layer below. The transport layer is where the TCP and User Datagram Protocol (UDP) protocol resides. TCP provides a connection-orientated reliable transport. UDP provides a connectionless unreliable transport. The internet layer is where the IP resides. IP provides the basis for for connectionless, best-effort delivery of packets. The TCP/UDP packets are wrapped in these IP packets. The network layer has the Media Access Control protocols, which is the generic name for the low-level hardware protocols used to access the physical network. The hardware layer represents the electrical signals used to transmit the MAC protocols.

3.6.1 What exists

Packaging the skill

Packaging the skill in reality is packaging code to be executed on some remote machine. In C and C++ it would be using some existing library. For convenience these are using tarred up and compressed. In Java one would use an existing jar file. In Python one would use a module (generally a file which is the equivalent of a libary).

Transmitting the packaged skill

Transmitting the packaged skill in C/C++ would involve serializing the library. This is not built into the language and would require some work; essentially writing sending and receiving code which would make the transfer happen. Java has the capability of serializing easily when it comes to a class but not in respect of a jar file where more work similar to C/C++ is required. Python can easily serialize a file which represents a module via the act of pickling. Again some sending and receiving code would be required to allow transfer.

Loading the skill

Loading the skill can be done in many languages using the concept of dynamic loading. With dynamic loading at runtime libraries of code are loaded and unloaded at the behest of the calling client code. The question then becomes, how can the application remember what skills have been dynamically loaded and whether those skills are active.

Remembering the skill

To see how a skill is remembered we must first look at the distinction between program code and program state.

By way of example, Java code is made up of an application. The program code is the executables that make up that application. The program state is the objects that are created by the code and reside in memory. When the program is halted those objects die. Nothing about the program state is remembered. This is true of most of the other popular programming languagues like C, C++, Python and Ruby. The program code is separate from the program state.

In other lesser known programming languagues like Lisp and Smalltalk there is no real distinction between program code and program state. When an application, more commonly called the image, is saved both program code and program state can be preserved [61, page 477] [12]. In fact this is often the way applications are distributed with Lisp and Smalltalk.

What needs to be remembered for our purposes the application knowing what skills are available and are active. A skill then is a combination of program code and program state. The saved program code remembers the skill. The saved program state remembers whether the skill is available and whether it is active or inactive.

Remembering the program code and program state can be done in a variety of ways. In image based programming languagues like Lisp and Smalltalk the skill along with its state is remembered. With virtualization of operating systems work has been done on remembering the state of the operating system and transfering mini operating systems as skills with state preserved [56]. This is only in the preliminary stages.

In the absence of such image persistence the program code is generally remembered. The program state could be remembered by writing new configuration files, writing new

code which it links to (not a usual thing) or update tables in a database with textual values or binary object instances (as in Java Two Enterprise Edition (J2EE) with enterprise Java beans or in Python with pickled objects).

One extension to the Java model deserves mention, a specification for a framework written in Java called OSGi. It allows for "an open service platform for the delivery and management of multiple applications and services to all types of networked devices in home, vehicle, mobile and other environments" [53, FAQ Qu.1]. Several implementations have been done in Java (see Knopflerfish as an example [40]). With OSGi one enters a profile when beginning the framework. This profile maps to a program state. Each skill, more properly known as a bundle, can be installed, updated, started, stopped and deleted. When the framework is stopped and restarted the framework remembers that the profile has certain bundles installed and their state (started, stopped, etc.). Whilst not as fine grained as a Smalltalk or Lisp image it is nevertheless enough for learning and remembering.

3.6.2 An evaluation

When it comes to all the things required in respect of skills - packaging, transmitting, loading and remembering - C/C++ is not preferred. Although it can package and load skills it cannot easily transmit and remember them. Java, with the addition of OSGi, can package a skill, load it and remember it. It cannot transmit those skills. This is done, however, with some degree of complexity. Python cannot transmit them but it can package a skill, load it and remember it. It does this with a degree of simplicity. For this reason it is preferred over OSGi.

3.7 Summary

We determined there were no existing implementations of robotic systems which had all the necessary structures to make a RANET. We then looked at individual robots which had most of the structures of a RANET. Again nothing was found. We then covered the areas of wireless communication, peer to peer protocols and skill packagers and loaders to see if existing structures could be found which we could then put together to form a RANET, deferring the choice of a single board computer, operating system and non-volatile storage until later.

For the wireless transceiver we narrowed our choice to Bluetooth and WiFi. The peer to peer protocol was narrowed to JXTA, P2PS and GNUtella. For the skill packager and skill loader our choice is now limited to Python and OSGi with Python perferred.

In the next chapter we will look at the evolution of our design based upon these initial choices.

Chapter 4

STRUCTURES IN A RANET

Chapter Summary - We decide upon the structures in the RANET. This is done by first referring to the structures required for robotic collaboration for a RANET from Chapter 2. Over time we evolve the design, using existing structures from Chapter 3 and where gaps occur, extending some existing structures.

4.1 Evolution of the Design

In Chapter 2 we decided upon the structures required for collaboration. They are:

- robot base
- single board computer
- wireless transceiver
- operating system
- peer to peer protocol
- skill packager
- non-volatile storage
- dynamic skill loader
- localization system

To limit the problem domain the robot base and a localization system have been excluded (See Item 1 of Figure 4.1). The robot base allows movement. It typically has its own micro-controller which another single board computer would exercise to move the robot about. The localization system would be a thesis in itself. For our purposes it is enough if one robot is in contact with another, with no need to calculate one's own position or the position of another.

In Chapter 3 we have explored general COTS which did not have all the necessary structures to make a RANET before looking at individual COTS products which may have some of the structures required. We narrowed our choices for some of the structures as follows:

- a single board computer - deferred;
- a wireless transceiver - Bluetooth and WiFi;

- an operating system - deferred;
- a peer to peer protocol - JXTA, P2PS and GNUtella;
- a skill packager - Python and OSGi with Python preferred;
- a non-volatile storage - deferred; and
- a dynamic skill loader - Python and OSGi with Python preferred.

We now explore further concrete components which will satisfy the structure of a RANET. They can be third party COTS products, extensions we make to third party COTS components or components we make. This is not a strictly paper exercise. Each component has been tried and a reference implementation has been done of the RANET design.

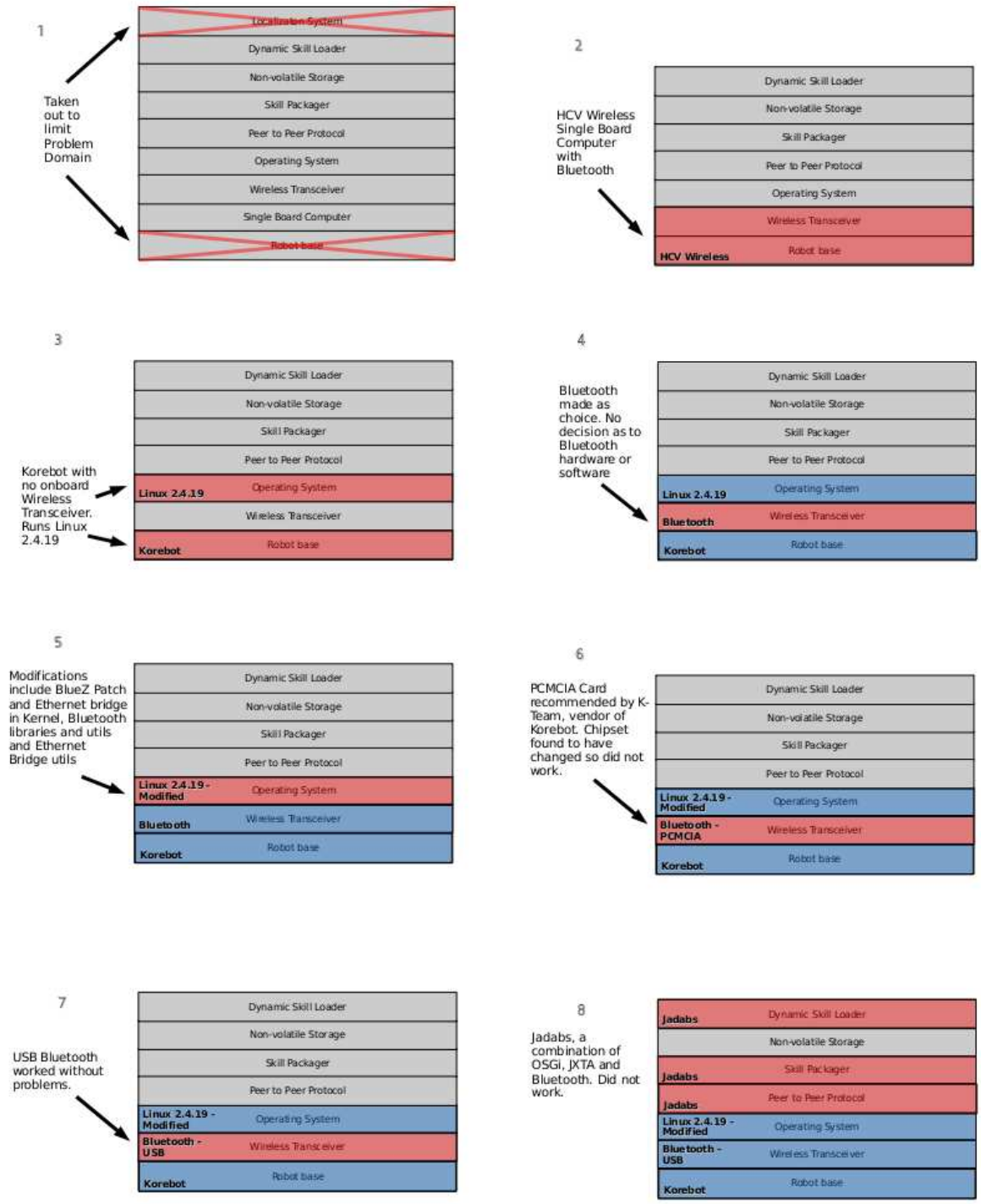


Figure 4.1: Evolution of the Design - covers choice of single board computer (Items 1-3), wireless transceiver (Items 4-7) and start of peer to peer protocol (Item 8)

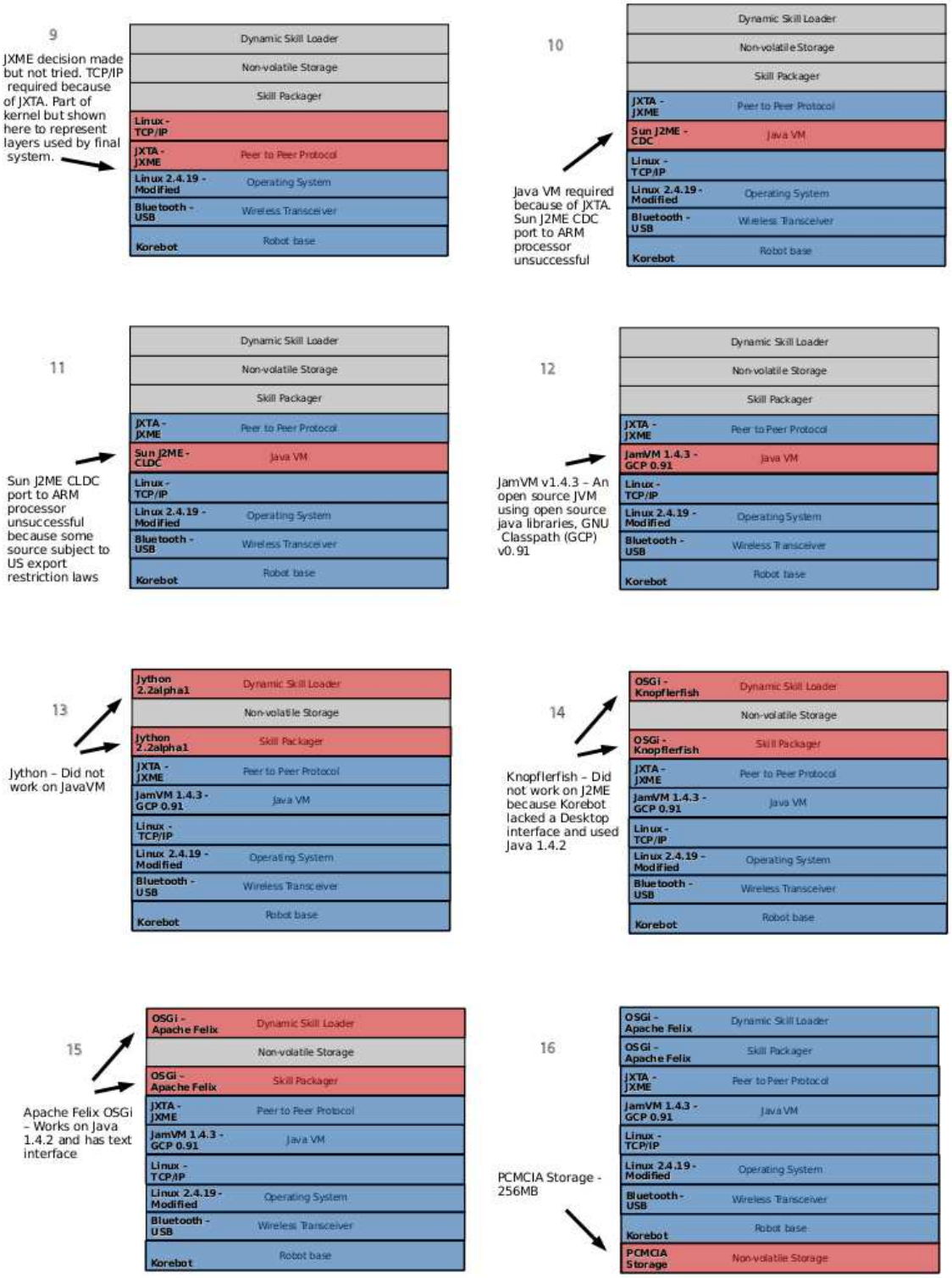


Figure 4.2: Evolution of the Design (cont.) - includes balance of choice for peer to peer protocol (Item 9), Java virtual machine (Items 10-12), skill packager/dynamic skill loader (Items 13-15) and non-volatile storage (Item 16)

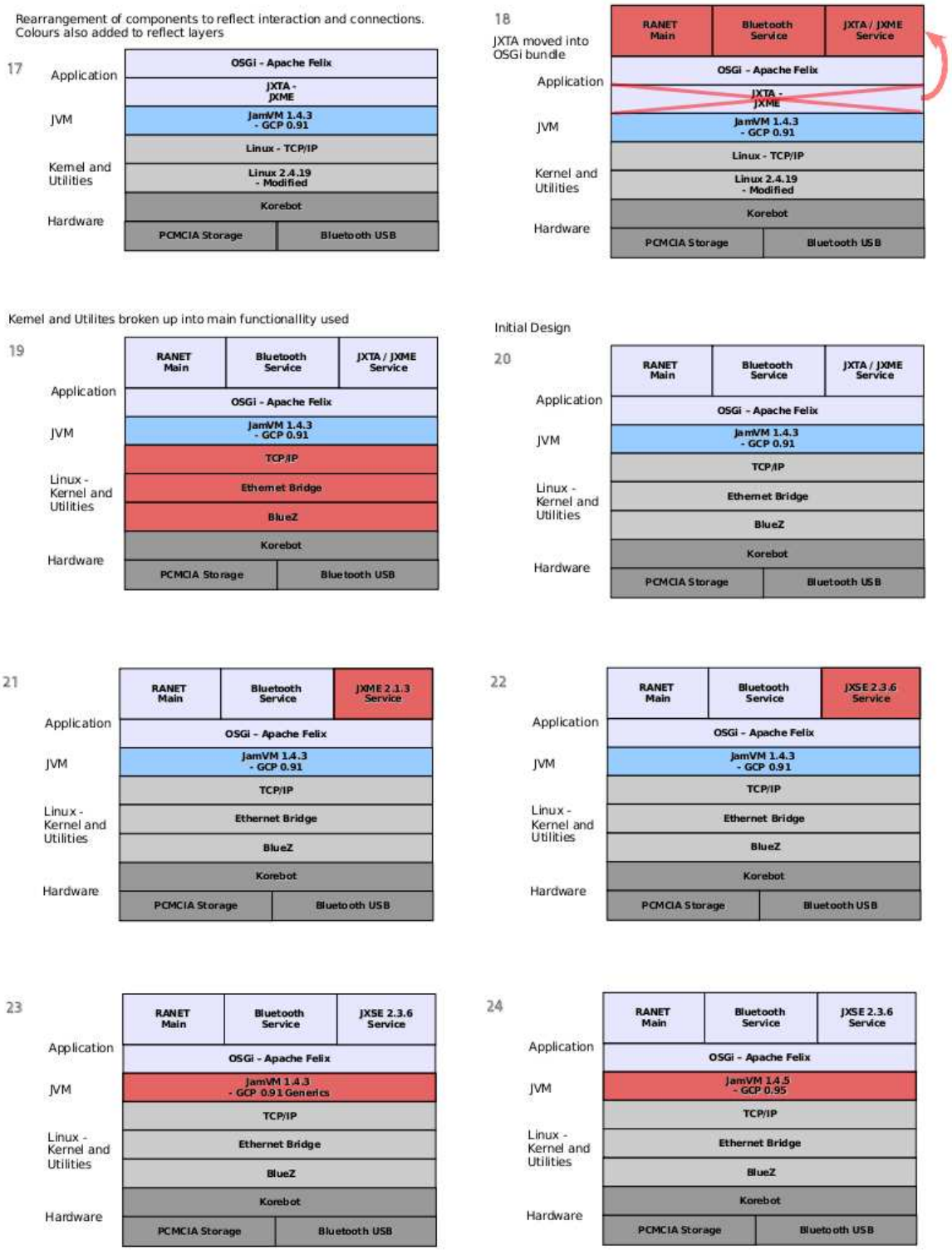


Figure 4.3: Evolution of the Design (cont.) - includes use of OSGi bundles (Item 18), breakup of kernel/utilities into main functionality (Item 19), initial design (Item 20), JXME 2.1.3 and JXSE 2.3.6 failures (Items 21-22), JamVM 1.4.3/GCP 0.91 upgrade which lacked JSR166 concurrency (Item 23) and JamVM 1.4.5/GCP 0.95 upgrade which did not work on ARM processor (Item 24)

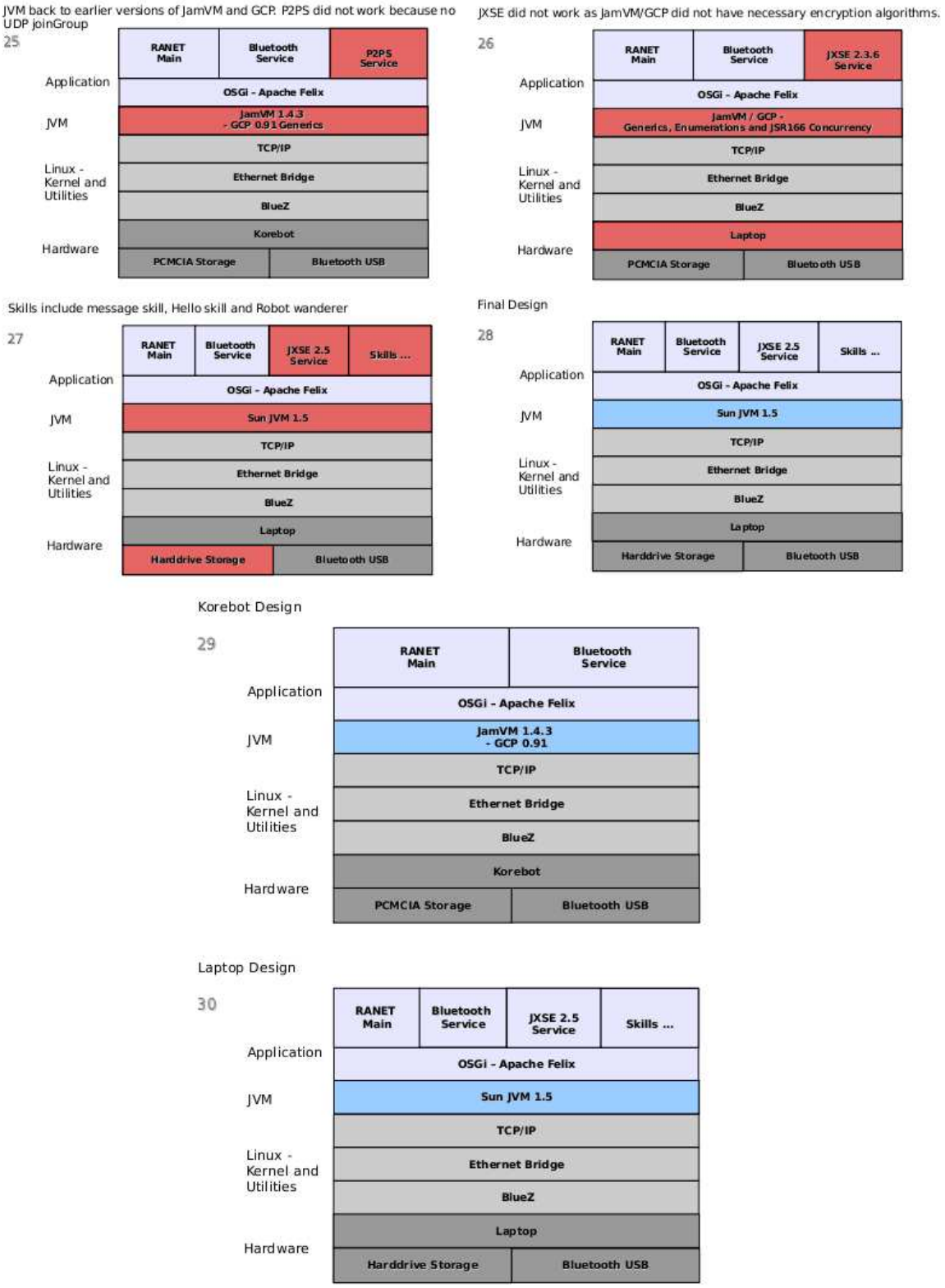


Figure 4.4: Evolution of the Design (cont.) - includes P2PS failure because of absence of UDP multi-cast functionality (Item 25), JXSE 2.3.6 failure because of lack of encryption algorithms (Item 26) and final success with Sun Virtual Machine1.5.0 (Item 27)

4.1.1 Single board computer

It may appear that choosing a single board computer is a self-contained choice. In truth it affects and is affected by other components that are chosen.

For example the choice of wireless transceiver affects the single board computer choice. The only two cheap and readily available wireless transceivers are Bluetooth and WiFi. As we have chosen Bluetooth (see Section 4.1.2 as to why) some reliable freely available Bluetooth software is required. There are some proprietary choices but the only viable open source choice is BlueZ. This runs only on Linux. So the choice of the wireless transceiver makes us choose Linux as the operating system. This eliminates many micro controllers that could otherwise be used. Further it eliminates many larger fully featured single board computers which only run Windows CE.

A few single board computers have been evaluated. A single board computer with a built in Bluetooth transceiver from HCV Wireless has been tried (See Item 2 of Figure 4.1). It has been abandoned because the vendor does not continue to offer Linux support. Two other boards have been analyzed; Korebot [43] and gumstix [27].

Korebot is a product made by K-Team, a dedicated robotics company. They are the makers of the Kheperra III, which has the Korebot as the motherboard. Korebot has an ARM processor and a lot of I/O including USB master connections, compact flash card connections, serial and USB Ethernet. The RAM is 64MB and it has 32MB flash for persistent memory. The Korebot has a special form factor similar to PC104 form factor[1]. This form factor allows the board to be connected to a few other boards in stacks. It runs a Linux operating system which is based upon the Familiar distribution[2]. This Familiar distribution has Linux packages running into the tens of thousands which one can install on the Korebot.

Gumstix makes a motherboard the size of a gum-stick packet. It has an ARM processor and various configurations of RAM and flash, ranging from 64MB RAM and 4MB flash up to 128MB RAM and 32MB flash. The I/O is very limited. The form factor makes it not as robust as the Korebot. It has it own version of Linux. No package management is offered.

After comparing the above two boards the Korebot has been chosen (see Item 3 of Figure 4.1) for the following reasons:

- The need to have a built-in Bluetooth transceiver or at least capacity to connect a Bluetooth transceiver.
- The need to have the Linux operating system supported by the vendors of the platform so the BlueZ Bluetooth software can be used.
- The I/O offered. At first it was the Personal Computer Memory Card International Association (PCMCIA) interface offered. Eventually it came to be the USB master controller which allowed a Bluetooth USB device to be used.
- The robustness of the platform because of the form factor.
- The number of utilities offered.

[1] PC104 form factor is a ruggedised smaller version of PC motherboards used in embedded devices.

[2] Familiar is a Linux distribution intended for PDAs and other small devices. It uses ipkg and the JFFS2 filesystem, a filesystem intended for flash memory.

4.1.2 Wireless Transceiver

With the wireless transceiver, infrared, WiFi and Bluetooth, the most ubiquitous options, have been considered. To decide between Bluetooth and WiFi we need to compare their Quality of Service (QoS). For our purposes the Quality of Service required is that which allows collaboration between members in RANET necessary for a soccer game. It will be defined using the following characteristics:

1. The discovery time of devices in the network and subsequent connection time.

2. The bandwidth; this is the the throughput measured in bits per second (bps). A minimum throughput can be specified [20, page 795].

3. Loss of data; in a wireless setting this would be through wireless interference.

4. Power consumption and service coverage (signal strength and how it can vary over time) [50, page 3]. In mobile ad hoc networks excessive power consumption can cause decrease in service coverage.

5. Ad hoc networking capability; this relates to robustness. In the absence of an ad hoc networking capability a single node is required to transmit data from one node to another. If this node is taken out the networking capability disappears.

A **minimum** QoS is not required. Minimum QoS is needed where you have hard real-time requirements, say a heart pacemaker sending an electrical signal every fifteen (15) milliseconds otherwise the heart will cease to beat. It is hard to see the need for imposition of a minimum QoS for the RANET. It is more in the nature of a soft real-time requirement [23]. As long as you limit the number of devices that are part of a network you can then limit the amount of traffic that is generated.

We cannot consider the physical transmitter in isolation. Each of the mentioned characteristics are affected by the intervening software layers between the transmitter and the application. Because of the choice of the peer to peer protocol (see page 43) we have to go through the TCP/IP stack when communicating. This is true whether we choose Bluetooth or WiFi. Consequently we first need to see how QoS is affected by TCP/IP in a wireless network. We will then consider each of the above characteristics.

How Wireless QoS is affected by TCP/IP

Any message needs to go through the TCP/IP stack. The TCP/IP protocol suite has proved itself on fixed line networks with client-server based services. However using the TCP/IP protocol suite in a wireless and ad hoc environment problems can arise [21].

The TCP/IP protocol is broken up into the application, transport, internet, network and hardware layers.

Application Layer At the application layer sufficient QoS would be given by adaptive techniques. This does not occur in classical TCP/IP stacks because of the independence of the layers. Independence is good in that parts of the stack can be replaced with no alteration of other layers. Knowledge of other layers destroys this independence. In wireless networks knowledge of at least the hardware layer allows one to know whether packet loss is due to congestion or merely a temporary loss in signal strength.

Transport Layer At the transport layer the TCP protocol assumes any packet loss relates to congestion although in fact it may relate more to errors in transmission of the data packet. It then uses its congestion control and avoidance algorithms. This reduces the throughput when in fact the TCP protocol need not reduce the rate at which it send packets [50, page 11]. Work has been done on modifying the TCP protocol so it can distinguish between packet loss to due to errors as opposed to congestion.

Internet Layer At the internet layer the internet routing tables would still only have the tables containing the destination internet protocol address and the next internet protocol address necessary to get to that destination. The difficulty lies in deriving the tables and maintaining them. RANETs would have routes from one device to another which would be transitory. Maintaining the routes may result in excessive traffic which is a problem in devices with limited battery life. Various solutions have been put forward. These include deriving the tables using modified flat and hierarchical structures, changing the means of discovery of network routes to an on-demand basis and limiting the degree of penetration for flood-based query mechanisms.

Network Layer At the network layer difficulties arise with the hidden and exposed terminal problems. The hidden terminal problem is where two nodes, T1 and T2, not in each others range try to transmit simultaneously to a receiver, R1. The exposed terminal problem is where T1 attempts to transmit to some R1 whilst T2 tries to transmit to R2. In some circumstances for local area networks using Carrier Sense Multiple Access T1 backs off when R1 would have received the transmission without any possibility of a collision. IEEE 802.11 does have collision avoidance which solves the hidden terminal problem but does not solve the exposed terminal problem. Further it only supports the best effort delivery. Various solutions have been put forward which unfortunately only work with single hop wireless networks. Some schemes give priority to real-time traffic but again this does not solve the exposed terminal problem.

Hardware Layer At the hardware level for wireless networks, signal-to-noise ratios vary over time. Estimation of how this would vary would require adaption in the upper layers.

What seems to be the thrust of new research is softening the boundaries between the layers of the TCP/IP protocol. Status information is exchanged between each layer. The layers then adapt to provide better QoS.

We will not consider adaptive TCP/IP stacks further. This is to again limit the problem domain. Consideration of any one of these layers and how to make it adaptive would be a thesis in itself. In any commercial grade RANET it would have to be considered.

Device discovery and connection time

Bluetooth has a device discovery and connection time of 23.55 seconds [34]. The 802.11b standard for WiFi has a device discovery and connection time of 1.15 seconds [74].

Bandwidth

Bluetooth 1.x has a data rate of 433.9Kbps (Bluetooth 2.x's higher data rate of of 2.1Mbps we will ignore, Bluetooth 1.x being more ubiquitous and lower cost). For WiFi we will concentrate on 802.11b's data rate of 11Mbps (again 802.11b being more ubiquitous and

lower cost than 802.11g). Each of these data rates will be reduced because of the TCP/IP stack overhead. We will do a paper analysis for both.

When looking at transmission over Bluetooth one would look at the size of the IP datagram that could be transmitted. This would be dependant on the underlying Maximum Transfer Unit (MTU) that could be transmitted. With Bluetooth 1.x there is asymmetric transmission of data and the symmetric data transmission used in the PAN profile. Under Bluetooth 1.x, the symmetric data transmission can be configured to a maximum rate of 433.9 Kilo bits per second (Kbps) [38, pages 25 to 27]. In these circumstances the number of bits of payload that can be transmitted is 2712 bits.

The Internet Protocol discussed would be Internet Protocol Version 4 (IPv4). This IP datagram would be contained within the Bluetooth L2CAP packet. The datagram has a header of 20 octets [69, page 36]. Assuming there was no options this would leave 2552 bits.

If a TCP Datagram was used a further 24 octets of TCP header would be taken up [20, page 221]. This leaves 2350 bits for the payload which reduces the amount of data proportionally to 92.1%. This would give a transmission rate of 399.6 Kbps.

In each piconet you have one master and up to seven (7) slaves. Assuming equal transmission time to each of the slaves the transmission rate would drop by one-seventh (1/7) which gives a transmission rate of 57.1 Kbps.

With JXTA, messages are transmitted in XML which is text. Assuming 8 bits for each ASCII symbol we have room for 7137 letters.

Assuming the average series of words in XML to be 5 characters long, 1427 words per second can be sent. This appears to be enough for messaging. However, it would probably not be enough for file transfer if the files were of any large size. This would have to be done in binary format.

Using Bluetooth with no TCP/IP stack would make little difference as the transmission time in the same eight (8) device piconet drops to one-seventh (1/7) of 433.9 Kbps; giving a transmission rate of 61.9 Kbps.

KR080309

Where wireless interference takes place, Bluetooth has an Automatic retransmission request (ARQ) system which allows for retransmission of lost or corrupted packets. The TCP component in the TCP/IP stack would also retransmit that data. Unfortunately the congestion control and avoidance algorithms in TCP would also kick in, decreasing the size of the window in which to send data packets. This would cause a fall in throughput when what was needed was a mere retransmission of a packet of the same size.

For WiFi the number of bytes in the underlying MTU is generally considered to be 1500, giving 12000 bits [64]. Make the same assumptions for the IP datagram and TCP header the number of bits is reduced to 11648. The throughput falls proportionally to 98.7%. This would give a transmission rate of 10.86Mbps.

Data loss

With wireless devices that work in environments with much unintentional and intentional electro-magnetic emission, interference with transmission and reception of data can occur. We now discuss whether Bluetooth or WiFi is better at handling interference.

Bluetooth transmits in the 2.4GHz band and has interference from intentional emitters such as 802.11b (WiFi) and Home RF and unintentional emitters such as microwave ovens and overhead cables [38, page 29]. Specifically, the effect at ten (10) metres or greater is negligible with the throughput staying at 90 percent [38, page 32]. At distances of less than ten (10) metres the throughput drops, from about 50 percent when the receiving

Characteristic	WiFi	Bluetooth
Discovery and Connection time	1.15 seconds	23.55 seconds
Bandwidth	10.86Mbps	399.6Kbps (57.1Kbps in 8 member piconet)
Data loss (interference)	0m - 50% throughput, 7.5m - 5% throughput	<10m - 50% throughput, 60m - 0%
Data loss (no interference)	nodes interfere with each other	nodes have switched network
Power consumption	500mW	60mW
Ad hoc networking capability	Limited	robust up to 8 member piconet

Table 4.1: Wifi and Bluetooth comparison

device is at 0 metres to about 5 percent at 7.5 metres [38, page 32].

WiFi is similar with great interference taking place at less than ten (10) metres with throughput at 50% dropping off to zero at approximately 60 metres [38, page 33].

In the absence of any wireless interference and with a limited number of devices Bluetooth comes into its own. This is because at any one time one the master and one slave are communicating; only one node in group is transmitting at any one time. This does not occur with WiFi where two nodes in a group could transmit at the same time; their signals colliding with each other.

Power consumption

Power consumption in is Bluetooth is 60 mW versus 500 mW for WiFi [63].

Ad hoc networking capability

Bluetooth has efficient ad hoc networking compared to WiFi's cumbersome ad hoc networking [54, pages 484 to 485]. Bluetooth has the capacity for up to 7 slaves and depending on the Bluetooth chipset some scatternet capacity.

An evaluation

Bluetooth is preferred (see Item 4 of Figure 4.1). This is because of its lower power requirements and better ad hoc networking capability.

4.1.3 Operating system

As previously stated, the choice of wireless transceiver, Bluetooth meant Linux was chosen as the operating system. This is because of the need to use BlueZ software to control the Bluetooth transceiver. Other factors also made it the standout choice. The open source software used on Linux is a major factor. We have been required during the making of the reference implementations to patch the kernel with a BlueZ change and with OSGi inspect the source code to find a bug. This would have all but been impossible with closed source software. The ease of development within a Unix environment is also a major factor.

As the peer to peer protocol requires TCP/IP in addition to the normal BlueZ functionality (see page 43), the BlueZ profile, PAN is also required. PAN allows for TCP/IP

packets to travel over Bluetooth frames instead of Ethernet frames. A patch has been applied to the 2.4.19 Linux kernel (see Item 5 of Figure 4.1 and Section C.1.1) to allow it to provide the full PAN functionality. Further, all the Bluetooth libraries and utilities needed to support PAN need to be freshly compiled (see page 77). This allows an IP link over Bluetooth to occur.

When using the Linux operating system the startup scripts are written as per the BlueZ document "How-To set up common PAN scenarios with BlueZ's integrated PAN support" [14]. This automates the IP connection over Bluetooth.

The startup script is "dev-up". It requires an Ethernet Bridge (see Figure 4.5). Obtaining an Ethernet bridge means a 802.1D kernel module and a utility called "brctl" has to be cross compiled. Such Ethernet bridges connect different Ethernet networks together transparently. For members of a virtual network, the Ethernets will appear to be one Ethernet [17]. The Ethernet Bridge stops a device having multiple IP addresses when it is a master of the piconet. Incidentally, the Ethernet bridge has a Spanning Tree Protocol (STP) which determines the shortest route from one Ethernet to another and also eliminates any cycles from the topology. This potentially could maintain the topology of a scatternet RANET. For the the purposes of our piconet RANET we have turned off STP.

We first tried the onboard Bluetooth transceiver for the HCV single board computer. As we require the BlueZ stack we had to abandon this transceiver when the vendor no longer offered Linux support (see Item 2 of Figure 4.1). The vendor of the Korebot recommended a acPCMCIA Bluetooth card (see Item 6 of Figure 4.1). This has not worked because of a lack of Linux driver maturity in this area (see Diary Excerpt in Section D.1 and email trail excerpt in Section D.2 where the author has to patch the kernel to no avail). Lastly a USB Bluetooth had been successfully tried (see Item 7 of Figure 4.1 and Diary Excerpt in Section D.3).

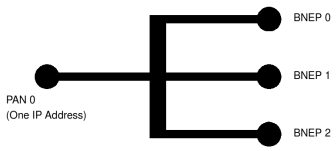


Figure 4.5: Ethernet Bridge

4.1.4 Peer to Peer protocol

The choice of which peer to peer protocol to use has been narrowed to GNUtella, JXTA and P2PS (Note that P2PS is so close to JXTA in what it does that where we refer to JXTA it is taken to refer to P2PS). We now assess what protocol would better suit a RANET. To do that we need to explore the concept of groups; what is their composition, how they are formed, the ways in which information is distributed amongst its members and how they cooperate.

Group Composition

In most peer to peer protocols there is no concept of groups, the composition of a group instead being tied very closely to the network's topology. The group itself is as large as the number of machines using the protocol that are reachable on the network. This means that if all are reachable, all are part of the group. There is no real understanding of being part of a group [47, pages 25 to 38].

Groups are very important in peer to peer networks. Their greatest feature is not allowing communication but limiting it. If messages (be they data, text, code, etc.) are restricted to the group, it reduces flooding of the network. In wireless communication anyone can send and receive just by a broadcast. Groups are required so the participants in any conversation are limited.

It seems then that a robot would need to know that it is a member of a RANET. This knowledge would mean communication could be limited to RANET members. Further, not belonging to a RANET would allow a robot to not participate in any present conversation taking place. This would allow for the existence of separate teams in a robotic soccer game; each team not being privy to the plans of the other.

JXTA has the concept of a group which is known to the device. In JXTA, a device or peer is defined to be a member of group if it holds that group's unique ID [72, page 7]. In this way it can be a member of a RANET.

Group Formation

Groups must be first formed by its members. This involves discovery of the group, joining it and leaving it. The possibility exists too of handover from one group to another.

Group discovery is done in a variety of ways amongst various peer to peer protocols. With GNUtella a device participating in the GNUtella network acts as both a client and a server and so is termed a "servent". Once a device (G1) is connected to a servent (G2), it Pings G2. G2 in turn Pings the servents it knows, G3 and G4. G4 replies with a Pong, following the route taken to reach it back to G1. This is turns happens with G2 and G3. In this way G1 discovers the GNUtella group. There is no limit or boundary to the devices running the GNUtella protocol it can discover [24, page 3]. JXTA, allows a device to send out a message, called a discovery request. Other devices in the local area respond with messages, more specifically called advertisements, describing the groups they belong to [72, page 39] [3].

Joining a group with specific details (say a name "Blue Soccer Team") does not occur in GNUtella. The group joining is circumstantial as discussed above (see also [48, pages 25 to 38]). Once other devices using the protocol are discovered, no joining is required for file transfer or messaging to take place. In JXTA, a device or peer is defined to be a member of group if it holds that group's unique identification (ID) [72, page 7]. However, after a JXTA group member has found other groups members, it does not mean it joins the group automatically. With JXTA, one must first join a group before skills can be obtained from group members [72, page 57]. Some group security may be beneficial for a RANET; say preventing a rogue robot from joining the RANET to cause harm. Group security varies between all the peer to peer protocols; from nothing like GNUtella to supplying of usernames and passwords or generated keys in the case of JXTA [47, pages 22 to 38] [72].

[3] This method of discovery has been used to develop a prototype application for mobile terminals [39]. Unfortunately the developed application could not be run on real world mobile terminals, only on mobile terminal emulators.

Leaving groups is not covered by GNUtella. JXTA has the concept of resigning from a group.

Handover from one group to another occurs in mobile telephone networks when a mobile phone is physically leaving one cell and joining another. Peer to peer protocols predominately exist at the application layer [47, pages 25 to 38] and seem to emphasize the individual. There is no group or a leader that makes a decision to hand an individual device over to another group. JXTA, the only reviewed protocol with group functionality, places the onus on the individual to leave a group and join another [72].

Distribution of Information to the Group

In pure peer to peer networks all nodes in the graph are similar if not the same. Pure P2P networks distribute group information to all other nodes equally. In practice this rarely happens because of the flooding of the network that would occur [7]. Instead, some nodes form a backbone for the group, such as the JXTA Rendezvous nodes, so they can distribute necessary information [72, pages 16 to 17].

A robot can distribute information about itself or other RANET members. This information is sent by a robot pushing the information to the network or a robot requesting information and another robot responding. More simply, information can be pushed to the network or pulled from it. Whether information should be pushed or pulled would depend on how urgent it was and the traffic congestion it may cause.

GNUtella in the main has a pull mechanism[4]. Flooding of the network with the GNUtella Pings and Queries is prevented by the protocol having time to live (TTL) packets which are decremented at each hop. At a predetermined number of hops from the first querying/pinging device the packets die. The protocol is silent as to whether the information is cached or disposed of.

JXTA both pushes and pulls its information. To explain, JXTA has four types of peers, minimal edge peer, full-featured edge peer, rendezvous peer and relay peer [72, pages 15 to 16]. The minimal edge peer only sends and receives messages but does not cache any advertisements or route any messages. The full featured edge peer does all that a minimal edge peer can do but also caches advertisements. A Rendezvous peer does all that a full featured edge peer can do but also routes discovery requests and maintains information on other peers. A Relay peer routes messages to peers through firewalls. A peer can have both the functionality of a Rendezvous and a Relay peer.

The pushing of information is done by the Shared Resource Distributed Index. Whenever an advertisement is published by an edge peer, details are provided to its list of Rendezvous peers. This information is periodically propagated through the various Rendezvous peers contained within the group. This pushing of information increases the amount of traffic but would probably limit the extent and time of any query.

Queries in JXTA are done using a local broadcast and through the Rendezvous peers.

GNUtella has been shown not to scale well because of it group distribution information mechanism [7]. JXTA would limit the amount of traffic. Whether the slow push of information about advertisements would work in a RANET is another matter.

[4]The exception to this is where a direction connection cannot be made by the first querying device, say where a firewall prevents a direct HTTP connection being made. When this occurs the first querying device sends a Push to the servent that has the file. That servent then attempts to form a connection [18].

Group Cooperation

Group cooperation involves messaging to achieve some joint intention. It also involves transfer of skills to assist each other in this joint intention. The main thrust of this thesis is to provide a design for a network which can allow group cooperation. Two questions need to be asked. Firstly, does the type of messaging that GNUtella or JXTA offers support the concept of group cooperation? Secondly, is there any mechanism for skill discovery and transfer?

GNUtella has no messaging mechanism. The only way to send messages would be to write messages to message files and then query on those message files. This is at best a workaround. JXTA has messaging, using pipes to allow messages to flow between peers.

Skill discovery is dealt with in a variety of ways. The GNUtella protocol, for example, sends a Query to another device running the GNUtella protocol for a file. If a hit is not found the servent sends a Query out to all its known servents. If a match is found on a device it replies with a QueryHit, following the original route taken to reach it by first querying device. Using HTTP the file is then downloaded [24, page 3]. Skill discovery requires the querying device to know up front the name of the skill. GNUtella could be configured so a file could contain a list of all other files. This would then allow a device to advertise the skills (files) that it has.

The JXTA protocol displays skills using advertisements [72, pages 84 to 101] which are obtained through a skill discovery request.

Many P2P protocols deal with file transfer [47, page 23] using File Transfer Protocol (FTP) or HTTP [24, page 3]. The JXTA protocol has the concept of pipes that a computer uses to send and receive messages [72, pages 84 to 101]. These pipes are the conduit along which skill transfer takes place.

Skill transfer could occur using both GNUtella and JXTA. JXTA provides a better method of skills discovery.

An evaluation

The JXTA peer to peer protocol has all the necessary functionality; group discovery, group formation, skill discovery, file transfer and messaging.

We have decided each robot would be a full-featured edge peer; able to send and receive messages and also cache advertisements. It discovers other robots in the RANET by doing a local area broadcast. There will be no backbone of Rendezvous peers as per Item 1 Figure 4.6 upon which information would be distributed. Being full-featured edge peers, the robots form a homogeneous environment as per Item 2 Figure 4.6.

With JXTA there is:

- Jadabs, a version written in Java which has a JXTA implementation, also combines OSGi and uses Bluetooth instead of TCP/IP [32].
- P2PS, a JXTA clone written in Java.
- JXTA-C, a version written in the C programming language.
- JXSE, a version written in Java intended for desktop computers and servers.
- JXME, a version written in Java intended for embedded devices.

P2P2, JXTA-C, JXSE and JXME all run on TCP/IP. Jadabs has been tried (see Item 8 of Figure 4.1). Unfortunately, the reference implementation is incomplete so the author cannot run a precompiled version or compile Jadabs (See the email trail in Section

F.1 and the Diary excerpt in Section F.2 regarding the author's efforts). It is not worth the effort to port the JXTA-C libraries to the Korebot as there is Java byte code, the JXME libraries, that should run without modification on the Korebot. JXME is our JXTA choice (see Item 9 of Figure 4.1).

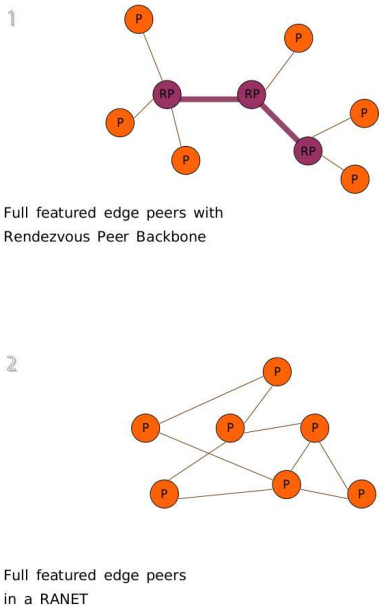


Figure 4.6: Typical JXTA topology versus RANET JXTA topology

4.1.5 Java Virtual Machine

To support JXME we require a Java Virtual Machine (JVM) to run on the Korebot. Attempts made at porting the Sun Java Virtual Machine CDC to the Korebot have met with little success (see Section E.2 and Item 10 of Figure 4.2). Attempts at porting a smaller Sun Virtual Machine (the CLDC) came up against export license restrictions laws of the United States (see Item 11 of 4.2). These laws prevent release of all the source necessary to port the JVM. We settled on JamVM (version 1.4.3), a Java Virtual Machine which is easily ported to the ARM processor (see Item 12 of Figure 4.2 and Section E.3.1 as to building it.). Although it only can interpret the bytecode, it is a fully functional JVM and very small in size [41]. It uses the GNU Classpath libraries (version 0.91), an open source implementation of the Java libraries. It is similar to the Sun CDC running Java version 1.4.2.

4.1.6 Skill Packager and Dynamic Skill Loading

Dynamic loading of skills is required for the robots in the RANET to learn and remember without restarting the Java Virtual Machine. Two options have been considered when looking at skill packaging and dynamic skill loading; Jython and OSGi.

Jython, not Python, is required because of the JXTA implementation being written in Java (see Item 13 of Figure 4.2). Jython runs on the Java virtual machine and can use existing Java libraries without modification. Python/Jython can pickle amongst other things, classes and object instances [55]. This way not only can a skill's program code be packaged but also its program state. Unfortunately Jython does not run on JamVM 1.4.3 or the earlier 1.4.2 version, running only on Sun Java virtual machines (see Diary Excerpt in Appendix G).

OSGi is a specification written by OSGi alliance (formerly the Open Services Gateway Initiative) which allows for "an open service platform for the delivery and management of multiple applications and services to all types of networked devices in home, vehicle, mobile and other environments" [53, Question 1]. Several implementations have been done in Java (see Knopflerfish as an example [40]). It allows services to be loaded, run, paused, resumed and deleted. We considered at first Knopflerfish but have chosen Apache Felix (see Items 14 and 15 of Figure 4.2). It has the advantage of being written in Java 1.4.2, allowing it to be run on many embedded devices (Java Two Micro Edition (J2ME) only runs on Java 1.4.2. Later versions of Java introduced Generics and enumerations). It further had a text interface by default as opposed to the GUI interface used by Knopflerfish. This allows it to be used without modification on the Korebot, which only has a terminal interface.

With loading skills we need to load skills at runtime and also have them remembered in some way. OSGi loads the skill. One can then query the OSGi framework to find the service containing the skill. Lastly the OSGi framework remembers what skills have been loaded and also whether they are active or dormant.

4.1.7 Non-volatile Storage

We have add some 256MB of acPCMCIA flash to the Korebot in addition to the 32MB of flash already present. This is so the robot can remember skills received by writing to persistent memory (see Item 16 of Figure 4.2).

4.2 Extra Structures

We now have some structures (the third party COTS) which appears to meet the design but some analysis is necessary to ensure there were no gaps. Gaps exist in the Bluetooth PAN and not having OSGi bundles which wrap up the JXTA functionality and the business logic for the RANET.

4.2.1 PAN using Ethernet Bridge

The Bluetooth PAN using the Ethernet bridge works if some robotic devices are always slaves and one robotic device was always a master. However master devices can not always be expected to be available. The RANET needs to work so that any RANET member can come into range of another RANET member and always form an ad hoc network. This needs to occur without the constraint of always being a master or a slave.

The PAN profile is adapted as follows (see also Figure 4.7):

1. The robot starts out a slave.
2. It scans for Bluetooth addresses.
3. The slave then receives Bluetooth addresses from remote devices.

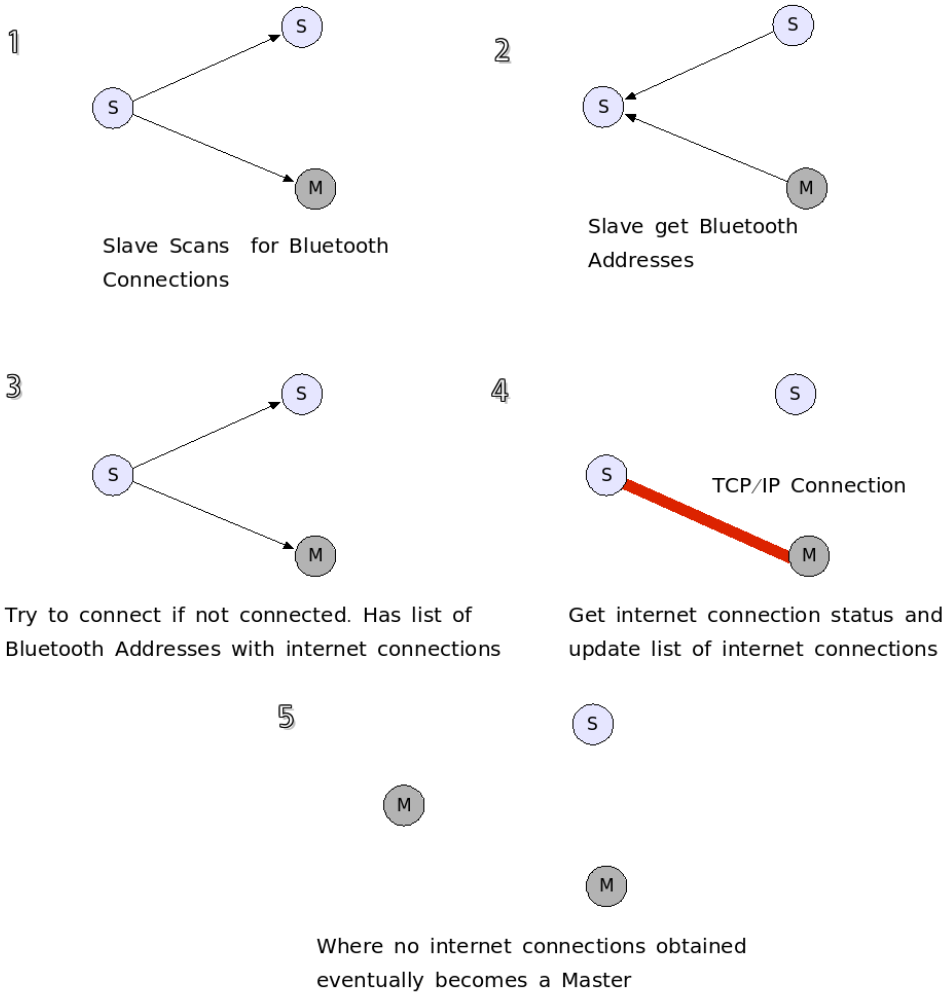


Figure 4.7: Adapted PAN protocol

4. It then tries to connect to each of the devices it has not already connected to.
5. It then checks its internet connection status and updates its internet connections.
6. If eventually it makes no connections as a slave it becomes a master (This is done on the assumption that there are no devices in the immediate area or that they are all slaves).
7. As a master it listens for connections.
8. It checks its internet connection status and updates its internet connections.
9. If eventually no connections are formed it becomes a slave.
10. The process then repeats.

Previously the slave device did not use an Ethernet bridge for IP connections. The periodic changing of a device from slave to master has altered the design so like a master device, a slave now adds TCP/IP connections to the Ethernet bridge.

Adapting the PAN profile in this way we believe ensures that in a piconet, an ad hoc network can always be formed. This has not been done before. It is preferred to other Bluetooth protocols in that it can dynamically form ad hoc networks. Most extensions to Bluetooth protocols require either Global Positioning System (GPS) transceivers so the robots at intervals can transmit to each other the necessary information on each device's location and then form a scatternet based upon the reported positions or using the strength of the Bluetooth signal to decide upon who will be the master.

Adapting the PAN profile in this way one can form scatternets. This is because slaves can have more than one TCP/IP connection and slaves continue to try and make connections. If the Bluetooth transceivers supported it, a scatternet can form with slaves as the connection points between piconets.

We have turned this extension to the Bluetooth PAN profile into an OSGi bundle (see Item 18 in Figure 4.3).

4.2.2 OSGi bundles - JXTA and the Application

JXTA is a library so we need an OSGi bundle to wrap up the JXTA libraries and call out the JXTA library functions necessary for a RANET. Some code also needs to be written for the RANET which used the Bluetooth and JXTA bundles.

4.3 The design

Possible choices for each structure required and the structure chosen are summarized in Table 4.2. The structures chosen are shown in Item 20 of Figure 4.3. They are:

- Korebot for the Single board computer.
- Bluetooth USB stick for the wireless transceiver.
- Linux for the operating system.
- JXTA(JXME) for the Peer to Peer protocol.
- OSGi for the skill packager and dynamic skill loader.
- acPCMCIA flash storage for non-volatile storage.
- JamVM, for the Java virtual machine.
- Bluetooth OSGi bundle for the extension to the Bluetooth PAN profile.
- JXTA OSGi bundle which uses some functions from the JXTA library.
- RANET OSGi bundle for the business logic.

STRUCTURES REQUIRED	POSSIBLE CHOICES	CHOICE MADE	REASONS
Single board computer	HCV wireless, gumstix, Korebot	Korebot	vendor Linux support, Bluetooth Tx/Rx, I/O, robustness, ipkg
Wireless transceiver	Infrared, Bluetooth, WiFi	Bluetooth USB dongle	Lower power reqs, better ad hoc networking capability
Operating system	Microsoft CE, Linux, None	Linux	BlueZ (Previous choice), open source, development environment
P2P protocol	GNUtella, JXTA, P2PS	JXTA(JXME)	group discovery, group formation, skill discovery, file transfer and messaging
Skill packager & Dynamic skill loader	pickled objects, OSGi	OSGi	better skill packaging, Jython not working
Non-volatile storage	USB flash, acPCMCIA flash	acPCMCIA flash	Ease of use, robustness
Java Virtual Machine	Sun VMs, JamVM	JamVM	Sun JVMs not working and export restrictions

Table 4.2: Structures Chosen

Chapter 5

TESTING THE DESIGN

Chapter Summary - To test the final design we implemented a reference implementation and ran some tests of that implementation. In doing so we find some deficiencies in the third party software, specifically JXTA and JamVM/GNU Classpath

5.1 The Reference Implementation

Some testing of the components has been done when arriving at the initial design. More testing of the components has occurred when to see whether the overall design works. This is where where we have found deficiencies in the open source software we used, JXTA and JamVM/GNU Classpath.

5.1.1 Problems using JXTA and JamVM/GNU Classpath together

Reference implementations have been tried from the JXTA community website [37]. The first one tried was JXME version 2.1.3 (see Item 21 of Figure 4.3). This is a JXTA implementation written in Java J2ME which is version 1.4.2 of the Java programming language. It has been found that it cannot operate in an environment where there are no Rendezvous peers. Further it cannot become a Rendezvous peer itself. This means one could not have a homogeneous RANET using JXME. Some RANET members would need to run JXSE. This is at odds with any two robots being able to form a link (see Diary Excerpt in Section I.1 for the full details).

Because the version of Java running on the Korebot is 1.4.2 the last version of JXSE (version 2.3.6) that was meant to use version 1.4.2 has been tried (see Item 22 of Figure 4.3). It has been found that although it is meant to use version 1.4.2 of Java in fact it uses Java 1.5 Generics and Enumerations (see Diary Excerpt in Section I.2).

Accordingly a version of JamVM 1.4.3 that uses GNU Classpath 0.91 Generics has been tried. This lacks the JSR166 Concurrency libraries used by JXSE 2.3.6 (see Diary Excerpt in Section E.3.2 on how to build it and Item 23 of Figure 4.3). JamVM 1.4.5 has been tried along with GNU Classpath 0.95 which includes the JSR166 Concurrency libraries. Unfortunately some code changes prevent JamVM from executing Java code (see Diary Excerpt in Section E.3.2 and Item 24 in Figure 4.3). In summary, the Korebot (the target robotic device) cannot use the JXSE functionality because it cannot not run the Sun Virtual Machine. The JXSE implementation requires Java 5 and the JSR166 Concurrency libraries.

P2PS (a JXTA clone) has been tried. The GNU Classpath libraries do not support some UDP functionality which is required for the broadcast queries (see the Diary excerpt in Section I.3 and Item 25 of 4.4).

Figure 5.1: Test Bench

We have tried a later version of JamVM/GNU Classpath on a Linux laptop and find it does not have the necessary encryption algorithms (see the Diary excerpts in Section I.5 detailing the error and Item 26 of 4.4).

5.1.2 A Workaround

In the end the JXTA component has to be tested on computers which run Sun Virtual Machines version 1.5 (see Item 27 of Figure 4.4), in our case Laptops with x86 processors, not ARM ones. JXTA and the Sun Java virtual machines work together.

JXSE 2.5 has been used as it is the easiest to configure. A 1.5 Sun Java Virtual Machine has been used as it likely it will be ported to embedded systems in the near future (Java 1.4.2 used to only run on Desktops with Java 1.3.0 running on embedded systems. It is likely processor power will increase to such a degree that Java 1.5 will soon run on embedded systems.).

The two final designs are referred to in Item 29 and Item 30 of Figure 4.4 for the Korebot and a Laptop respectively. Implementations of these two designs have been used in the tests referred to in Section 5.2.

5.2 Testing the Reference Implementations

The test setup is two Laptops running Ubuntu Linux and the Korebot (see Figure 5.1 which shows the base test bench). Attached to each is a Bluetooth USB dongle. The intention is to connect on a Class A 10.x.x.x network. Monitoring is done on a Class C 192.168.x.x network. This ensures that no packets are transmitted erroneously through the wired ethernet. Video capture is done on the monitoring computer. A further test has been done using a larger computer as one of the participants in the RANET. This allows video capture to occur locally on the RANET member. eliminating the need for the Class C network. Note that the Bluetooth extension has worked on the Korebot. The Korebot

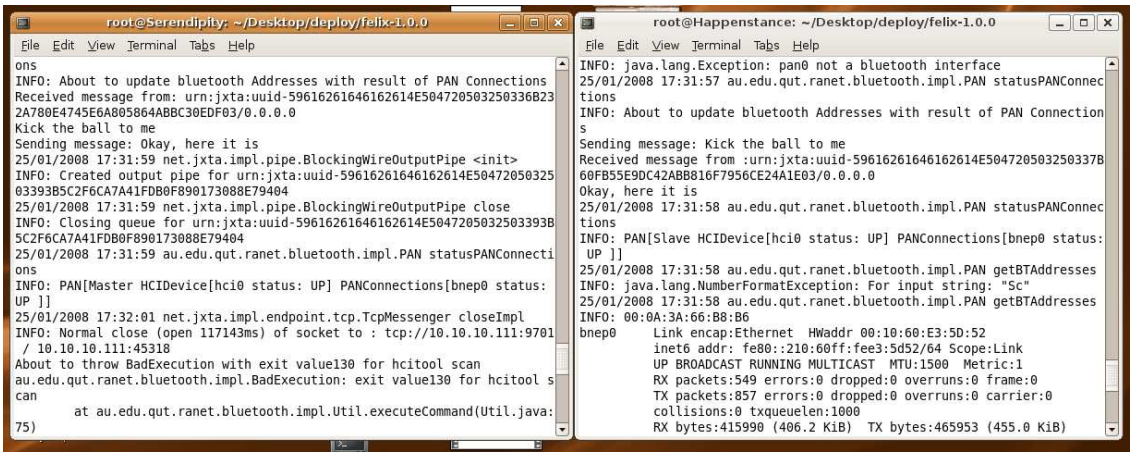


Figure 5.2: Two Laptops Messaging

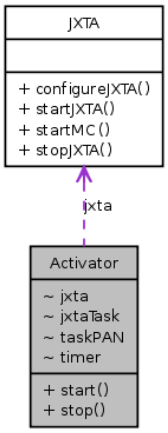


Figure 5.3: RANET Multicast Class Diagram

and a Linux laptop have been able to form an ad hoc network, bringing up a TCP/IP connection. Note that the programming examples from *JXTA JavaTM Standard Edition v2.5: Programmers Guide* [36] have been a useful guide.

5.2.1 Messaging between two laptops

Messaging has been done first using the two laptops which run Ubuntu Linux. The client laptop connects using a known JXTA advertisement to the server laptop; the advertisement relating to multicast messaging. A message to kick the ball is sent. It is received and a response sent saying "Here it is" (see Figure 5.2). This messaging has been presented in a paper at a conference on mini-robotics [57].

What is going on? The OSGi bundles involved are RANET Multicast, Bluetooth Service (a wrapped up BluetoothImpl jar file) and JXTA Multicast Service (a wrapped up JXTA Multicast Impl). Each of these are discussed in turn.

```
00163     private static class PANConnectionTask extends TimerTask
00164     {
00165         Bluetooth bluetooth;
00166         public PANConnectionTask(Bluetooth _bluetooth)
00167         {
00168             super();
00169             bluetooth = _bluetooth;
00170         }
00171
00172         public void run()
00173         {
00174             //bluetooth.
00175             bluetooth.statusPANConnections();
00176             Thread.yield();
00177         }
00178     }
00179
00180     private static class JXTAConnectionTask extends TimerTask
00181     {
00182         JXTA jxta = null;
00183         Bluetooth bluetooth = null;
00184         boolean result = false;
00185
00186         public JXTAConnectionTask(JXTA _jxta, Bluetooth _bluetooth)
00187         {
00188             super();
00189             jxta = _jxta;
00190             bluetooth = _bluetooth;
00191         }
00192
00193         public void run()
00194         {
00195             if (bluetooth.havePANConnections()) {
00196                 jxta.startMC();
00197             }
00198             Thread.yield();
00199         }
```

Figure 5.4: RANET Multicast Code Excerpt

RANET Multicast

The RANET Multicast application (see Figure 5.3) is the bundle that uses the Bluetooth Service and the JXTA Multicast Service. The Bluetooth Service and the JXTA Multicast Service are called out in PANConnectionTask and JXTAConnectionTask respectively (see the code excerpt in Figure 5.4). When the RANET Multicast application begins the Bluetooth link is attempted in PANConnectionTask (see Figure 5.5). After the link is established a query for a multicast service is broadcast by the client in JXTAConnectionTask. If the query finds the multicast service the same query delivers the message "Kick the ball to me". The multicast service provides a directed response "Okay, here it is".

Listed in Figure 5.6 is the client laptop's standard output. The output is similar for the server laptop.

Bluetooth Service

The Bluetooth Service (see Figure 5.7) is an OSGi bundle which wraps up the Bluetooth Implementation jar file, exposing only the functionality it desires. The services that this bundle provide can be registered with the OSGi framework.

Bluetooth Implementation

BluetoothImpl on initialization causes the PAN and the ethernet bridge (Bridge) to be created. Its two main operations after initialization is seeking and maintaining PAN (TCP/IP) connections and to advise other classes whether there are any PAN connections.

The main classes (see Figures 5.8 and 5.8) are BluetoothImpl, PAN and Bridge. When using Bluetooth Implementations the following command line utilities are required:

- For BluetoothImpl - hciconfig and hcitool.

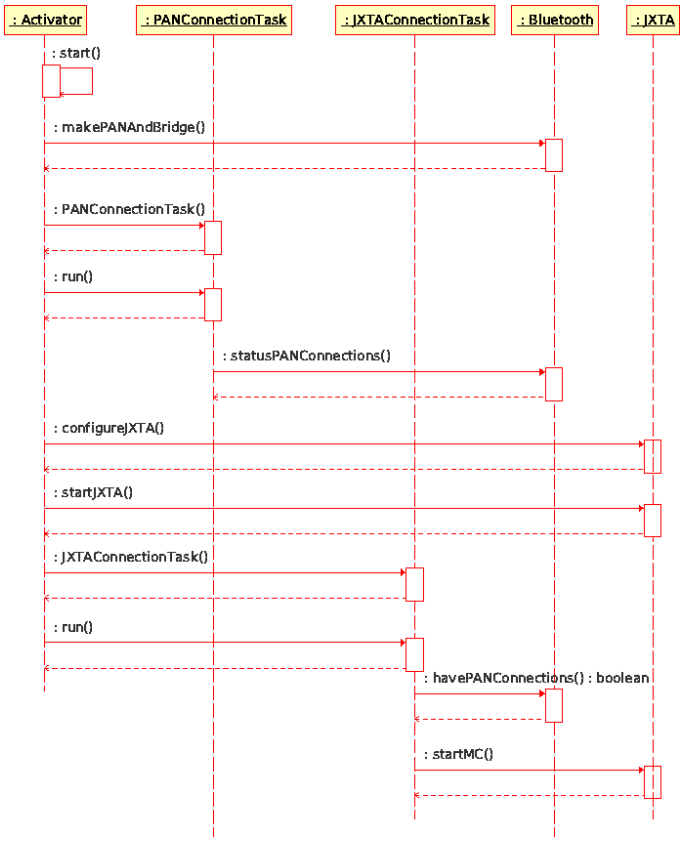


Figure 5.5: RANET Multicast Sequence Diagram

```
CLIENTSIDE:
44BFEAD20A38CB8A9C44B03/0.0.0.0
Okay, here it is
24/09/2007 10:37:29 au.edu.qut.ranet.bluetooth.impl.PAN statusPANConnections
INFO: PAN[Master HCIDevice[hci0 status: UP] PANConnections[bnep0 status: UP ]]
24/09/2007 10:37:40 au.edu.qut.ranet.bluetooth.impl.PAN getBTAddresses
INFO: java.lang.NumberFormatException: For input string: "Sc"
bnep0     Link encap:Ethernet  HWaddr 00:0A:3A:66:B8:B6
          inet6 addr: fe80::20a:3aff:fe66:b8b6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:34 errors:0 dropped:0 overruns:0 frame:0
          TX packets:63 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:18864 (18.4 KiB)  TX bytes:19413 (18.9 KiB)
24/09/2007 10:37:40 au.edu.qut.ranet.bluetooth.impl.PAN statusPANConnections
INFO: PAN Connection bnep0 status: UP

...

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:107 errors:0 dropped:0 overruns:0 frame:0
          TX packets:107 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:8881 (8.6 KiB)  TX bytes:8881 (8.6 KiB)
24/09/2007 10:37:40 au.edu.qut.ranet.bluetooth.impl.PAN statusPANConnections
INFO: java.lang.Exception: lo not a bluetooth interface
pan0      Link encap:Ethernet  HWaddr 00:0A:3A:66:B8:B6
          inet addr:10.10.10.222  Bcast:10.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::200:ff:fe00:0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:33 errors:0 dropped:0 overruns:0 frame:0
          TX packets:125 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:18594 (18.1 KiB)  TX bytes:24115 (23.5 KiB)
24/09/2007 10:37:40 au.edu.qut.ranet.bluetooth.impl.PAN statusPANConnections
INFO: java.lang.Exception: pan0 not a bluetooth interface
24/09/2007 10:37:40 au.edu.qut.ranet.bluetooth.impl.PAN statusPANConnections
INFO: About to update bluetooth Addresses with result of PAN Connections
Sending message: Kick the ball to me
Received message from :urn:jxta:uuid-59616261646162614E50472050325033AD2463515\
0944BFEAD20A38CB8A9C44B03/0.0.0.0
Okay, here it is

...
```

Figure 5.6: Client RANET Multicast output

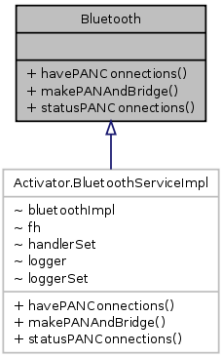


Figure 5.7: Bluetooth Service Class Diagram

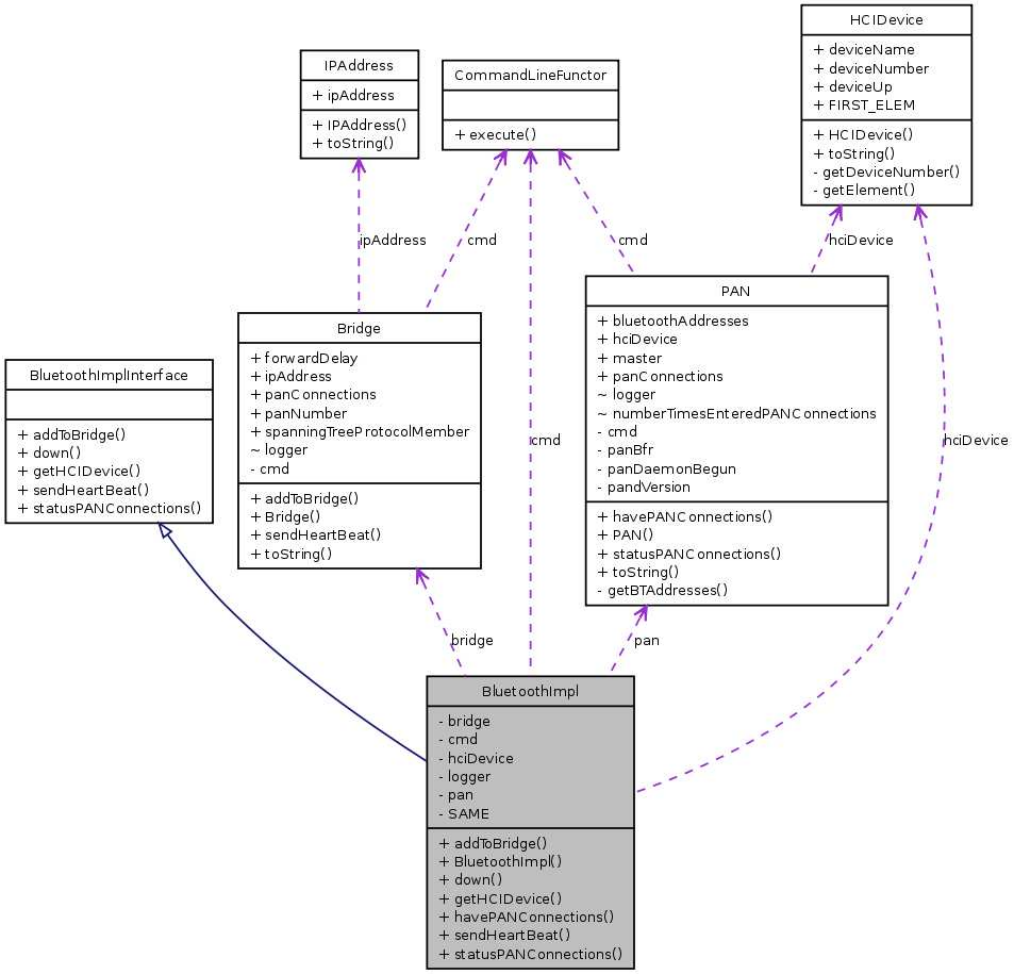


Figure 5.8: Bluetooth Implementation Class diagram

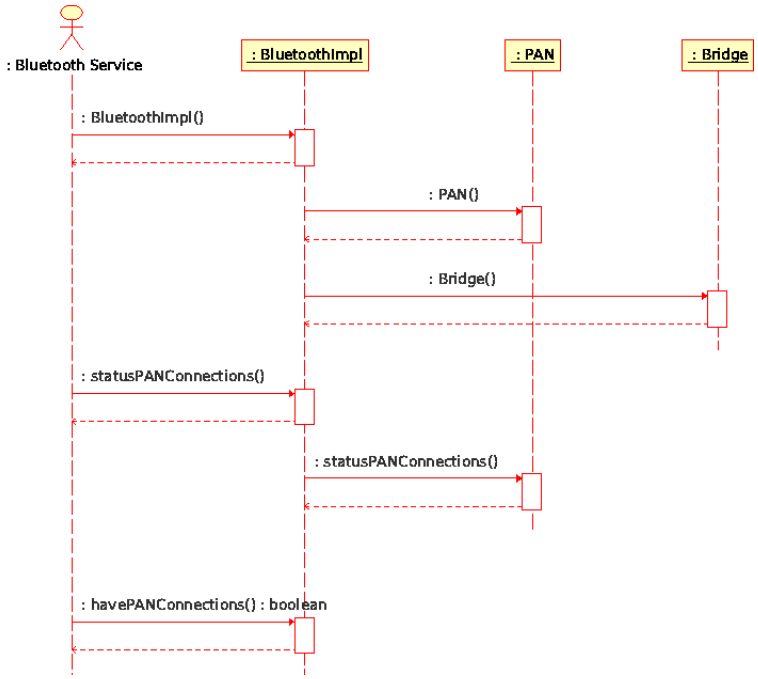


Figure 5.9: Bluetooth Implementation Sequence Diagram

- For Bridge - brctl, ifconfig and route.
- For PAN - pand, hcitool, ifconfig and start-stop-daemon.

When running BluetoothImpl the ulimit command needs to set the file number descriptors to 2048. This is done by typing "ulimit -n 2048". Because of the slightly different behavior between earlier and later versions of the pand utility the PAN class uses a BLUETOOTH.properties like the following:

```
pandVersion=2.25
```

This is the pand version number.

JXTA Service

The JXTA Service uses the JXTA implementation jar file (see Figure 5.11a). It makes an OSGi bundle. The services that this bundle provide can be registered with the OSGi framework.

JXTA Implementation

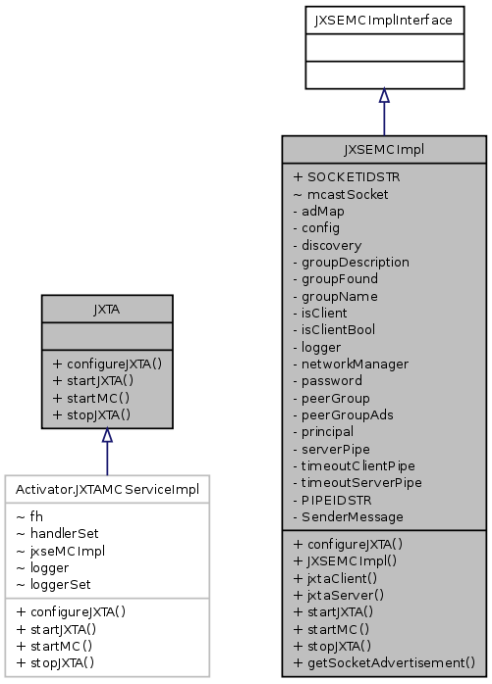
This JXTA Multicast implementation operates as either client or server depending on the contexts of the JXTA.properties file (see Figure 5.10).

A client has in one frame a query request and a message. The query request looks for a JXTA Multicast Server advertisement. Once such an advertisement is found a connection is made and the message is delivered.

The JXTA implementation (see Figure 5.11b) uses the JXSE Release Candidate 2.5. This implementation has many helper classes. These helper classes can easily create pure ad hoc networks; networks which do not require seeding of some super peer. Broadcasts are done using IP multicast. The IP multicast is enabled in the Bridge class (see Figure

```
# JXTA properties for peer group, etc
peerName=BlueOne
groupName=BlueSoccerTeam
groupDescription=The Blue Soccer Team
principal=BlueOneUser
password=password
isClient=Yes
timeoutServerPipe=0
timeoutClientPipe=60000
```

Figure 5.10: JXTA properties file



(a) JXTA Service (b) JXTA Implementation

Figure 5.11: JXTA Multicast Class Diagrams

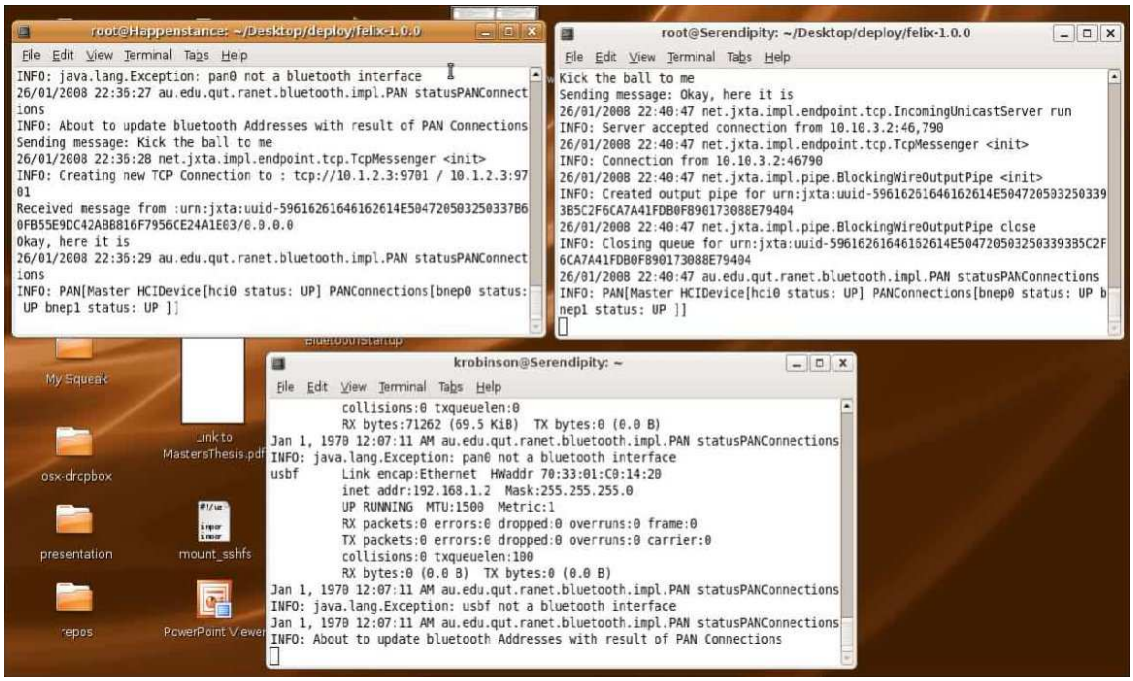


Figure 5.12: Messaging through the Korebot

5.8) with the route command "route add -net 224.0.0.0 netmask 240.0.0.0 dev pan0". This allows broadcasting to occur using the 224.0.1.85 IP address.

5.2.2 Messaging between two laptops through the Korebot

Figure 5.12 shows messaging between two laptops through the Korebot. This demonstrates two things. The first is that the Korebot is able to establish and maintain an ad hoc network. The second is the Ethernet bridge on the Korebot allows connections to be made transparently; The fist laptop is making a connection through the Korebot to the second laptop. RANET Main uses Bluetooth Service and JXTA Multicast Service on the two laptops and only Bluetooth Service on the Korebot.

An analysis of memory on the Korebot when running the Bluetooth extension shows 4108KB of memory used after startup of some components. Memory usage jumps to 5368KB when starting two small bundles in this framework. The persistent memory used for the framework amounts to 132.5KB. The total persistent memory used amounts to 17.8MB. Most of this relates to the Java libraries.

Execution times were recorded using a Java library supplied logger and were found to be in the order of hundreds of milliseconds.

5.2.3 Messaging between two laptops on different class networks

Figure 5.13 shows messaging between two laptops on different class networks. This demonstrates that the Ethernet bridge truly fulfils its promise of allowing different class networks to communicate to each other. You do not need to have an IP address which is of the same class network to communicate. The only further thing to be done is to specify the default gateway for IP addresses travelling outside the network class as per Figure 5.13.

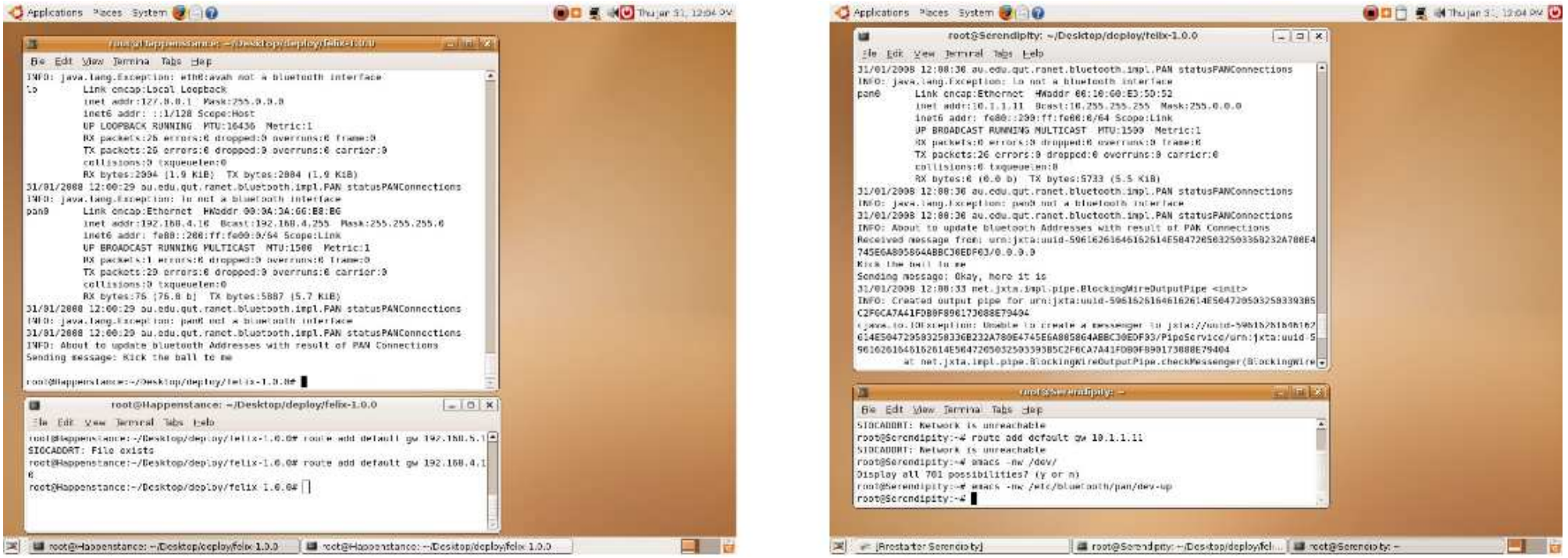


Figure 5.13: Messaging on different class networks

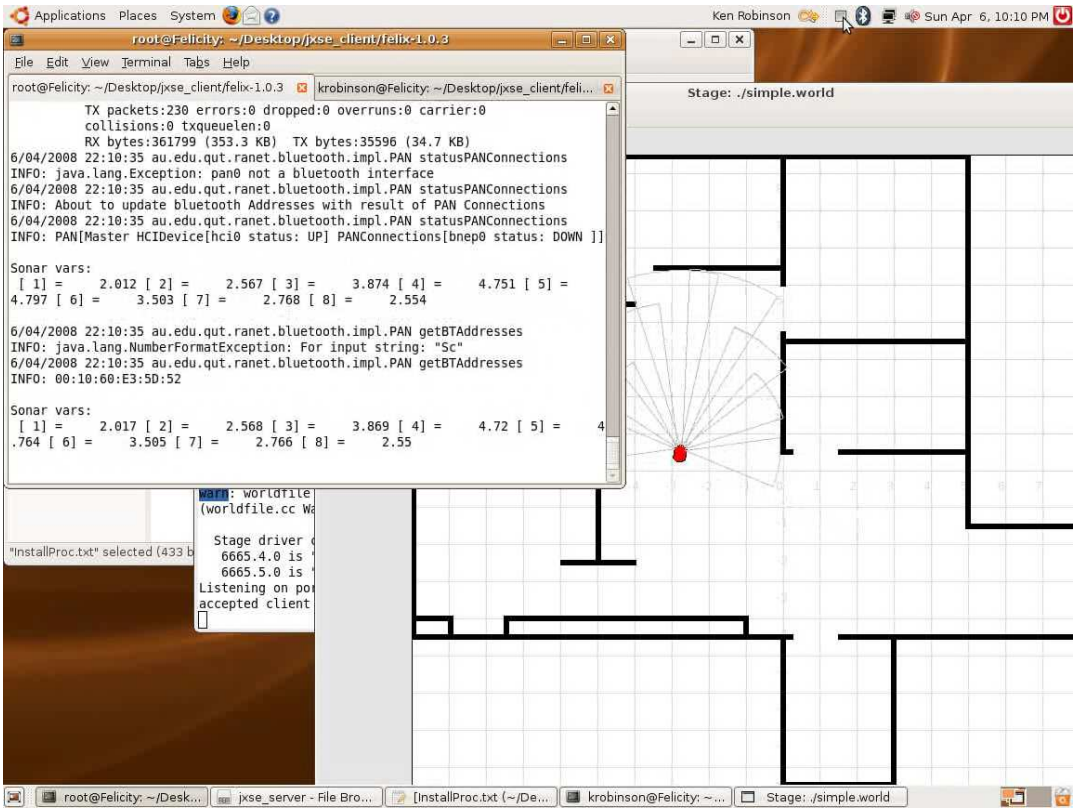


Figure 5.14: Skill transfer on client desktop

5.2.4 Skill transfer between laptop and desktop

In all the preceding tests regarding messaging the client already knows the JXTA advertisement used to connect to the server. Although this makes sense for common services like messaging it will not do for the less needed skills (This comes from the robotic soccer analysis. Most of the work relates to messages sent between each robot. The skills are transferred occasionally.). In those circumstances the client should first probe its environment to see what skills are available. It then should receive the JXTA advertisements showing the skills it is interested in. Lastly, it should then connect to the server having the skill, have it transferred and loaded.

We have demonstrated the concept using a hello skill, being able to print a pretty version of "Hello" (see Appendix J). A better example is needed. We have already referred to Player/Stage/Gazebo in Table 3.2. This is software in C/C++ which has a TCP/IP Server listening for commands which are sent to the robot. This is the Player part. A real robot can be used. Alternatively a simulated robot either in two dimensions (Stage) or three dimensions (Gazebo) can be used. We have used Player here. A Java client has been written for Player/Stage [33] with the same people (specifically Radu Bogdan Rusu) providing a space wanderer example. We have wrapped up the this example in an OSGi bundle along with the Java client libraries it depends upon.

It has been difficult to do video capture of the skill transfer on two separate devices. Accordingly video capture has been done running the client and the server on one laptop (see Figure 5.15) without using a Bluetooth link. Next video capture of the server was done with the client on the desktop and the server on a laptop. The link between them is a Bluetooth link (see Figure 5.14).

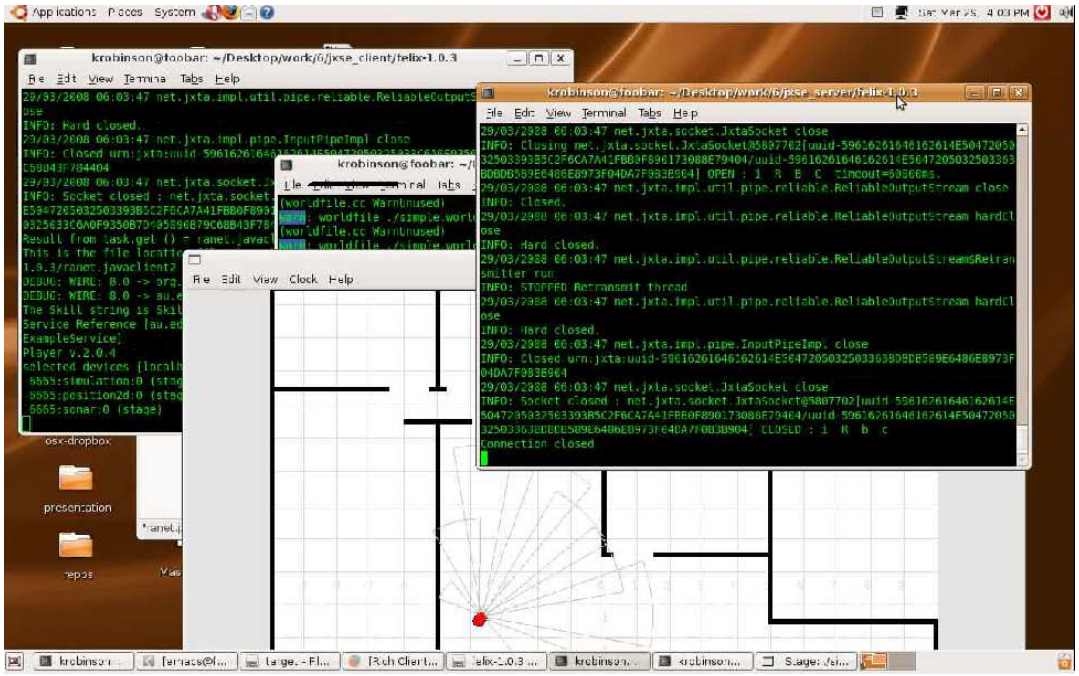


Figure 5.15: Skill transfer on same host

The server OSGi bundles used (see Figure 5.16b) are:

- RANET Server (Activator);
- JXTA Discovery File Server Service (wraps up JXTA Discovery File Server implementation); and
- Bluetooth Service (wraps up Bluetooth implementation).

The client OSGi bundles used (see Figure 5.16a) are:

- RANET Client (Activator);
- JXTA Discovery File Client Service (wraps up JXTA Discovery File Client implementation); and
- Bluetooth Service (wraps up Bluetooth implementation).

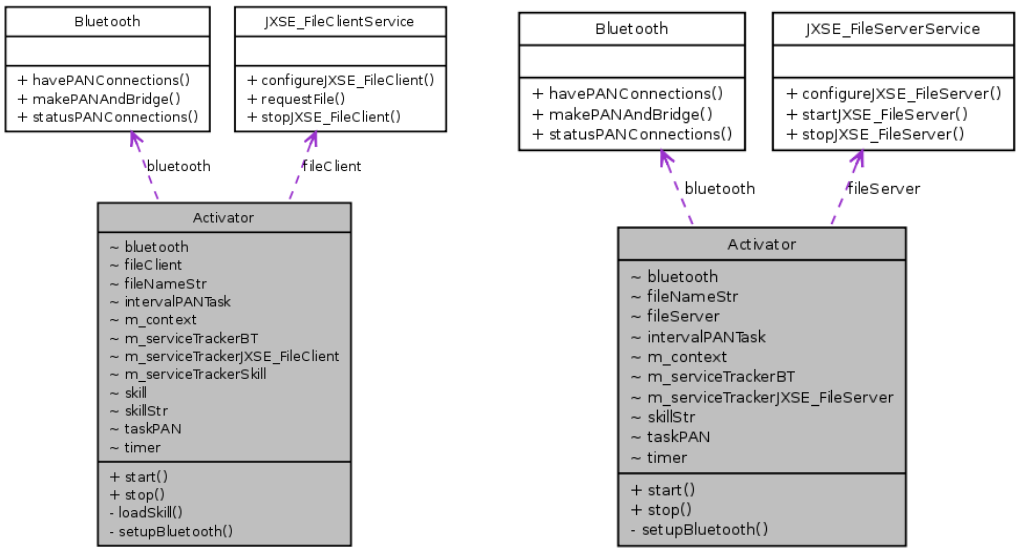
The skill OSGi bundle used is a Wanderer skill which wraps up the space wanderer example.

Bluetooth link established

The skill transfer begins with the client desktop and server laptop starting up. Both the client and the server wait until the Bluetooth service establishes a Bluetooth link (as per Figure 5.9).

Server advertises skill

The JXTA Discovery File Server Service (the server) advertises the Space Wanderer skill in form of a JXTA advertisement (see Figure 5.16b and Figure 5.17). This is done by first



(a) Client (b) Server

Figure 5.16: Skill Transfer Class Diagrams

```
jxta:MSA :
        MSID : urn:jxta:uuid-DDC23AEDFAFF43E98C82F503BFBCF04234\
10BB37AF674FCCBBF8A2DE126CDBF006
        Name : SPEC:Skill=Wanderer
        Crtr :
        SURI :
        Vers : Version 0.1
        jxta:PipeAdvertisement :
                Id : urn:jxta:uuid-59616261646162614E5047205032\
503393B5C2F6CA7A41FBB0F890173088E79404
                Type : JxtaUnicast
                Name : Skill=Wanderer
```

Figure 5.17: JXTA Module Spec Advertisement

```
00166            pipeAdv = createSocketAdvertisement(skillStr);
00167
00168            // set it in the module spec ad
00169            msAdv.setPipeAdvertisement(pipeAdv);
00170
00171            // display the advertisement as a plain text document.
00172            StructuredTextDocument doc =
00173                (StructuredTextDocument)
00174                msAdv.getDocument(MimeMediaType.XMLUTF8);
00175
00176            StringWriter out = new StringWriter();
00177            doc.sendToWriter(out);
00178            System.out.println(out.toString());
00179            out.close();
00180
00181            // publishe locally and remotely
00182            publishModuleSpecAd(msAdv);
00183
00184            serverSocket = new JxtaServerSocket(netPeerGroup, pipeAdv, 10);
00185            serverSocket.setSoTimeout(0);
00186
00187        } catch (IOException e) {
00188            System.out.println("failed to create a server socket");
00189            e.printStackTrace();
00190            System.exit(-1);
00191        }
00192
00193        while (true) {
00194            try {
00195                System.out.println("Waiting for connections");
00196                Socket socket = serverSocket.accept();
00197                if (socket != null) {
00198                    System.out.println("New socket connection accepted");
00199                    Thread thread = new Thread(new ConnectionHandler(socket), "Connection Handler Thread");
00200                    thread.start();
00201                }
```

Figure 5.18: Skill Transfer Server Code Excerpt

making a Class Spec advertisement. From this a Module Class Id is used to create the Module Spec advertisement. Lastly a Pipe advertisement representing a JxtaServerSocket is created and added to the Module Spec advertisement. Note the Name field of both the Module Spec advertisement and the Pipe advertisement contains "Skill=Wanderer". The Module Spec advertisement is published so other devices can see the advertisement and the JxtaServerSocket waits for connections (see Figure 5.18).

Client discovers and obtains skill

The JXTA Discovery File Client Service (the JXTA client) searches for the Module Spec advertisement containing the words "SPEC:Skill=Wanderer". When it finds the Module Spec advertisement the JXTA client extracts the Pipe advertisement (see Figure 5.19). It uses this to connect to JxtaServerSocket. The skill is transferred and saved to disk.

The skill

The Wanderer OSGi bundle implements a skill interface which has a method "invokeSkill()" (see Figure 5.20).

Client loads and uses skill

The RANET Client then loads the skill using the OSGi framework (see Figure 5.16a). It then starts keeping track of any services that are wandering skills. When it finds one, it invokes the skill (see Figure 5.21).

```
00152         System.out.println("we found the service advertisement");
00153         // Ok get the service advertisement as a Spec Advertisement
00154         ModuleSpecAdvertisement msAdv =
00155             (ModuleSpecAdvertisement) en.nextElement();
00156
00157         try {
00158             // let's print the advertisement as a plain text document
00159             StructuredTextDocument doc =
00160                 (StructuredTextDocument) msAdv.getDocument
00161                 (MimeMediaType.TEXT_DEFAULTENCODING);
00162             StringWriter out = new StringWriter();
00163             doc.sendToWriter(out);
00164             System.out.println(out.toString());
00165             out.close();
00166             // we can find the pipe to connect to the service
00167             // in the advertisement.
00168             pipeAdv = msAdv.getPipeAdvertisement();
00169             // Ok we have our pipe advertiseemnt to talk to the service
00170             // create the output pipe endpoint to connect
00171             // to the server, transfer the file
00172
00173         } catch (Exception ex) {
00174             ex.printStackTrace();
00175             System.out.println("Client: Error connecting to service");
00176         }
00177
00178
00179         System.out.println("Connecting to the server");
00180
00181         JxtaSocket socket = null;
00182
00183         while(!socketConnected()) {
00184             try {
00185                 socket = new JxtaSocket(netPeerGroup,
00186                                         // no specific peerid
00187                                         null,
00188                                         pipeAdv,
00189                                         // connection timeout: 5 seconds
00190                                         5000,
00191                                         // reliable connection
00192                                         true);
00193                 setSocketConnected(true);
```

Figure 5.19: Skill Transfer Client Code Excerpt No.1

```
00023 public class Activator implements BundleActivator
00024 {
00025     public void start(BundleContext context)
00026     {
00027         Properties props = new Properties();
00028         props.put("Skill", "Wanderer");
00029         context.registerService(
00030             SpaceWandererExampleService.class.getName(),
00031             new SpaceWandererExampleImpl(),
00032             props);
00033     }
00034
00040     public void stop(BundleContext context)
00041     {
00042         // NOTE: The service is automatically unregistered.
00043     }
00044
00048     public static class SpaceWandererExampleImpl
00049         implements SpaceWandererExampleService, Skill
00050     {
00051         public void invokeSkill() {
00052             begin();
00053         }
00054
00055         public void begin()
00056         {
00057             SpaceWandererExample ex =
00058                 new SpaceWandererExample();
00059             ex.begin();
00060         }
00061     }
```

Figure 5.20: Wanderer Skill Code Excerpt

```
00224         loadSkill(file, context);
00225
00226         String filterSkill = "(Skill=" + skillStr + ")";
00227         m_serviceTrackerSkill =
00228            new ServiceTracker(m_context,
00229                               m_context.createFilter(filterSkill),
00230                               null);
00231         m_serviceTrackerSkill.open();
00232
00233         try {
00234             while (true) {
00235                 skill =
00236                     m_serviceTrackerSkill.waitForService(1000); // 1 sec
00237                 if (skill == null) {
00238                     System.out.println("Did not get the skill");
00239
00240                 } else {
00241                     Method reflectedSkill = skill.getClass().getMethod("invokeSkill",
00242                                                                        null);
00243                     reflectedSkill.invoke(skill, null);
00244                 }
00245             }
00246
00247         } catch (Exception e) {
00248             e.printStackTrace();
00249             System.exit(-1);
00250         }
00251     }
00252
00253     private void loadSkill(java.io.File file, BundleContext context) {
00254
00255         String fileURLString = null;
00256
00257         try {
00258             fileURLString = file.toURI().toURL().toString();
00259             System.out.println("This is the file location " + fileURLString);
00260             Bundle bundle = context.installBundle(fileURLString);
00261             bundle.start();
00262         } catch (java.net.MalformedURLException e) {
00263             System.out.println("Bad URL " + fileURLString);
00264             e.printStackTrace();
00265
```

Figure 5.21: Skill Transfer Client Code Excerpt No.2

Chapter 6

CONCLUSIONS

Chapter Summary - We conclude, outlining the novel aspects of the final design, look at some future work and investigation and give an overall view of the process of designing the RANET

6.1 Summary

In this thesis we have designed and tested a peer to peer network called RANET that allows members to:

- Form ad hoc networks through a Bluetooth TCP/IP link that is constantly sought after and maintained by each RANET member (see Section 4.2.1;
- Send and receive messages via a known JXTA pipe to each other or through another RANET member, even if the other RANET member is on a different class of network (see Sections 5.2.1, 5.2.2 and 5.2.3); and
- Discover skill on other RANET members, have one of those skills transferred to themselves, load that skill and use it (see Section 5.2.4).

The integration of the components has been difficult and time consuming.

The Linux kernel on the Korebot did not have the necessary functionality for BlueZ, Ethernet bridging and an earlier choice for the Bluetooth transceiver. Kernel patches and extra libraries and utilities had to be added (see Sections 4.1.3 and Appendix C).

Finding a fully functional Java virtual machine freely available for an ARM processor (given its use in embedded devices) was difficult with both Sun virtual machines not able to be ported either through difficulty in the codebase or the United States export restriction laws (see Section E.2). The open source Java virtual machine (JamVM) and Java libraries (GNU Classpath - GCP) worked up until the JXTA and P2PS peer to peer protocol libraries were used. Essentially JXTA required Java version 1.5.0 and JamVM/GCP only had part of the 1.5.0 Java functionality available on the Korebot (see Sections 5.1 and E.3. For P2PS, the JamVM/GCP did not have certain UDP multicast functionality (see Sections 5.1 and I.3).

The Bluetooth PCMCIA card recommended by the Korebot vendor did not work, even with patches applied (see Sections 4.1.3, D.1 and D.2). It was only through examination of the Korebot that a USB master connection was found and the idea of using a Bluetooth USB dongle was thought of (see Section D.3).

Using Jython pickled objects as a way of packing and loading skills did not work as Jython failed to work on JamVM/GCP on the Korebot (see Appendix G). OSGi did work reasonably well from the very first.

JXME did not work as advertised. With JXSE It was not until version 2.5 that it was fairly intuitive and easy to use (see Section 5.1 and Appendix I).

To conclude, some open source software components do not have sufficient maturity or stability. Some frameworks over complicate things, where simplicity would be preferred. Incomplete documentation is the norm. However the support provided by the Korebot vendor, the makers of Apache Felix OSGi and the author of JamVM was very good.

6.2 Novel Aspects of Design

It is important to highlight the novel aspects of this design. They are the adhoc nature of the Bluetooth TCP/IP link, the provision of wireless services containing skills and the use of the Ethernet bridge on all devices.

Previously with Bluetooth connections the device was either a master or a slave. As a master it could only accept connections. As a slave it could only seek connections. This meant a device needed to have prior knowledge of the other device it was trying to connect to. Potentially one could have collections of masters or slaves close to other, wanting but unable to communicate. Now with the extension to the PAN Profile devices periodically flip between master and slave (see Section 4.2.1 and pages 52 and 52). This means that eventually all devices will become part of the network. As discussed previously, this could work on scatternets as well, as each slave still tries to connect even after it has made a connection with a master. The slave would be members or two or more piconets and so gang together piconets into a wider scatternet. If a robot was to break the link by moving too far away from the device, upon coming back in contact it would eventually make the link once more.

Wireless services in the form of messaging or skills is available to members of a RANET. This is done by coupling the OSGi framework with JXTA. Now robots can search for JXTA advertised messaging services. Similarly they can search for JXTA advertised OSGi skills which can be transferred to the seeker, loaded and invoked. These bundles are remembered by the OSGi framework; effectively it loads and remembers them.

The Ethernet bridge was previously only used on master devices in the Bluetooth PAN profile (see Section 4.2.1 and Appendix D.4). Adding the Ethernet bridge to slaves allows them to have more than one Ethernet link, thereby allowing scatternets. The use of the Ethernet bridge allows searching for skills, messaging and skill transfer to occur across multiple nodes in the RANET (see Section 5.2.2). This has demonstrated that multi-hop skill queries, messaging and skill transfer on a piconet can occur. Also devices in a RANET do not need need to have an IP address which is a member of the same network class (see Section 5.2.3). This means that each device that wishes to participate in a RANET can have an arbitrary IP address.

6.3 Recommendations for improvement

Deficiencies in some third party software were found, specifically JXTA and JamVM/GNU Classpath. These deficiencies could be fixed by porting the JXTA C implementation to the target embedded platform and potentially eliminating the TCP/IP layer or using UDP instead of TCP. Unlike Java which is dependant on the virtual machine, the C libraries would be the same across different machines. This would work on the Korebot and its ARM processor.

Eliminating TCP/IP means writing a JXTA implementation which uses Bluetooth. No such implementation exists. However the advantages would be great. With Bluetooth

Device discovery is built into the protocol, Automatic retransmission request (ARQ) of Bluetooth packets occur if they are corrupted. Built in redundancy gives a greater probability that packets will be received. Communication occurs over dedicated connections, eliminating both the exposed and hidden terminal problems[51]. It eliminates problems with TCP limiting the size of the packets when there is no need, say through temporary signal fade. It also allows for a somewhat higher data rate.

Using UDP instead of TCP would eliminate congestion problems that may occur because of signal fade. Alternatively an adaptive TCP/IP stack could be used to eliminate such congestion.

If STP were turned on in the Ethernet bridge, multi-hop skill queries, messaging and skill transfer could occur across RANET scatternets. The STP would need to turned on to prevent cycles from occurring and stop flooding of the network. How it would contribute to traffic on the network (it periodically exchanges Bridge Protocol Data Units) would need to be considered.

Future investigation could occur on how to seed the configuration for the extension to the PAN Bluetooth protocol so a Bluetooth TCP/IP link is more quickly formed. Also an extension to the JXTA Client could allow them to readvertise the skill they had just received, thus enabling transfer of the skill throughout the network.

6.4 Some Observations

We believe that JXTA is too complex. More simple peer to peer protocols need to be used with the capacity to turn on and off features like security and authentication or at least allowing for different means to authenticate. P2PS has this feature. But for the fact that the open source Java environment did not had UDP datagrams the P2PS libraries would have worked on the Korebot.

For embedded programming one needs more immediate feedback. Programming on a desktop computer, transferring the jar files to the target and then running them is a slow process especially through a serial connection (see Appendix B). A programming language where there is an interpreter which allows a researcher to try things at will on the target computer would speed up progress. Higher level constructs than Java is capable of would also be helpful. Both these features are found in languages like Lisp, Python, Ruby, Erlang, etc. Further, distribution and fault tolerance are not built into the Java language as they are in Erlang and Mozart/Oz. In particular, Erlang treats talking to a local node on the same device the same way as a remote device [3, pages 167-177]. Maybe the problem is we are trying to implement a P2P system in the wrong domain.

Appendix A

Robot Soccer Analysis

An excel spreadsheet was done in the preliminary stages of the analysis of robotic soccer. A screenshot is shown below (See Figure A). With the electronic copy of this document is a file "SystemAnalysis.xls". This can be opened up using Microsoft Office.

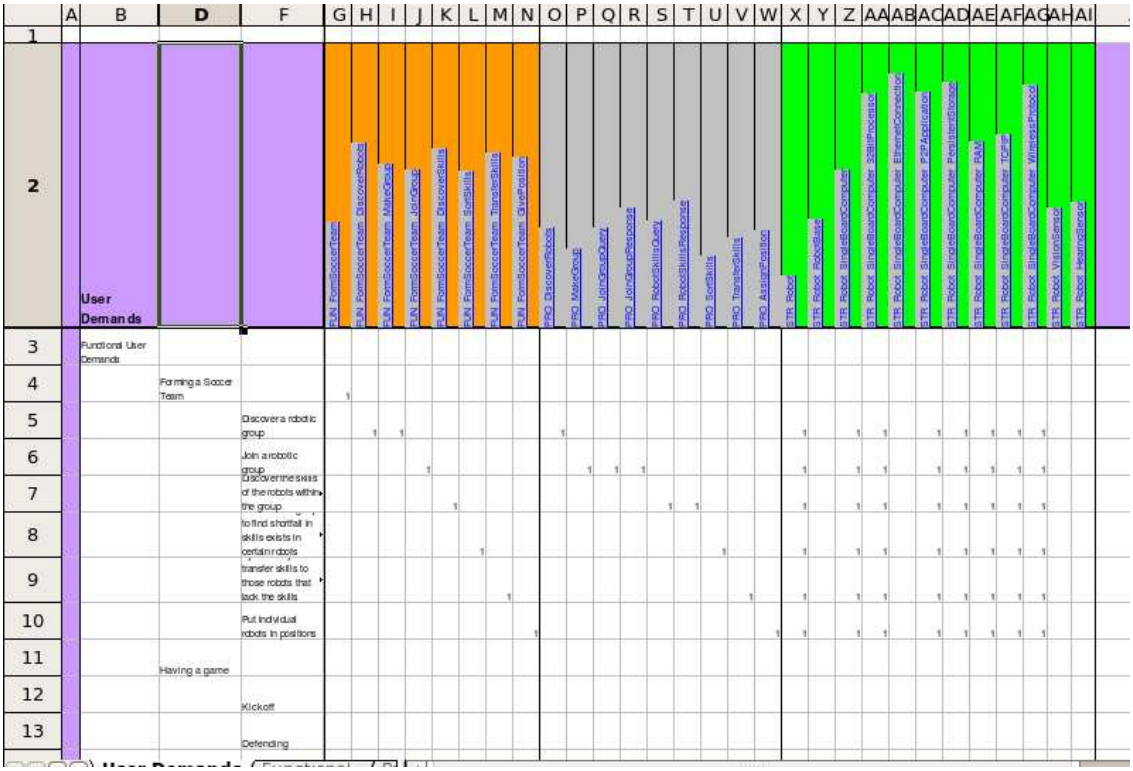


Figure A.1: Robot Soccer Analysis

Appendix B

Korebot

This is taken from the Korebot.UserManual downloadable from K-Team http://www.k-team.com. Use the terminal program minicom with settings 115200 Bps, 8 data bits, No parity and 2 stop bits. You should get the startup sequence as below. The output here is after kernel patch as per the Linux appendix so there is a bad flag message which you can safely ignore. In amongst all those messages you should see the login prompt. Type "root" for user and then "rootme" for the password. You should then get a login prompt.

```
flasd memory
FlaBoot - Minimal Flash Booter
(c) 2002 - 2004 Stephane Magnenat & Julien Pilet
This program is GPL
KoreBot ROM, serial #00000001
Now loading...
Now booting...
Uncompressing Linux.............................\
....................... done,.
Linux version 2.4.19-kb9 (pbureau@koala33.k-teami.com)\
 (gcc version 2.95.3 20014
CPU: XScale-PXA255 revision 6

...

Mounting local filesystems...
Setting the System Clock using the Hardware Clock as \
reference...
INIT: Entering runlevel: 2
Enabling memory overcommit
Configuring network interfaces: done.
Not recalculating module dependencies
Loading modules: pxa-uda1342-i2c otg243
Starting OpenBSD Secure Shell server: sshd.
Starting syslogd/klogd: done
Starting PCMCIA services: cardmgr.
cardmgr[195]: watching 2 sockets
Uniform Multi-Platform E-IDE driver Revision: 6.31
ide: Assuming 50MHz system bus speed for PIO modes; \
override with idebus=xx
korebot_pcmcia_configure_socket(): Requesting Vcc 33 \
 and Vpp 0 for socket 1.
korebot_pcmcia_configure_socket(): Reset off sock 1
korebot_pcmcia_configure_socket(): Requesting Vcc 33 \
and Vpp 0 for socket 1.
korebot_pcmcia_configure_socket(): Reset off sock 1
Trying to free nonexistent resource <f7000000-f700000f>
```

```
hda: SanDisk SDCFB-256, ATA DISK drive
ide0 at 0xf7000000-0xf7000007,0xf700000e on irq 37
ide_cs: hda: Vcc = 3.3, Vpp = 0.0

korebot login: hda: bad special flag: 0x03
hda: 501760 sectors (257 MB) w/1KiB Cache, CHS=980/16/32
Partition check:
 /dev/ide/host0/bus0/target0/lun0: p1
 /dev/ide/host0/bus0/target0/lun0: p1
 /dev/ide/host0/bus0/target0/lun0: p1
pc : [<40107354>]    lr : [<0001f270>]    Not tainted
sp : bffffb38  ip : 0001f271  fp : 00088d88
r10: 401b4950  r9 : 00086d68  r8 : bffffba4
r7 : 00000001  r6 : 000a8000  r5 : 401b5508  r4 : 00088d98
r3 : 00000000  r2 : 00000049  r1 : 00088d98  r0 : 00088d90
Flags: nZCv  IRQs on  FIQs on  Mode USER_32  Segment user
Control: 397F  Table: A3C50000  DAC: 00000015
```

Appendix C

Linux

This appendix covers the patching and configuration of the Linux kernel on the Korebot so it runs the BlueZ stack, the Ethernet Bridge and Multicast functionality. It also covers the various Bluetooth utilities which allow a TCP/IP link to be made. Note that you require a Linux PC (The Test Computer) with the latest BlueZ stack. This is so the Korebot can inter-operate with another BlueZ device. You also need two USB Bluetooth dongles and the Korebot software CD. A mini USB adapter is also required to connect the USB Bluetooth dongle to the Korebot. All the software was cross-compiled with a 2.95.3 gcc toolchain. Later toolchain versions may not require the patches mentioned below.

For the BluetoothImpl to work the Bluez kernel, libraries and utilities are required from the Bluez website.

C.1 Bluez kernel

C.1.1 Applying the Patch

The Korebot toolchain (provided on the CD that comes with the Korebot or downloaded from the K-Team website) needs to be unpacked. The Linux kernel that has been modified for the Korebot also needs to obtained (again from the CD or the website) and unpacked. Change directories so you are in the unpacked directory linux-2.4.19-kb11.

The patch from patch-2.4.19-mh18.gz on the BlueZ web site (URL www.bluez.org) is required. Apply the patch using the command “patch -p1 -i ../patch-2.4.19-mh18”. Ignore the first patch to the file. It relates in fact to some Solaris I/O as it refers to a SPARC processor. The sequence is outline below

```
\krobinson@Possibility:~/sw/linux-2.4.19-kb11$ patch -p1 -i ../patch-2.4.19-mh18
can't find file to patch at input line 4
Perhaps you used the wrong -p or --strip option?
The text leading up to this was:
--------------------------
|diff -urN linux-2.4.19-kb11/arch/arm/kernel/ioctl32.c \
linux-2.4.19-mh18/arch/sparc64/kernel/ioctl32.c
|--- linux-2.4.19-kb11/arch/arm/kernel/ioctl32.c \
 2002-08-03 02:39:43.000000000 +0200
|+++ linux-2.4.19-mh18/arch/arm/kernel/ioctl32.c \
 2004-08-01 16:29:19.000000000 +0200
--------------------------
File to patch:
Skip this patch? [y] y
Skipping patch.
4 out of 4 hunks ignored
```

```
patching file CREDITS
Hunk #2 succeeded at 2593 (offset 9 lines).
patching file Documentation/Configure.help
Hunk #1 succeeded at 11360 (offset 72 lines).
Hunk #2 succeeded at 13313 (offset 108 lines).
Hunk #3 succeeded at 20587 (offset 167 lines).
Hunk #4 succeeded at 20614 (offset 167 lines).
Hunk #5 succeeded at 20682 (offset 167 lines).
Hunk #6 succeeded at 20711 (offset 167 lines).
Hunk #7 succeeded at 20728 (offset 167 lines).
patching file Documentation/devices.txt
patching file Documentation/firmware_class/firmware_sample_driver.c
patching file Documentation/firmware_class/hotplug-script
patching file Documentation/firmware_class/README
patching file drivers/bluetooth/bfusb.c
patching file drivers/bluetooth/bluecard_cs.c
patching file drivers/bluetooth/bt3c_cs.c
patching file drivers/bluetooth/btuart_cs.c
patching file drivers/bluetooth/Config.in
patching file drivers/bluetooth/dtl1_cs.c
patching file drivers/bluetooth/hci_bcsp.c
patching file drivers/bluetooth/hci_bcsp.h
patching file drivers/bluetooth/hci_h4.c
patching file drivers/bluetooth/hci_h4.h
patching file drivers/bluetooth/hci_ldisc.c
patching file drivers/bluetooth/hci_uart.h
patching file drivers/bluetooth/hci_usb.c
patching file drivers/bluetooth/hci_usb.h
patching file drivers/bluetooth/Makefile
patching file drivers/bluetooth/Makefile.lib
patching file drivers/char/pcmcia/serial_cs.c
patching file drivers/char/serial.c
Hunk #1 succeeded at 860 (offset 18 lines).
patching file drivers/input/Config.in
patching file drivers/input/keybdev.c
patching file drivers/input/Makefile
patching file drivers/input/uinput.c
patching file drivers/isdn/avmb1/capidrv.c
patching file drivers/isdn/avmb1/kcapi.c
patching file drivers/usb/Config.in
Hunk #1 succeeded at 41 (offset 9 lines).
patching file drivers/usb/hid-core.c
patching file include/linux/firmware.h
patching file include/linux/input.h
patching file include/linux/net.h
patching file include/linux/uinput.h
patching file include/net/bluetooth/bluetooth.h
patching file include/net/bluetooth/hci_core.h
patching file include/net/bluetooth/hci.h
patching file include/net/bluetooth/l2cap.h
patching file include/net/bluetooth/rfcomm.h
patching file include/pcmcia/ciscode.h
patching file kernel/ksyms.c
Hunk #2 succeeded at 552 (offset 2 lines).
patching file lib/Config.in
patching file lib/firmware_class.c
patching file lib/Makefile
patching file MAINTAINERS
Hunk #1 succeeded at 274 (offset 14 lines).
patching file net/bluetooth/af_bluetooth.c
patching file net/bluetooth/bnep/bnep.h
patching file net/bluetooth/bnep/Config.in
```

```
patching file net/bluetooth/bnep/core.c
patching file net/bluetooth/bnep/crc32.c
patching file net/bluetooth/bnep/crc32.h
patching file net/bluetooth/bnep/Makefile
patching file net/bluetooth/bnep/netdev.c
patching file net/bluetooth/bnep/sock.c
patching file net/bluetooth/cmtp/capi.c
patching file net/bluetooth/cmtp/cmtp.h
patching file net/bluetooth/cmtp/Config.in
patching file net/bluetooth/cmtp/core.c
patching file net/bluetooth/cmtp/Makefile
patching file net/bluetooth/cmtp/sock.c
patching file net/bluetooth/Config.in
patching file net/bluetooth/hci_conn.c
patching file net/bluetooth/hci_core.c
patching file net/bluetooth/hci_event.c
patching file net/bluetooth/hci_sock.c
patching file net/bluetooth/hidp/Config.in
patching file net/bluetooth/hidp/core.c
patching file net/bluetooth/hidp/hidp.h
patching file net/bluetooth/hidp/Makefile
patching file net/bluetooth/hidp/sock.c
patching file net/bluetooth/l2cap.c
patching file net/bluetooth/Makefile
patching file net/bluetooth/rfcomm/Config.in
patching file net/bluetooth/rfcomm/core.c
patching file net/bluetooth/rfcomm/crc.c
patching file net/bluetooth/rfcomm/Makefile
patching file net/bluetooth/rfcomm/sock.c
patching file net/bluetooth/rfcomm/tty.c
patching file net/bluetooth/sco.c
patching file net/bluetooth/syms.c
patching file net/netsyms.c
```

C.1.2 Editing the Config

With the kernel you have a configuration file "config-korebot" which you copy to ".config" and then modify as per the diff below. This should enable the BlueZ options. Essentially all the BNEP options should be turned on. These are used by the PAN to get the TCP/IP connection.

```
diff -c /home/krobinson/sw/linux-2.4.19-kb11/.config \
/home/krobinson/sw/linux-2.4.19-kb11/config-korebot
*** /home/krobinson/sw/linux-2.4.19-kb11/.config     \
   Wed May 17 11:54:09 2006
--- /home/krobinson/sw/linux-2.4.19-kb11/config-korebot \
Wed May 17 11:54:09 2006
***************
*** 1063,1096 ****
  # CONFIG_USB_AUERSWALD is not set
  # CONFIG_USB_BRLVGER is not set

-
  #
  # Bluetooth support
  #
  CONFIG_BLUEZ=m
  CONFIG_BLUEZ_L2CAP=m
  CONFIG_BLUEZ_SCO=m
- CONFIG_BLUEZ_RFCOMM=m
- CONFIG_BLUEZ_RFCOMM_TTY=y
```

```
- CONFIG_BLUEZ_BNEP=m
- CONFIG_BLUEZ_BNEP_MC_FILTER=y
- CONFIG_BLUEZ_BNEP_PROTO_FILTER=y
- CONFIG_BLUEZ_HIDP=m

  #
  # Bluetooth device drivers
  #
  CONFIG_BLUEZ_HCIUSB=m
! # CONFIG_BLUEZ_HCIUSB_SCO is not set
  CONFIG_BLUEZ_HCIUART=m
  CONFIG_BLUEZ_HCIUART_H4=y
- CONFIG_BLUEZ_HCIUART_BCSP=y
- CONFIG_BLUEZ_HCIBFUSB=m
  CONFIG_BLUEZ_HCIDTL1=m
! CONFIG_BLUEZ_HCIBT3C=m
! CONFIG_BLUEZ_HCIBLUECARD=m
! CONFIG_BLUEZ_HCIBTUART=m
! CONFIG_BLUEZ_HCIVHCI=m

  #
  # Kernel hacking
--- 1063,1085 ----
  # CONFIG_USB_AUERSWALD is not set
  # CONFIG_USB_BRLVGER is not set

  #
  # Bluetooth support
  #
  CONFIG_BLUEZ=m
  CONFIG_BLUEZ_L2CAP=m
  CONFIG_BLUEZ_SCO=m

  #
  # Bluetooth device drivers
  #
  CONFIG_BLUEZ_HCIUSB=m
! CONFIG_BLUEZ_USB_FW_LOAD=y
! CONFIG_BLUEZ_USB_ZERO_PACKET=y
  CONFIG_BLUEZ_HCIUART=m
  CONFIG_BLUEZ_HCIUART_H4=y
  CONFIG_BLUEZ_HCIDTL1=m
! # CONFIG_BLUEZ_HCIVHCI is not set

  #
  # Kernel hacking
```

C.1.3 Making the Kernel

Make the kernel image using "make dep", "make bzImage", "make modules" and "make modules_install INSTALL_MOD_PATH=../tmp" in the top directory of 2.4.19-kb11 directory. This makes the image and the modules as per 6.4 of the KoreBot.UserManual. Then make a bootable image using the toolchain by putting in "/usr/local/korebot-tools-0.1.2/src/flaboot-0.2.1" the image and running "make".

C.1.4 Installing the Kernel

The 2.4.19 Korebot kernel does not work properly with 2.6.x kernels. It was found that the message "usbep0: setup begin: zero-length OUT?" kept repeating when an ethernet

connection through the USB port was tried. The work around is transferring files via minicom using the file transfer utility.

You need to zip or tar up files before sending them across via minicom. Unfortunately the tar utility on Korebot is broken. The easiest solution is to use an open source utility Info-Zip Unzip from http://www.info-zip.org/pub/infozip/UnZip.html. You need to change the Makefile at unix/Makefile to have arm-linuz-gcc and arm-linux-as the gcc toolchain to use. Cross compile and transfer it across.

Zip up all the files for the kernel and send them across. As per the Korebot user manual do "eraseall /dev/mtd/0" and then copy flaboot_korebot.bin to /dev/mtd/0. This erases the old kernel and puts the new kernel in its place.

C.1.5 Testing the BlueZ in the kernel

The drivers which are meant to be used with Bluetooth PCMCIA card are immature and at least for the card chosen do not work. Use the USB Bluetooth dongles. The drivers for them are mature and work for most USB Bluetooth dongles. On the Korebot there are two mini master USB connections. Use either of those. You may need to get an adapter as most USB Bluetooth dongles do not have a mini USB connection.

After the USB Bluetooth dongle is inserted in the Korebot insert a similar dongle in your Test Computer. Test that it works by doing the following commands.

Korebot

This tests the Bluetooth USB driver.

```
# modprobe hci_usb
Using /lib/modules/2.4.19-kb11/kernel/net/bluetooth/bluez.o
BlueZ Core ver 2.4 Copyright (C) 2000,2001 Qualcomm Inc
Written 2000,2001 by Maxim Krasnyansky <maxk@qualcomm.com>
Using /lib/modules/2.4.19-kb11/kernel/drivers/bluetooth/hci_usb.o
BlueZ HCI USB driver ver 2.7 Copyright (C) 2000,2001 Qualcomm Inc
Written 2000,2001 by Maxim Krasnyansky <maxk@qualcomm.com>
usb.c: registered new driver hci_usb
```

See the state of hciconfig before bringing up the hci0 interface.

```
# modprobe hci\_usb
hci0:   Type: USB
        BD Address: 00:00:00:00:00:00 ACL MTU: 0:0  SCO MTU: 0:0
        DOWN
        RX bytes:0 acl:0 sco:0 events:0 errors:0
        TX bytes:0 acl:0 sco:0 commands:0 errors:0
```

Now bring up the interface.

```
# hciconfig hci0 up
hci0:   Type: USB
        BD Address: 00:02:72:B0:C5:FB ACL MTU: 192:8  SCO MTU: 64:8
        UP RUNNING PSCAN ISCAN
        RX bytes:69 acl:0 sco:0 events:8 errors:0
        TX bytes:27 acl:0 sco:0 commands:7 errors:0
```

Lastly load the BNEP module "bnep".

```
# modprobe bnep
Using /lib/modules/2.4.19-kb11/kernel/net/bluetooth/l2cap.o
BlueZ L2CAP ver 2.3 Copyright (C) 2000,2001 Qualcomm Inc
Written 2000,2001 by Maxim Krasnyansky <maxk@qualcomm.com>
Using /lib/modules/2.4.19-kb11/kernel/net/bluetooth/bnep/bnep.o
BlueZ BNEP ver 1.2
Copyright (C) 2001,2002 Inventel Systemes
Written 2001,2002 by Clement Moreau <clement.moreau@inventel.fr>
Written 2001,2002 by David Libault <david.libault@inventel.fr>
Copyright (C) 2002 Maxim Krasnyanskiy <maxk@qualcomm.com>
```

Test Computer

Repeat the above commands.

Korebot

Run a hcitool dev command. This will show you the local Bluetooth devices.

```
# hcitool dev
Devices:
        hci0    00:02:72:B0:C5:FB
```

Now run a hcitool scan command. This will hopefully show the Test Computer Bluetooth address.

```
# hcitool scan
Scanning ...
        00:0A:3A:66:B8:B6       Possibility-0
```

C.2 Bluez libraries and utilities

C.2.1 Libraries

BlueZ libraries are cross-compiled with the following commands.

```
#./configure --host=arm-linux --target=arm-linux --prefix=/home/krobinson/bluez-libs-2.25/tmp
# make
# make install
Libraries have been installed in:
   /home/krobinson/sw/bluez-libs-2.25/tmp/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the '-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the 'LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the 'LD_RUN_PATH' environment variable
     during linking
   - use the '-Wl,--rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to '/etc/ld.so.conf'
```

C.2.2 Utilities

BlueZ utilities like "pand" which are required for the Bluetooth network. Within the bluez-utils there is an error that occurs at least when compiling with the 2.95.3 gcc

toolchain. Found that you need to add "#include ¡ sys/param.h ¿ " at the top of each file which complains that the PATH_MAX variable is undefined.

Cross compile bluez-utils by running the following commands. The "–with-bluez" option is where the BlueZ libraries are located.

```
# ./configure --host=arm-linux --target=arm-linux \
--prefix=/home/krobinson/sw/bluez-utils-2.25/tmp \
--with-bluez=/home/krobinson/sw/bluez-libs-2.25/tmp
# make
# make install
```

C.2.3 Transfer libraries and utilities

The libraries and utilities are then transferred to the Korebot and put under "/usr/local".

C.3 Ethernet Bridge

The ethernet bridge options can be set as a module to be linked into the kernel instead as part of the kernel. Unlike most modules, the Linux kernel needs to be recompiled for the ethernet bridge to work. I discovered through old posts that one needs to run the command "make mrproper". You want to delete only the object files. As this command deletes the .config file and all the old object files, save old config first as "myconfig" or the like and then run the command.

C.3.1 Kernel

```
# mv .config myconfig
# make mrproper
```

Rename the "myconfig" back to to old config.

```
# mv myconfig .config
```

You then want to make some new configuration for the ethernet but keep all the options in the .config.

This is done by running the command "make oldconfig" which prompts you to answer questions if it does not find the answer in the .config file or the /arch/arm/defconf file.

```
# make oldconfig
```

The kernel is then made using "make dep" and "make bzImage". The modules are made using "make modules".

```
# make dep
# make bzImage
# make modules
```

Copy the image to flaboot in the Koreboot cross toolchain and run make.

```
# cp zImage /usr/local/korebot-tools-0.1.2/src/flaboot-0.2.1
# make
```

Transfer flaboot_korebot.bin to the Korebot. Then erase the previous kernel, install the new kernel and shutdown.

```
# eraseall /dev/mtd/0
# mv flaboot_korebot.bin to /dev/mtd/0.
# halt
```

The new kernel should take effect on startup.

C.3.2 Utilities

The utility "brctl" is still required and so configure the bridge.

```
krobinson@Possibility:~/sw/bridge-utils-1.1$ ./configure --host=arm-linux --target=arm-linux --prefix=/home/krob
...
config.status: creating libbridge/config.h
config.status: executing depfiles commands
```

And now make it.

```
krobinson@Possibility:~/sw/bridge-utils-1.1$ make
for x in libbridge brctl doc; do make  -C $x ; done
make[1]: Entering directory `/home/krobinson/sw/bridge-utils-1.1/libbridge'
arm-linux-gcc -Wall -g -I/usr/src/linux/include  -c libbridge_devif.c
arm-linux-gcc -Wall -g -I/usr/src/linux/include  -c libbridge_if.c
arm-linux-gcc -Wall -g -I/usr/src/linux/include  -c libbridge_init.c
arm-linux-gcc -Wall -g -I/usr/src/linux/include  -c libbridge_misc.c
ar rcs libbridge.a libbridge_devif.o libbridge_if.o libbridge_init.o libbridge_misc.o
arm-linux-ranlib libbridge.a
make[1]: Leaving directory `/home/krobinson/sw/bridge-utils-1.1/libbridge'
make[1]: Entering directory `/home/krobinson/sw/bridge-utils-1.1/brctl'
arm-linux-gcc -Wall -g -O2 -I../libbridge -I/usr/src/linux/include  -c brctl.c
arm-linux-gcc -Wall -g -O2 -I../libbridge -I/usr/src/linux/include  -c brctl_cmd.c
arm-linux-gcc -Wall -g -O2 -I../libbridge -I/usr/src/linux/include  -c brctl_disp.c
arm-linux-gcc  brctl.o brctl_cmd.o brctl_disp.o -L ../libbridge -lbridge  -o brctl
make[1]: Leaving directory `/home/krobinson/sw/bridge-utils-1.1/brctl'
make[1]: Entering directory `/home/krobinson/sw/bridge-utils-1.1/doc'
make[1]: Nothing to be done for `all'.
make[1]: Leaving directory `/home/krobinson/sw/bridge-utils-1.1/doc'
```

And now install it. The libraries "libbridge.a", "libbridge.so" and the utility "brctl" need to be transferred to the Korebot.

```
krobinson@Possibility:~/sw/bridge-utils-1.1$ make install
for x in libbridge brctl doc; do make  -C $x install; done
make[1]: Entering directory `/home/krobinson/sw/bridge-utils-1.1/libbridge'
mkdir -p /home/krobinson/sw/bridge-utils-1.1/tmp/include
install -m 644 libbridge.h /home/krobinson/sw/bridge-utils-1.1/tmp/include
mkdir -p /home/krobinson/sw/bridge-utils-1.1/tmp/lib
install -m 644 libbridge.a /home/krobinson/sw/bridge-utils-1.1/tmp/lib
make[1]: Leaving directory `/home/krobinson/sw/bridge-utils-1.1/libbridge'
make[1]: Entering directory `/home/krobinson/sw/bridge-utils-1.1/brctl'
mkdir -p /home/krobinson/sw/bridge-utils-1.1/tmp/sbin
/usr/bin/install -c -m 755 brctl  /home/krobinson/sw/bridge-utils-1.1/tmp/sbin
make[1]: Leaving directory `/home/krobinson/sw/bridge-utils-1.1/brctl'
make[1]: Entering directory `/home/krobinson/sw/bridge-utils-1.1/doc'
mkdir -p /home/krobinson/sw/bridge-utils-1.1/tmp/man/man8
install -m 644 brctl.8 /home/krobinson/sw/bridge-utils-1.1/tmp/man/man8
make[1]: Leaving directory `/home/krobinson/sw/bridge-utils-1.1/doc'
```

C.3.3 Integrating ethernet bridge and Bluetooth

For new instances of bnep to be added to the bridge, a dev-up script needs to be added to "/etc/bluetooth/pan/dev-up". Note that for versions 3.9 and above of bluez-utils a bug you require to specify where the "dev-up" script is by using the option "–devup ¡ path to dev up script ¿ / dev-up" when using "pand".

Dev up script used from HowTo PAN for Bluetooth connection.

```
# cat /etc/bluetooth/pan/dev-up
#!/bin/sh
# 'dev-up' script to do dynamic bridge management
brctl addif pan0 $1   # $1 is the new name, passed by 'pand'
ifconfig $1 0.0.0.0
```

C.4 Testing Ethernet Bridge and Bluetooth

We now test that everything is working. On the target Korebot take a usbf interface down, bring up the Bluetooth interface at the hardware level, add an interface to the bridge, bring up the bridge interface and then listen for Bluetooth connections.

```
#ifconfig usbf down
#ifconfig
#hciconfig hci0 up
#hciconfig
#brctl addbr pan0
#brctl setfd pan0 0
#brctl stp pan0 off
#ifconfig pan0 192.168.1.1
#pand --listen --role GN
#brctl addif pan0 bnep0
```

On the host Linux computer become root, do similar tasks, scan for the Bluetooth address, try to connect repeatedly using the Bluetooth address that comes up as a result of the scan. Test then whether the Bluetooth interface comes up.

```
#su (and password)
#hcidump (show details of connections)
#modprobe hci_usb
#modprobe bnep
#hcitool scan
#pand --connect <BTAddress> (have to repeatedly try before success).
```

C.5 Kernel Configuration

Configuration file for the kernel is as follows:

```
#
# Automatically generated make config: don't edit
#
CONFIG_ARM=y
# CONFIG_EISA is not set
# CONFIG_SBUS is not set
# CONFIG_MCA is not set
CONFIG_UID16=y
CONFIG_RWSEM_GENERIC_SPINLOCK=y
# CONFIG_RWSEM_XCHGADD_ALGORITHM is not set
# CONFIG_GENERIC_BUST_SPINLOCK is not set
# CONFIG_GENERIC_ISA_DMA is not set

#
# Code maturity level options
#
CONFIG_EXPERIMENTAL=y
# CONFIG_OBSOLETE is not set

#
```

```
# Loadable module support
#
CONFIG_MODULES=y
# CONFIG_MODVERSIONS is not set
CONFIG_KMOD=y

#
# System Type
#
# CONFIG_ARCH_ANAKIN is not set
# CONFIG_ARCH_ARCA5K is not set
# CONFIG_ARCH_CLPS7500 is not set
# CONFIG_ARCH_CLPS711X is not set
# CONFIG_ARCH_CO285 is not set
CONFIG_ARCH_PXA=y
# CONFIG_ARCH_EBSA110 is not set
# CONFIG_ARCH_CAMELOT is not set
# CONFIG_ARCH_FOOTBRIDGE is not set
# CONFIG_ARCH_INTEGRATOR is not set
# CONFIG_ARCH_OMAHA is not set
# CONFIG_ARCH_L7200 is not set
# CONFIG_ARCH_MX1ADS is not set
# CONFIG_ARCH_RPC is not set
# CONFIG_ARCH_RISCSTATION is not set
# CONFIG_ARCH_SA1100 is not set
# CONFIG_ARCH_SHARK is not set
# CONFIG_ARCH_AT91RM9200DK is not set

#
# Archimedes/A5000 Implementations
#

#
# Archimedes/A5000 Implementations (select only ONE)
#
# CONFIG_ARCH_ARC is not set
# CONFIG_ARCH_A5K is not set

#
# Footbridge Implementations
#
# CONFIG_ARCH_CATS is not set
# CONFIG_ARCH_PERSONAL_SERVER is not set
# CONFIG_ARCH_EBSA285_ADDIN is not set
# CONFIG_ARCH_EBSA285_HOST is not set
# CONFIG_ARCH_NETWINDER is not set

#
# SA11x0 Implementations
#
# CONFIG_SA1100_ACCELENT is not set
# CONFIG_SA1100_ASSABET is not set
# CONFIG_ASSABET_NEPONSET is not set
# CONFIG_SA1100_ADSAGC is not set
# CONFIG_SA1100_ADSBITSY is not set
# CONFIG_SA1100_ADSBITSYPLUS is not set
# CONFIG_SA1100_BRUTUS is not set
# CONFIG_SA1100_CEP is not set
# CONFIG_SA1100_CERF is not set
# CONFIG_SA1100_H3100 is not set
# CONFIG_SA1100_H3600 is not set
# CONFIG_SA1100_H3800 is not set
```

```
# CONFIG_SA1100_H3XXX is not set
# CONFIG_H3600_SLEEVE is not set
# CONFIG_SA1100_EXTENEX1 is not set
# CONFIG_SA1100_FLEXANET is not set
# CONFIG_SA1100_FREEBIRD is not set
# CONFIG_SA1100_FRODO is not set
# CONFIG_SA1100_GRAPHICSCLIENT is not set
# CONFIG_SA1100_GRAPHICSMASTER is not set
# CONFIG_SA1100_HACKKIT is not set
# CONFIG_SA1100_BADGE4 is not set
# CONFIG_SA1100_JORNADA720 is not set
# CONFIG_SA1100_HUW_WEBPANEL is not set
# CONFIG_SA1100_ITSY is not set
# CONFIG_SA1100_LART is not set
# CONFIG_SA1100_NANOENGINE is not set
# CONFIG_SA1100_OMNIMETER is not set
# CONFIG_SA1100_PANGOLIN is not set
# CONFIG_SA1100_PLEB is not set
# CONFIG_SA1100_PT_SYSTEM3 is not set
# CONFIG_SA1100_SHANNON is not set
# CONFIG_SA1100_SHERMAN is not set
# CONFIG_SA1100_SIMPAD is not set
# CONFIG_SA1100_SIMPUTER is not set
# CONFIG_SA1100_PFS168 is not set
# CONFIG_SA1100_VICTOR is not set
# CONFIG_SA1100_XP860 is not set
# CONFIG_SA1100_YOPY is not set
# CONFIG_SA1100_USB is not set
# CONFIG_SA1100_USB_NETLINK is not set
# CONFIG_SA1100_USB_CHAR is not set
# CONFIG_SA1100_SSP is not set

#
# Intel PXA250/210 Implementations
#
# CONFIG_ARCH_LUBBOCK is not set
# CONFIG_ARCH_PXA_IDP is not set
# CONFIG_ARCH_PXA_CERF is not set
# CONFIG_ARCH_TRIZEPS2 is not set
CONFIG_ARCH_KOREBOT=y
CONFIG_ARCH_KOREBOT_KTEAM=y
# CONFIG_ARCH_KOREBOT_SBOT is not set
# CONFIG_ARCH_KOREBOT_LE is not set
CONFIG_PXA_USB=y
CONFIG_PXA_USB_NETLINK=y
CONFIG_PXA_USB_CHAR=y

#
# CLPS711X/EP721X Implementations
#
# CONFIG_ARCH_AUTCPU12 is not set
# CONFIG_ARCH_CDB89712 is not set
# CONFIG_ARCH_CLEP7312 is not set
# CONFIG_ARCH_EDB7211 is not set
# CONFIG_ARCH_FORTUNET is not set
# CONFIG_ARCH_GUIDEA07 is not set
# CONFIG_ARCH_P720T is not set
# CONFIG_ARCH_EP7211 is not set
# CONFIG_ARCH_EP7212 is not set
# CONFIG_ARCH_ACORN is not set
# CONFIG_FOOTBRIDGE is not set
# CONFIG_FOOTBRIDGE_HOST is not set
```

```
# CONFIG_FOOTBRIDGE_ADDIN is not set

#
# Processor Type
#
CONFIG_CPU_32=y
# CONFIG_CPU_26 is not set
# CONFIG_CPU_ARM610 is not set
# CONFIG_CPU_ARM710 is not set
# CONFIG_CPU_ARM720T is not set
# CONFIG_CPU_ARM920T is not set
# CONFIG_CPU_ARM922T is not set
# CONFIG_PLD is not set
# CONFIG_CPU_ARM926T is not set
# CONFIG_CPU_ARM1020 is not set
# CONFIG_CPU_ARM1026 is not set
# CONFIG_CPU_SA110 is not set
# CONFIG_CPU_SA1100 is not set
CONFIG_CPU_32v5=y
CONFIG_CPU_XSCALE=y
CONFIG_XSCALE_CACHE_ERRATA=y
# CONFIG_CPU_32v3 is not set
# CONFIG_CPU_32v4 is not set

#
# Processor Features
#
# CONFIG_DISCONTIGMEM is not set

#
# General setup
#
# CONFIG_PCI is not set
# CONFIG_ISA is not set
# CONFIG_ISA_DMA is not set
# CONFIG_ZBOOT_ROM is not set
CONFIG_ZBOOT_ROM_TEXT=0
CONFIG_ZBOOT_ROM_BSS=0
CONFIG_CPU_FREQ=y
CONFIG_HOTPLUG=y

#
# PCMCIA/CardBus support
#
CONFIG_PCMCIA=y
# CONFIG_I82092 is not set
# CONFIG_I82365 is not set
# CONFIG_TCIC is not set
# CONFIG_PCMCIA_CLPS6700 is not set
# CONFIG_PCMCIA_SA1100 is not set
CONFIG_PCMCIA_PXA=y

#
# MMC device drivers
#
CONFIG_MMC=m
CONFIG_MMC_PXA=m
CONFIG_MMC_BLOCK=m
CONFIG_MMC_PARTITIONS=y
CONFIG_NET=y
CONFIG_SYSVIPC=y
# CONFIG_BSD_PROCESS_ACCT is not set
```

```
CONFIG_SYSCTL=y
# CONFIG_XIP_KERNEL is not set

#
# At least one math emulation must be selected
#
CONFIG_FPE_NWFPE=y
# CONFIG_FPE_FASTFPE is not set
CONFIG_KCORE_ELF=y
# CONFIG_KCORE_AOUT is not set
# CONFIG_BINFMT_AOUT is not set
CONFIG_BINFMT_ELF=y
# CONFIG_BINFMT_MISC is not set
# CONFIG_PM is not set
# CONFIG_ARTHUR is not set
CONFIG_CMDLINE="console=ttyS0,115200 mem=64M noinitrd rootfstype=jffs2 \
init=/linuxrc root=/dev/mtdblock1 rw mtdparts=phys:1m(kernel),-(root)"
CONFIG_ALIGNMENT_TRAP=y

#
# Parallel port support
#
# CONFIG_PARPORT is not set

#
# Memory Technology Devices (MTD)
#
CONFIG_MTD=y
# CONFIG_MTD_DEBUG is not set
CONFIG_MTD_PARTITIONS=y
# CONFIG_MTD_CONCAT is not set
CONFIG_MTD_REDBOOT_PARTS=y
CONFIG_MTD_CMDLINE_PARTS=y
# CONFIG_MTD_AFS_PARTS is not set

#
# User Modules And Translation Layers
#
CONFIG_MTD_CHAR=y
CONFIG_MTD_BLOCK=y
# CONFIG_FTL is not set
# CONFIG_NFTL is not set

#
# RAM/ROM/Flash chip drivers
#
CONFIG_MTD_CFI=y
# CONFIG_MTD_JEDECPROBE is not set
CONFIG_MTD_GEN_PROBE=y
# CONFIG_MTD_CFI_ADV_OPTIONS is not set
CONFIG_MTD_CFI_INTELEXT=y
# CONFIG_MTD_CFI_AMDSTD is not set
# CONFIG_MTD_RAM is not set
# CONFIG_MTD_ROM is not set
# CONFIG_MTD_ABSENT is not set
# CONFIG_MTD_OBSOLETE_CHIPS is not set
# CONFIG_MTD_AMDSTD is not set
# CONFIG_MTD_SHARP is not set
# CONFIG_MTD_JEDEC is not set

#
# Mapping drivers for chip access
```

```
#
CONFIG_MTD_PHYSMAP=y
CONFIG_MTD_PHYSMAP_START=0
CONFIG_MTD_PHYSMAP_LEN=2000000
CONFIG_MTD_PHYSMAP_BUSWIDTH=4
# CONFIG_MTD_LUBBOCK is not set
# CONFIG_MTD_NORA is not set
# CONFIG_MTD_ARM_INTEGRATOR is not set
# CONFIG_MTD_CDB89712 is not set
# CONFIG_MTD_SA1100 is not set
# CONFIG_MTD_DC21285 is not set
# CONFIG_MTD_IQ80310 is not set
# CONFIG_MTD_FORTUNET is not set
# CONFIG_MTD_PXA_CERF is not set
# CONFIG_MTD_EPXA is not set
# CONFIG_MTD_AUTCPU12 is not set
# CONFIG_MTD_EDB7312 is not set
# CONFIG_MTD_IMPA7 is not set
# CONFIG_MTD_TRIZEPS2 is not set
# CONFIG_MTD_PCI is not set

#
# Self-contained MTD device drivers
#
# CONFIG_MTD_PMC551 is not set
# CONFIG_MTD_SLRAM is not set
# CONFIG_MTD_MTDRAM is not set
CONFIG_MTD_BLKMTD=m

#
# Disk-On-Chip Device Drivers
#
# CONFIG_MTD_DOC1000 is not set
# CONFIG_MTD_DOC2000 is not set
# CONFIG_MTD_DOC2001 is not set
# CONFIG_MTD_DOCPROBE is not set

#
# NAND Flash Device Drivers
#
# CONFIG_MTD_NAND is not set

#
# Plug and Play configuration
#
# CONFIG_PNP is not set
# CONFIG_ISAPNP is not set

#
# Block devices
#
# CONFIG_BLK_DEV_FD is not set
# CONFIG_BLK_DEV_XD is not set
# CONFIG_PARIDE is not set
# CONFIG_BLK_CPQ_DA is not set
# CONFIG_BLK_CPQ_CISS_DA is not set
# CONFIG_CISS_SCSI_TAPE is not set
# CONFIG_BLK_DEV_DAC960 is not set
# CONFIG_BLK_DEV_UMEM is not set
# CONFIG_BLK_DEV_LOOP is not set
# CONFIG_BLK_DEV_NBD is not set
CONFIG_BLK_DEV_RAM=y
```

```
CONFIG_BLK_DEV_RAM_SIZE=4096
CONFIG_BLK_DEV_INITRD=y

#
# Multi-device support (RAID and LVM)
#
# CONFIG_MD is not set
# CONFIG_BLK_DEV_MD is not set
# CONFIG_MD_LINEAR is not set
# CONFIG_MD_RAID0 is not set
# CONFIG_MD_RAID1 is not set
# CONFIG_MD_RAID5 is not set
# CONFIG_MD_MULTIPATH is not set
# CONFIG_BLK_DEV_LVM is not set

#
# Networking options
#
CONFIG_PACKET=m
CONFIG_PACKET_MMAP=y
# CONFIG_NETLINK_DEV is not set
# CONFIG_NETFILTER is not set
# CONFIG_FILTER is not set
CONFIG_UNIX=y
CONFIG_INET=y
# CONFIG_IP_MULTICAST is not set
# CONFIG_IP_ADVANCED_ROUTER is not set
CONFIG_IP_PNP=y
# CONFIG_IP_PNP_DHCP is not set
CONFIG_IP_PNP_BOOTP=y
# CONFIG_IP_PNP_RARP is not set
# CONFIG_NET_IPIP is not set
# CONFIG_NET_IPGRE is not set
# CONFIG_ARPD is not set
# CONFIG_INET_ECN is not set
# CONFIG_SYN_COOKIES is not set
# CONFIG_IPV6 is not set
# CONFIG_KHTTPD is not set
# CONFIG_ATM is not set
# CONFIG_VLAN_8021Q is not set

#
#
#
# CONFIG_IPX is not set
# CONFIG_ATALK is not set

#
# Appletalk devices
#
# CONFIG_DEV_APPLETALK is not set
# CONFIG_DECNET is not set
CONFIG_BRIDGE=m
# CONFIG_X25 is not set
# CONFIG_LAPB is not set
# CONFIG_LLC is not set
# CONFIG_NET_DIVERT is not set
# CONFIG_ECONET is not set
# CONFIG_WAN_ROUTER is not set
# CONFIG_NET_FASTROUTE is not set
# CONFIG_NET_HW_FLOWCONTROL is not set
```

```
#
# QoS and/or fair queueing
#
# CONFIG_NET_SCHED is not set

#
# Network testing
#
# CONFIG_NET_PKTGEN is not set

#
# Network device support
#
CONFIG_NETDEVICES=y

#
# ARCnet devices
#
# CONFIG_ARCNET is not set
# CONFIG_DUMMY is not set
# CONFIG_BONDING is not set
# CONFIG_EQUALIZER is not set
# CONFIG_TUN is not set
# CONFIG_ETHERTAP is not set

#
# Ethernet (10 or 100Mbit)
#
CONFIG_NET_ETHERNET=y
# CONFIG_ARM_AM79C961A is not set
# CONFIG_ARM_CIRRUS is not set
# CONFIG_SUNLANCE is not set
# CONFIG_SUNBMAC is not set
# CONFIG_SUNQE is not set
# CONFIG_SUNGEM is not set
# CONFIG_NET_VENDOR_3COM is not set
# CONFIG_LANCE is not set
# CONFIG_NET_VENDOR_SMC is not set
# CONFIG_NET_VENDOR_RACAL is not set
# CONFIG_NET_ISA is not set
# CONFIG_NET_PCI is not set
# CONFIG_NET_POCKET is not set

#
# Ethernet (1000 Mbit)
#
# CONFIG_ACENIC is not set
# CONFIG_DL2K is not set
# CONFIG_MYRI_SBUS is not set
# CONFIG_NS83820 is not set
# CONFIG_HAMACHI is not set
# CONFIG_YELLOWFIN is not set
# CONFIG_SK98LIN is not set
# CONFIG_TIGON3 is not set
# CONFIG_FDDI is not set
# CONFIG_HIPPI is not set
# CONFIG_PLIP is not set
CONFIG_PPP=m
# CONFIG_PPP_MULTILINK is not set
# CONFIG_PPP_FILTER is not set
CONFIG_PPP_ASYNC=m
CONFIG_PPP_SYNC_TTY=m
```

```
CONFIG_PPP_DEFLATE=m
CONFIG_PPP_BSDCOMP=m
# CONFIG_PPPOE is not set
CONFIG_SLIP=m
CONFIG_SLIP_COMPRESSED=y
CONFIG_SLIP_SMART=y
CONFIG_SLIP_MODE_SLIP6=y

#
# Wireless LAN (non-hamradio)
#
CONFIG_NET_RADIO=y
# CONFIG_STRIP is not set
# CONFIG_WAVELAN is not set
# CONFIG_ARLAN is not set
# CONFIG_AIRONET4500 is not set
# CONFIG_AIRONET4500_NONCS is not set
# CONFIG_AIRONET4500_PROC is not set
CONFIG_HERMES=m
CONFIG_HOSTAP=m
CONFIG_SPECTRUM_CS=m

#
# Wireless Pcmcia cards support
#
CONFIG_PCMCIA_HERMES=m
CONFIG_HOSTAP_CS=m
# CONFIG_AIRO_CS is not set
CONFIG_NET_WIRELESS=y

#
# Token Ring devices
#
# CONFIG_TR is not set
# CONFIG_NET_FC is not set
# CONFIG_RCPCI is not set
# CONFIG_SHAPER is not set

#
# Wan interfaces
#
# CONFIG_WAN is not set

#
# PCMCIA network device support
#
# CONFIG_NET_PCMCIA is not set

#
# Amateur Radio support
#
# CONFIG_HAMRADIO is not set

#
# IrDA (infrared) support
#
CONFIG_IRDA=m

#
# IrDA protocols
#
CONFIG_IRLAN=m
```

```
CONFIG_IRNET=m
CONFIG_IRCOMM=m
CONFIG_IRDA_ULTRA=y

#
# IrDA options
#
CONFIG_IRDA_CACHE_LAST_LSAP=y
CONFIG_IRDA_FAST_RR=y
# CONFIG_IRDA_DEBUG is not set

#
# Infrared-port device drivers
#

#
# SIR device drivers
#
CONFIG_IRTTY_SIR=m
CONFIG_IRPORT_SIR=m

#
# Dongle support
#
# CONFIG_DONGLE is not set

#
# FIR device drivers
#
# CONFIG_USB_IRDA is not set
# CONFIG_NSC_FIR is not set
# CONFIG_WINBOND_FIR is not set
# CONFIG_TOSHIBA_FIR is not set
# CONFIG_SMC_IRCC_FIR is not set
# CONFIG_ALI_FIR is not set
# CONFIG_VLSI_FIR is not set
CONFIG_PXA_FIR=m

#
# ATA/ATAPI/MFM/RLL support
#
CONFIG_IDE=y

#
# IDE, ATA and ATAPI Block devices
#
CONFIG_BLK_DEV_IDE=m

#
# Please see Documentation/ide.txt for help/info on IDE drives
#
# CONFIG_BLK_DEV_HD_IDE is not set
# CONFIG_BLK_DEV_HD is not set
CONFIG_BLK_DEV_IDEDISK=m
# CONFIG_IDEDISK_MULTI_MODE is not set
# CONFIG_IDEDISK_STROKE is not set
# CONFIG_BLK_DEV_IDEDISK_VENDOR is not set
# CONFIG_BLK_DEV_IDEDISK_FUJITSU is not set
# CONFIG_BLK_DEV_IDEDISK_IBM is not set
# CONFIG_BLK_DEV_IDEDISK_MAXTOR is not set
# CONFIG_BLK_DEV_IDEDISK_QUANTUM is not set
# CONFIG_BLK_DEV_IDEDISK_SEAGATE is not set
```

```
# CONFIG_BLK_DEV_IDEDISK_WD is not set
# CONFIG_BLK_DEV_COMMERIAL is not set
# CONFIG_BLK_DEV_TIVO is not set
CONFIG_BLK_DEV_IDECS=m
# CONFIG_BLK_DEV_IDECD is not set
# CONFIG_BLK_DEV_IDETAPE is not set
# CONFIG_BLK_DEV_IDEFLOPPY is not set
# CONFIG_BLK_DEV_IDESCSI is not set
# CONFIG_IDE_TASK_IOCTL is not set

#
# IDE chipset support/bugfixes
#
# CONFIG_BLK_DEV_CMD640 is not set
# CONFIG_BLK_DEV_CMD640_ENHANCED is not set
# CONFIG_BLK_DEV_ISAPNP is not set
# CONFIG_IDE_CHIPSETS is not set
# CONFIG_IDEDMA_AUTO is not set
# CONFIG_DMA_NONPCI is not set
# CONFIG_BLK_DEV_IDE_MODES is not set
# CONFIG_BLK_DEV_ATARAID is not set
# CONFIG_BLK_DEV_ATARAID_PDC is not set
# CONFIG_BLK_DEV_ATARAID_HPT is not set

#
# SCSI support
#
CONFIG_SCSI=m

#
# SCSI support type (disk, tape, CD-ROM)
#
CONFIG_BLK_DEV_SD=m
CONFIG_SD_EXTRA_DEVS=40
# CONFIG_CHR_DEV_ST is not set
# CONFIG_CHR_DEV_OSST is not set
# CONFIG_BLK_DEV_SR is not set
CONFIG_CHR_DEV_SG=m

#
# Some SCSI devices (e.g. CD jukebox) support multiple LUNs
#
CONFIG_SCSI_DEBUG_QUEUES=y
CONFIG_SCSI_MULTI_LUN=y
CONFIG_SCSI_CONSTANTS=y
CONFIG_SCSI_LOGGING=y

#
# SCSI low-level drivers
#
# CONFIG_SCSI_7000FASST is not set
# CONFIG_SCSI_ACARD is not set
# CONFIG_SCSI_AHA152X is not set
# CONFIG_SCSI_AHA1542 is not set
# CONFIG_SCSI_AHA1740 is not set
# CONFIG_SCSI_AACRAID is not set
# CONFIG_SCSI_AIC7XXX is not set
# CONFIG_SCSI_AIC7XXX_OLD is not set
# CONFIG_SCSI_DPT_I2O is not set
# CONFIG_SCSI_ADVANSYS is not set
# CONFIG_SCSI_IN2000 is not set
# CONFIG_SCSI_AM53C974 is not set
```

```
# CONFIG_SCSI_MEGARAID is not set
# CONFIG_SCSI_BUSLOGIC is not set
# CONFIG_SCSI_DMX3191D is not set
# CONFIG_SCSI_DTC3280 is not set
# CONFIG_SCSI_EATA is not set
# CONFIG_SCSI_EATA_DMA is not set
# CONFIG_SCSI_EATA_PIO is not set
# CONFIG_SCSI_FUTURE_DOMAIN is not set
# CONFIG_SCSI_GDTH is not set
# CONFIG_SCSI_GENERIC_NCR5380 is not set
# CONFIG_SCSI_INITIO is not set
# CONFIG_SCSI_INIA100 is not set
# CONFIG_SCSI_NCR53C406A is not set
# CONFIG_SCSI_NCR53C7xx is not set
# CONFIG_SCSI_PAS16 is not set
# CONFIG_SCSI_PCI2000 is not set
# CONFIG_SCSI_PCI2220I is not set
# CONFIG_SCSI_PSI240I is not set
# CONFIG_SCSI_QLOGIC_FAS is not set
# CONFIG_SCSI_SIM710 is not set
# CONFIG_SCSI_SYM53C416 is not set
# CONFIG_SCSI_T128 is not set
# CONFIG_SCSI_U14_34F is not set
# CONFIG_SCSI_DEBUG is not set

#
# PCMCIA SCSI adapter support
#
# CONFIG_SCSI_PCMCIA is not set

#
# Synchronous Serial Interface
#
CONFIG_SSI=m

#
# SSI Bus Drivers
#
CONFIG_SSI_PXA=m
# CONFIG_SSI_KNET is not set

#
# SSI Device Drivers
#
# CONFIG_SSI_JUNO is not set

#
# I2O device support
#
# CONFIG_I2O is not set
# CONFIG_I2O_BLOCK is not set
# CONFIG_I2O_LAN is not set
# CONFIG_I2O_SCSI is not set
# CONFIG_I2O_PROC is not set

#
# ISDN subsystem
#
# CONFIG_ISDN is not set

#
# Input core support
```

```
#
CONFIG_INPUT=y
CONFIG_INPUT_KEYBDEV=m
CONFIG_INPUT_MOUSEDEV=m
CONFIG_INPUT_MOUSEDEV_SCREEN_X=1024
CONFIG_INPUT_MOUSEDEV_SCREEN_Y=768
CONFIG_INPUT_JOYDEV=m
CONFIG_INPUT_EVDEV=y
CONFIG_INPUT_UINPUT=m

#
# Character devices
#
CONFIG_VT=y
CONFIG_VT_CONSOLE=y
CONFIG_SERIAL=y
CONFIG_SERIAL_CONSOLE=y
# CONFIG_SERIAL_EXTENDED is not set
# CONFIG_SERIAL_NONSTANDARD is not set

#
# Serial drivers
#
# CONFIG_SERIAL_ANAKIN is not set
# CONFIG_SERIAL_ANAKIN_CONSOLE is not set
# CONFIG_SERIAL_AMBA is not set
# CONFIG_SERIAL_AMBA_CONSOLE is not set
# CONFIG_SERIAL_CLPS711X is not set
# CONFIG_SERIAL_CLPS711X_CONSOLE is not set
# CONFIG_SERIAL_21285 is not set
# CONFIG_SERIAL_21285_OLD is not set
# CONFIG_SERIAL_21285_CONSOLE is not set
# CONFIG_SERIAL_UART00 is not set
# CONFIG_SERIAL_UART00_CONSOLE is not set
# CONFIG_SERIAL_SA1100 is not set
# CONFIG_SERIAL_SA1100_CONSOLE is not set
# CONFIG_SERIAL_OMAHA is not set
# CONFIG_SERIAL_OMAHA_CONSOLE is not set
# CONFIG_SERIAL_AT91US3 is not set
# CONFIG_SERIAL_AT91US3_CONSOLE is not set
# CONFIG_SERIAL_8250 is not set
# CONFIG_SERIAL_8250_CONSOLE is not set
# CONFIG_SERIAL_8250_EXTENDED is not set
# CONFIG_SERIAL_8250_MANY_PORTS is not set
# CONFIG_SERIAL_8250_SHARE_IRQ is not set
# CONFIG_SERIAL_8250_DETECT_IRQ is not set
# CONFIG_SERIAL_8250_MULTIPORT is not set
# CONFIG_SERIAL_8250_HUB6 is not set
CONFIG_UNIX98_PTYS=y
CONFIG_UNIX98_PTY_COUNT=256

#
# I2C support
#
CONFIG_I2C=y
CONFIG_I2C_SLOW=y
# CONFIG_I2C_FAST is not set
# CONFIG_I2C_ALGOBIT is not set
# CONFIG_I2C_ALGOPCF is not set
CONFIG_I2C_PXA_ALGO=y
CONFIG_I2C_PXA_ADAP=y
CONFIG_I2C_CHARDEV=y
```

```
CONFIG_I2C_PROC=y
# CONFIG_I2C_DS1307 is not set

#
# L3 serial bus support
#
# CONFIG_L3 is not set
# CONFIG_L3_ALGOBIT is not set
# CONFIG_L3_BIT_SA1100_GPIO is not set

#
# Other L3 adapters
#
# CONFIG_L3_SA1111 is not set
# CONFIG_BIT_SA1100_GPIO is not set

#
# Mice
#
# CONFIG_BUSMOUSE is not set
# CONFIG_MOUSE is not set

#
# Joysticks
#
# CONFIG_INPUT_GAMEPORT is not set
# CONFIG_INPUT_NS558 is not set
# CONFIG_INPUT_LIGHTNING is not set
# CONFIG_INPUT_PCIGAME is not set
# CONFIG_INPUT_CS461X is not set
# CONFIG_INPUT_EMU10K1 is not set
CONFIG_INPUT_SERIO=m
CONFIG_INPUT_SERPORT=m

#
# Joysticks
#
# CONFIG_INPUT_ANALOG is not set
# CONFIG_INPUT_A3D is not set
# CONFIG_INPUT_ADI is not set
# CONFIG_INPUT_COBRA is not set
# CONFIG_INPUT_GF2K is not set
# CONFIG_INPUT_GRIP is not set
# CONFIG_INPUT_INTERACT is not set
# CONFIG_INPUT_TMDC is not set
# CONFIG_INPUT_SIDEWINDER is not set
CONFIG_INPUT_IFORCE_USB=m
CONFIG_INPUT_IFORCE_232=m
CONFIG_INPUT_WARRIOR=m
CONFIG_INPUT_MAGELLAN=m
CONFIG_INPUT_SPACEORB=m
CONFIG_INPUT_SPACEBALL=m
CONFIG_INPUT_STINGER=m
# CONFIG_INPUT_DB9 is not set
# CONFIG_INPUT_GAMECON is not set
# CONFIG_INPUT_TURBOGRAFX is not set
# CONFIG_QIC02_TAPE is not set

#
# Watchdog Cards
#
# CONFIG_WATCHDOG is not set
```

```
# CONFIG_NVRAM is not set
# CONFIG_RTC is not set
CONFIG_PXA_RTC=y
# CONFIG_DTLK is not set
# CONFIG_R3964 is not set
# CONFIG_APPLICOM is not set

#
# Ftape, the floppy tape device driver
#
# CONFIG_FTAPE is not set
# CONFIG_AGP is not set
# CONFIG_DRM is not set

#
# PCMCIA character devices
#
CONFIG_PCMCIA_SERIAL_CS=m

#
# Multimedia devices
#
CONFIG_VIDEO_DEV=m

#
# Video For Linux
#
CONFIG_VIDEO_PROC_FS=y
# CONFIG_I2C_PARPORT is not set

#
# Video Adapters
#
# CONFIG_VIDEO_PMS is not set
# CONFIG_VIDEO_CPIA is not set
# CONFIG_VIDEO_SAA5249 is not set
# CONFIG_TUNER_3036 is not set
# CONFIG_VIDEO_STRADIS is not set
# CONFIG_VIDEO_ZORAN is not set
# CONFIG_VIDEO_ZORAN_BUZ is not set
# CONFIG_VIDEO_ZORAN_DC10 is not set
# CONFIG_VIDEO_ZORAN_LML33 is not set
# CONFIG_VIDEO_ZR36120 is not set
# CONFIG_VIDEO_MEYE is not set
# CONFIG_VIDEO_CYBERPRO is not set

#
# Radio Adapters
#
# CONFIG_RADIO_CADET is not set
# CONFIG_RADIO_RTRACK is not set
# CONFIG_RADIO_RTRACK2 is not set
# CONFIG_RADIO_AZTECH is not set
# CONFIG_RADIO_GEMTEK is not set
# CONFIG_RADIO_GEMTEK_PCI is not set
# CONFIG_RADIO_MAXIRADIO is not set
# CONFIG_RADIO_MAESTRO is not set
# CONFIG_RADIO_MIROPCM20 is not set
# CONFIG_RADIO_MIROPCM20_RDS is not set
# CONFIG_RADIO_SF16FMI is not set
# CONFIG_RADIO_TERRATEC is not set
# CONFIG_RADIO_TRUST is not set
```

```
# CONFIG_RADIO_TYPHOON is not set
# CONFIG_RADIO_ZOLTRIX is not set

#
# File systems
#
# CONFIG_QUOTA is not set
# CONFIG_AUTOFS_FS is not set
# CONFIG_AUTOFS4_FS is not set
# CONFIG_REISERFS_FS is not set
# CONFIG_REISERFS_CHECK is not set
# CONFIG_REISERFS_PROC_INFO is not set
# CONFIG_ADFS_FS is not set
# CONFIG_ADFS_FS_RW is not set
# CONFIG_AFFS_FS is not set
# CONFIG_HFS_FS is not set
# CONFIG_BFS_FS is not set
# CONFIG_EXT3_FS is not set
# CONFIG_JBD is not set
# CONFIG_JBD_DEBUG is not set
CONFIG_FAT_FS=m
CONFIG_MSDOS_FS=m
# CONFIG_UMSDOS_FS is not set
CONFIG_VFAT_FS=m
# CONFIG_EFS_FS is not set
# CONFIG_JFFS_FS is not set
CONFIG_JFFS2_FS=y
CONFIG_JFFS2_FS_DEBUG=0
CONFIG_CRAMFS=y
CONFIG_TMPFS=y
CONFIG_RAMFS=y
# CONFIG_ISO9660_FS is not set
# CONFIG_JOLIET is not set
# CONFIG_ZISOFS is not set
# CONFIG_MINIX_FS is not set
# CONFIG_VXFS_FS is not set
# CONFIG_NTFS_FS is not set
# CONFIG_NTFS_RW is not set
# CONFIG_HPFS_FS is not set
CONFIG_PROC_FS=y
CONFIG_DEVFS_FS=y
CONFIG_DEVFS_MOUNT=y
# CONFIG_DEVFS_DEBUG is not set
CONFIG_DEVPTS_FS=y
# CONFIG_QNX4FS_FS is not set
# CONFIG_QNX4FS_RW is not set
CONFIG_ROMFS_FS=y
CONFIG_EXT2_FS=y
# CONFIG_SYSV_FS is not set
# CONFIG_UDF_FS is not set
# CONFIG_UDF_RW is not set
# CONFIG_UFS_FS is not set
# CONFIG_UFS_FS_WRITE is not set

#
# Network File Systems
#
# CONFIG_CODA_FS is not set
# CONFIG_INTERMEZZO_FS is not set
CONFIG_NFS_FS=y
CONFIG_NFS_V3=y
CONFIG_ROOT_NFS=y
```

```
# CONFIG_NFSD is not set
# CONFIG_NFSD_V3 is not set
CONFIG_SUNRPC=y
CONFIG_LOCKD=y
CONFIG_LOCKD_V4=y
# CONFIG_SMB_FS is not set
# CONFIG_NCP_FS is not set
# CONFIG_NCPFS_PACKET_SIGNING is not set
# CONFIG_NCPFS_IOCTL_LOCKING is not set
# CONFIG_NCPFS_STRONG is not set
# CONFIG_NCPFS_NFS_NS is not set
# CONFIG_NCPFS_OS2_NS is not set
# CONFIG_NCPFS_SMALLDOS is not set
# CONFIG_NCPFS_NLS is not set
# CONFIG_NCPFS_EXTRAS is not set
# CONFIG_ZISOFS_FS is not set
CONFIG_ZLIB_FS_INFLATE=y

#
# Partition Types
#
# CONFIG_PARTITION_ADVANCED is not set
CONFIG_MSDOS_PARTITION=y
# CONFIG_SMB_NLS is not set
CONFIG_NLS=y

#
# Native Language Support
#
CONFIG_NLS_DEFAULT="iso8859-1"
# CONFIG_NLS_CODEPAGE_437 is not set
# CONFIG_NLS_CODEPAGE_737 is not set
# CONFIG_NLS_CODEPAGE_775 is not set
# CONFIG_NLS_CODEPAGE_850 is not set
# CONFIG_NLS_CODEPAGE_852 is not set
# CONFIG_NLS_CODEPAGE_855 is not set
# CONFIG_NLS_CODEPAGE_857 is not set
# CONFIG_NLS_CODEPAGE_860 is not set
# CONFIG_NLS_CODEPAGE_861 is not set
# CONFIG_NLS_CODEPAGE_862 is not set
# CONFIG_NLS_CODEPAGE_863 is not set
# CONFIG_NLS_CODEPAGE_864 is not set
# CONFIG_NLS_CODEPAGE_865 is not set
# CONFIG_NLS_CODEPAGE_866 is not set
# CONFIG_NLS_CODEPAGE_869 is not set
# CONFIG_NLS_CODEPAGE_936 is not set
# CONFIG_NLS_CODEPAGE_950 is not set
# CONFIG_NLS_CODEPAGE_932 is not set
# CONFIG_NLS_CODEPAGE_949 is not set
# CONFIG_NLS_CODEPAGE_874 is not set
# CONFIG_NLS_ISO8859_8 is not set
# CONFIG_NLS_CODEPAGE_1250 is not set
# CONFIG_NLS_CODEPAGE_1251 is not set
# CONFIG_NLS_ISO8859_1 is not set
# CONFIG_NLS_ISO8859_2 is not set
# CONFIG_NLS_ISO8859_3 is not set
# CONFIG_NLS_ISO8859_4 is not set
# CONFIG_NLS_ISO8859_5 is not set
# CONFIG_NLS_ISO8859_6 is not set
# CONFIG_NLS_ISO8859_7 is not set
# CONFIG_NLS_ISO8859_9 is not set
# CONFIG_NLS_ISO8859_13 is not set
```

```
# CONFIG_NLS_ISO8859_14 is not set
# CONFIG_NLS_ISO8859_15 is not set
# CONFIG_NLS_KOI8_R is not set
# CONFIG_NLS_KOI8_U is not set
# CONFIG_NLS_UTF8 is not set

#
# Console drivers
#
CONFIG_PC_KEYMAP=y
# CONFIG_VGA_CONSOLE is not set

#
# Frame-buffer support
#
# CONFIG_FB is not set

#
# Sound
#
CONFIG_SOUND=m
# CONFIG_SOUND_BT878 is not set
# CONFIG_SOUND_CMPCI is not set
# CONFIG_SOUND_EMU10K1 is not set
# CONFIG_MIDI_EMU10K1 is not set
# CONFIG_SOUND_FUSION is not set
# CONFIG_SOUND_CS4281 is not set
# CONFIG_SOUND_ES1370 is not set
# CONFIG_SOUND_ES1371 is not set
# CONFIG_SOUND_ESSSOLO1 is not set
# CONFIG_SOUND_MAESTRO is not set
# CONFIG_SOUND_MAESTRO3 is not set
# CONFIG_SOUND_ICH is not set
# CONFIG_SOUND_RME96XX is not set
# CONFIG_SOUND_SONICVIBES is not set
# CONFIG_SOUND_TRIDENT is not set
# CONFIG_SOUND_MSNDCLAS is not set
# CONFIG_SOUND_MSNDPIN is not set
# CONFIG_SOUND_VIA82CXXX is not set
# CONFIG_MIDI_VIA82CXXX is not set
CONFIG_SOUND_PXA_UDA1342=m
# CONFIG_SOUND_PXA_UDA1342_DIFF is not set
# CONFIG_SOUND_PXA_UDA1342_DIFF is not set
# CONFIG_SOUND_OSS is not set
# CONFIG_SOUND_VIDC is not set
# CONFIG_SOUND_WAVEARTIST is not set
CONFIG_SOUND_PXA_AC97=m
# CONFIG_SOUND_TVMIXER is not set

#
# Multimedia Capabilities Port drivers
#
# CONFIG_MCP is not set
# CONFIG_MCP_SA1100 is not set
# CONFIG_MCP_UCB1200 is not set
# CONFIG_MCP_UCB1200_AUDIO is not set
# CONFIG_MCP_UCB1200_TS is not set
# CONFIG_MCP_UCB1400_TS is not set

#
# USB support
#
```

```
CONFIG_USB=y
# CONFIG_USB_DEBUG is not set

#
# Miscellaneous USB options
#
CONFIG_USB_DEVICEFS=y
# CONFIG_USB_BANDWIDTH is not set
# CONFIG_USB_LONG_TIMEOUT is not set

#
# USB Host Controller Drivers
#
# CONFIG_USB_EHCI_HCD is not set
# CONFIG_USB_UHCI is not set
# CONFIG_USB_UHCI_ALT is not set
# CONFIG_USB_OHCI is not set
# CONFIG_USB_OHCI_SA1111 is not set
CONFIG_USB_OTG=m

#
# USB Device Class drivers
#
# CONFIG_USB_AUDIO is not set
# CONFIG_USB_EMI26 is not set

#
#   USB Bluetooth can only be used with disabled Bluetooth subsystem
#
CONFIG_USB_STORAGE=m
CONFIG_USB_STORAGE_DEBUG=y
# CONFIG_USB_STORAGE_DATAFAB is not set
# CONFIG_USB_STORAGE_FREECOM is not set
# CONFIG_USB_STORAGE_ISD200 is not set
# CONFIG_USB_STORAGE_DPCM is not set
# CONFIG_USB_STORAGE_HP8200e is not set
# CONFIG_USB_STORAGE_SDDR09 is not set
# CONFIG_USB_STORAGE_JUMPSHOT is not set
# CONFIG_USB_ACM is not set
# CONFIG_USB_PRINTER is not set

#
# USB Human Interface Devices (HID)
#
CONFIG_USB_HID=y
CONFIG_USB_HIDINPUT=y
CONFIG_USB_HIDDEV=y
# CONFIG_USB_WACOM is not set

#
# USB Imaging devices
#
# CONFIG_USB_DC2XX is not set
# CONFIG_USB_MDC800 is not set
# CONFIG_USB_SCANNER is not set
# CONFIG_USB_MICROTEK is not set
# CONFIG_USB_HPUSBSCSI is not set

#
# USB Multimedia devices
#
# CONFIG_USB_IBMCAM is not set
```

```
CONFIG_USB_OV511=m
# CONFIG_USB_PWC is not set
# CONFIG_USB_SE401 is not set
# CONFIG_USB_STV680 is not set
# CONFIG_USB_VICAM is not set
# CONFIG_USB_DSBR is not set
# CONFIG_USB_DABUSB is not set

#
# USB Network adaptors
#
# CONFIG_USB_PEGASUS is not set
# CONFIG_USB_RTL8150 is not set
# CONFIG_USB_KAWETH is not set
# CONFIG_USB_CATC is not set
# CONFIG_USB_CDCETHER is not set
CONFIG_USB_USBNET=m

#
# USB port drivers
#
# CONFIG_USB_USS720 is not set

#
# USB Serial Converter support
#
# CONFIG_USB_SERIAL is not set
# CONFIG_USB_SERIAL_GENERIC is not set
# CONFIG_USB_SERIAL_BELKIN is not set
# CONFIG_USB_SERIAL_WHITEHEAT is not set
# CONFIG_USB_SERIAL_DIGI_ACCELEPORT is not set
# CONFIG_USB_SERIAL_EMPEG is not set
# CONFIG_USB_SERIAL_FTDI_SIO is not set
# CONFIG_USB_SERIAL_VISOR is not set
# CONFIG_USB_SERIAL_IPAQ is not set
# CONFIG_USB_SERIAL_IR is not set
# CONFIG_USB_SERIAL_EDGEPORT is not set
# CONFIG_USB_SERIAL_KEYSPAN_PDA is not set
# CONFIG_USB_SERIAL_KEYSPAN is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA28 is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA28X is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA28XA is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA28XB is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA19 is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA18X is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA19W is not set
# CONFIG_USB_SERIAL_KEYSPAN_USA49W is not set
# CONFIG_USB_SERIAL_MCT_U232 is not set
# CONFIG_USB_SERIAL_KLSI is not set
# CONFIG_USB_SERIAL_PL2303 is not set
# CONFIG_USB_SERIAL_CYBERJACK is not set
# CONFIG_USB_SERIAL_XIRCOM is not set
# CONFIG_USB_SERIAL_OMNINET is not set

#
# USB Miscellaneous drivers
#
# CONFIG_USB_RIO500 is not set
# CONFIG_USB_AUERSWALD is not set
# CONFIG_USB_BRLVGER is not set

#
```

```
# Bluetooth support
#
CONFIG_BLUEZ=m
CONFIG_BLUEZ_L2CAP=m
CONFIG_BLUEZ_SCO=m
CONFIG_BLUEZ_RFCOMM=m
CONFIG_BLUEZ_RFCOMM_TTY=y
CONFIG_BLUEZ_BNEP=m
CONFIG_BLUEZ_BNEP_MC_FILTER=y
CONFIG_BLUEZ_BNEP_PROTO_FILTER=y
CONFIG_BLUEZ_HIDP=m

#
# Bluetooth device drivers
#
CONFIG_BLUEZ_HCIUSB=m
# CONFIG_BLUEZ_HCIUSB_SCO is not set
CONFIG_BLUEZ_HCIUART=m
CONFIG_BLUEZ_HCIUART_H4=y
CONFIG_BLUEZ_HCIUART_BCSP=y
CONFIG_BLUEZ_HCIUART_BCSP_TXCRC=y
CONFIG_BLUEZ_HCIBFUSB=m
CONFIG_BLUEZ_HCIDTL1=m
CONFIG_BLUEZ_HCIBT3C=m
CONFIG_BLUEZ_HCIBLUECARD=m
CONFIG_BLUEZ_HCIBTUART=m
CONFIG_BLUEZ_HCIVHCI=m

#
# Kernel hacking
#
CONFIG_FRAME_POINTER=y
CONFIG_DEBUG_USER=y
# CONFIG_DEBUG_INFO is not set
# CONFIG_NO_PGT_CACHE is not set
# CONFIG_DEBUG_KERNEL is not set
# CONFIG_DEBUG_SLAB is not set
# CONFIG_MAGIC_SYSRQ is not set
# CONFIG_DEBUG_SPINLOCK is not set
# CONFIG_DEBUG_WAITQ is not set
# CONFIG_DEBUG_BUGVERBOSE is not set
# CONFIG_DEBUG_ERRORS is not set
# CONFIG_DEBUG_LL is not set
# CONFIG_DEBUG_DC21285_PORT is not set
# CONFIG_DEBUG_CLPS711X_UART2 is not set
```

Appendix D

Bluetooth

D.1 Diary Excerpt PCMCIA Bluetooth Card

```
17/05/06
--------
--------

Discoved that using make xconfig created a fresh config file and that \
pcmcia serial_cs module could not be made. Found that copying \
config-korebot to .config and then adding bluetooth stuff was enough \
and did compile on make dep make bzImage make modules make modules_instal

diff was

diff -c /home/krobinson/sw/linux-2.4.19-kb11/.config \
/home/krobinson/sw/linux-2.4.19-kb11/config-korebot
*** /home/krobinson/sw/linux-2.4.19-kb11/.config        Wed May 17 \
11:54:09 2006
--- /home/krobinson/sw/linux-2.4.19-kb11/config-korebot Wed May 17 \
11:54:09 2006
***************
*** 1063,1096 ****
  # CONFIG_USB_AUERSWALD is not set
  # CONFIG_USB_BRLVGER is not set

-
  #
  # Bluetooth support
  #
  CONFIG_BLUEZ=m
  CONFIG_BLUEZ_L2CAP=m
  CONFIG_BLUEZ_SCO=m
- CONFIG_BLUEZ_RFCOMM=m
- CONFIG_BLUEZ_RFCOMM_TTY=y
- CONFIG_BLUEZ_BNEP=m
- CONFIG_BLUEZ_BNEP_MC_FILTER=y
- CONFIG_BLUEZ_BNEP_PROTO_FILTER=y
- CONFIG_BLUEZ_HIDP=m

  #
  # Bluetooth device drivers
  #
  CONFIG_BLUEZ_HCIUSB=m
! # CONFIG_BLUEZ_HCIUSB_SCO is not set
  CONFIG_BLUEZ_HCIUART=m
  CONFIG_BLUEZ_HCIUART_H4=y
```

```
- CONFIG_BLUEZ_HCIUART_BCSP=y
- CONFIG_BLUEZ_HCIBFUSB=m
  CONFIG_BLUEZ_HCIDTL1=m
! CONFIG_BLUEZ_HCIBT3C=m
! CONFIG_BLUEZ_HCIBLUECARD=m
! CONFIG_BLUEZ_HCIBTUART=m
! CONFIG_BLUEZ_HCIVHCI=m

  #
  # Kernel hacking
--- 1063,1085 ----
  # CONFIG_USB_AUERSWALD is not set
  # CONFIG_USB_BRLVGER is not set

  #
  # Bluetooth support
  #
  CONFIG_BLUEZ=m
  CONFIG_BLUEZ_L2CAP=m
  CONFIG_BLUEZ_SCO=m

  #
  # Bluetooth device drivers
  #
  CONFIG_BLUEZ_HCIUSB=m
! CONFIG_BLUEZ_USB_FW_LOAD=y
! CONFIG_BLUEZ_USB_ZERO_PACKET=y
  CONFIG_BLUEZ_HCIUART=m
  CONFIG_BLUEZ_HCIUART_H4=y
  CONFIG_BLUEZ_HCIDTL1=m
! # CONFIG_BLUEZ_HCIVHCI is not set

  #
  # Kernel hacking


24/05/2006
----------
----------

Made kernel and installed it along with modules. Found that config \
did not allow for serial_cs. Need to rejig config and reinstall \
kernel and modules. With no ip connection had to do this the hard \
way via minicom transfer files function. Revisited man pages for \
pcmcia cardmgr and cardctl.

Will look at /etc/pcmcia/config and bluetooth.conf and blueooth for options.

PRODID_1="Socket"
PRODID_2="CF+ Personal Network Card Rev 2.5"
PRODID_3=""
PRODID_4=""
MANFID=0104,0096
FUNCID=2
PRODID_1="SanDisk"
PRODID_2="SDP"
PRODID_3="5/3 0.6"
PRODID_4=""
MANFID=0045,0401
FUNCID=4

Did various options in bluetooth.conf with no success. Still got below error.
```

```
Will need to speak to author of patch.

~ # dmesg
at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00cfe38>]
Function entered at [<c00cf794>] from [<c00d290c>]
Function entered at [<c00d27f8>] from [<c00bc918>]
 r8 = C01912A8  r7 = C3AD1F48  r6 = C3B591C0  r5 = 00000100
 r4 = 00000000
Function entered at [<c00bc6d8>] from [<c007c630>]
Function entered at [<c007c550>] from [<c004b200>]
 r8 = C397F6A0  r7 = C0287320  r6 = 00000000  r5 = C390B040
 r4 = C3B591C0
Function entered at [<c004b0f8>] from [<c004b0f4>]
 r8 = C001A784  r7 = C02F4000  r6 = 00000000  r5 = 00000102
 r4 = 00000102
Function entered at [<c004b0a8>] from [<c004b448>]
 r4 = 00000003
Function entered at [<c004b404>] from [<c001a5e0>]
 r7 = 00000005  r6 = 00000002  r5 = 00012934  r4 = 00000001
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00cfe38>]
Function entered at [<c00cf794>] from [<c00d290c>]
Function entered at [<c00d27f8>] from [<c00bc918>]
 r8 = C01912A8  r7 = C3AD1F48  r6 = C3B591C0  r5 = 00000100
 r4 = 00000000
Function entered at [<c00bc6d8>] from [<c007c630>]
Function entered at [<c007c550>] from [<c004b200>]
 r8 = C397F6A0  r7 = C0287320  r6 = 00000000  r5 = C390B040
 r4 = C3B591C0
Function entered at [<c004b0f8>] from [<c004b0f4>]
 r8 = C001A784  r7 = C02F4000  r6 = 00000000  r5 = 00000102
 r4 = 00000102
Function entered at [<c004b0a8>] from [<c004b448>]
 r4 = 00000003
Function entered at [<c004b404>] from [<c001a5e0>]
 r7 = 00000005  r6 = 00000002  r5 = 00012934  r4 = 00000001
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFC40  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFC40
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFC40  r4 = C3B01000
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
```

```
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFC40  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFC40
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFC40  r4 = C3B01000
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFC40  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFC40
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFC40  r4 = C3B01000
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFC40  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFC40
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFC40  r4 = C3B01000
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFB5C  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFB5C
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFB5C  r4 = C3B01000
```

```
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFB5C  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFB5C
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFB5C  r4 = C3B01000
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFB5C  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFB5C
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFB5C  r4 = C3B01000
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
Division by zero in kernel.
Function entered at [<c001fc2c>] from [<c0160938>]
Function entered at [<c00d01f8>] from [<c00d1f28>]
Function entered at [<c00d1ed8>] from [<c00c0cd0>]
 r8 = C38FC000  r7 = 00000000  r6 = 00000000  r5 = C3B01000
 r4 = C38FDEB8
Function entered at [<c00c0af4>] from [<c00c0e60>]
 r6 = C3B01000  r5 = BFFFFB5C  r4 = 00000000
Function entered at [<c00c0cf0>] from [<c00c1234>]
 r8 = 00005402  r7 = C3B01000  r6 = 00000002  r5 = BFFFFB5C
 r4 = C3B01000
Function entered at [<c00c1010>] from [<c00bd870>]
 r5 = BFFFFB5C  r4 = C3B01000
Function entered at [<c00bd360>] from [<c005b2e4>]
Function entered at [<c005b0e0>] from [<c001a5e0>]
 r8 = C001A784  r7 = 00000036  r6 = 00000003  r5 = 00000000
 r4 = 00000A30
~ #
```

D.2 Excerpt Email Trail PCMCIA Bluetooth Card

```
Is there any setting or settings which should be turned off or on? \
I tried to make everything that I can modules so that if the module \
```

is bad the kernel will not be bad.

I ran make dep and make bzImage and lastly make modules.
Everything seem to compile (so it must work, right?).

Is there any further tests I can do before I put the kernel and the \
modules on the korebot. I plan to do this as per 6.4 of your the \
Korebot user manual.

many thanks for your help BTW.
ken.
*
*
*
*
About This Post Close This Box
 Status: Comment
 Rating: -
 Member Vists: 5
 Total Vists: 101
*
About Author Close This Box
 pbureau

 Email: pbureau@k-team.com
 Status: Registered User
 Bio:
 Homepage
 User Profile

*

 Subject: Re: PCMCIA errors
 From: pbureau
 Date: 05/11/06 04:38 PM
Tools Close This Box
 Reply post Reply post
 Edit post (Disabled) Edit post
 Go to parent Go to parent
 Copy post to Favorite (Disabled) Copy post to Favorite
 Send post to your friends (Disabled) Send post to your friends
Hi,

I think you can proceed with the Programming. If the new kernel is \
not booting at all, you will need a KoreJtag to reflash the board though...

Pierre
*
*
*
*
About This Post Close This Box
 Status: Comment
 Rating: -
 Member Vists: 11
 Total Vists: 11
*
User Profile not shown!

*

Subject: Re: PCMCIA errors
From: ken(anon)
Date: 05/16/06 04:39 AM
Tools Close This Box
Reply post Reply post
Edit post (Disabled) Edit post
Go to parent Go to parent
Copy post to Favorite (Disabled) Copy post to Favorite
Send post to your friends (Disabled) Send post to your friends
Hi,
I've managed to intall the 2.4.19-kb11 kernel and modules with the \
latests bluez patch applied. The kernel came up fine along with the \
modules.
The problem is the socket bluetooth card that I use was not recognized \
as a serial device. I looked in the pcmcia directory in the modules \
transferred across and found that no serial_cs.o existed.

I've read in http://pcmcia-cs.sourceforge.net/ftp/README-2.4 that it \
would be better to use the kernel PCMCIA. Is this what has been done \
for the kb11 kernel? If so, how do I use it? If not, is it just a case \
of me enable that config option when making the kernel and modules?

many thanks for your help
ken.
*
*
*
*
About This Post Close This Box
Status: Comment
Rating: -
Member Vists: 4
Total Vists: 87
*
About Author Close This Box
pbureau

Email: pbureau@k-team.com
Status: Registered User
Bio:
Homepage
User Profile

*

Subject: Re: PCMCIA errors
From: pbureau
Date: 05/23/06 07:47 AM
Tools Close This Box
Reply post Reply post
Edit post (Disabled) Edit post
Go to parent Go to parent
Copy post to Favorite (Disabled) Copy post to Favorite
Send post to your friends (Disabled) Send post to your friends
Hello,

Is this option available as a module? Could you tell me where exactly \
to find the option in the kernel configuration...

Best,

```
Pierre
*
*
*
*
About This Post Close This Box
 Status: Comment
 Rating: -
 Member Vists: 8
 Total Vists: 8
*
User Profile not shown!

*

 Subject: Re: PCMCIA errors
 From: Ken(anon)
 Date: 06/06/06 02:14 PM
Tools Close This Box
 Reply post Reply post
 Edit post (Disabled) Edit post
 Go to parent Go to parent
 Copy post to Favorite (Disabled) Copy post to Favorite
 Send post to your friends (Disabled) Send post to your friends
Hi,
What I did was reconfigure the kernel so the option was set for serial_cs.\
 I found that this also did not work.

I then tried a different tack. I bought an adapter for the usb master mini\
 connection on the korebot. I found that the usb bluetooth device worked \
first time. I tried a Belkin F8T013 device (Broadcomm chipset) and a \
generic model (CSR chipset) and both work.

Many thanks for you help. My feeling is to avoid the pcmcia bluetooth \
cards altogether for the moment. I think there is some serious problem \
at least in the 2.4.19 kernel which is not rectified by applying the \
Mark Holtman bluetooth patch. I beleive this is because of the serial_cs \
driver.

regards,
Ken.
...
```

D.3 Diary Excerpt USB Bluetooth Dongle

```
25/05/06
--------
--------

Success.

Went out today and purchased a bluetooth adapter. Belkin 8T013.
Attached this to the host computer Possibility.

Attached HCV Wireless bluetooth card to korebot via mini to standard usb \
adapter with a female plug at end.

Some of the appropriate messages occurred on startup.
Typed ‘‘modprobe hci_usb’’.
/lib/modules/2.4.19-kb11/kernel/drivers/bluetooth # modprobe hci_usb
```

```
Using /lib/modules/2.4.19-kb11/kernel/net/bluetooth/bluez.o
BlueZ Core ver 2.4 Copyright (C) 2000,2001 Qualcomm Inc
Written 2000,2001 by Maxim Krasnyansky <maxk@qualcomm.com>
Using /lib/modules/2.4.19-kb11/kernel/drivers/bluetooth/hci_usb.o
BlueZ HCI USB driver ver 2.7 Copyright (C) 2000,2001 Qualcomm Inc
Written 2000,2001 by Maxim Krasnyansky <maxk@qualcomm.com>
usb.c: registered new driver hci_usb

Typed ‘‘hciconfig’’ got

hci0:   Type: USB
        BD Address: 00:00:00:00:00:00 ACL MTU: 0:0  SCO MTU: 0:0
        DOWN
        RX bytes:0 acl:0 sco:0 events:0 errors:0
        TX bytes:0 acl:0 sco:0 commands:0 errors:0
Typed ‘‘hciconfig hci0 up’’ got same message with UP with some events
hci0:   Type: USB
        BD Address: 00:02:72:B0:C5:FB ACL MTU: 192:8  SCO MTU: 64:8
        UP RUNNING PSCAN ISCAN
        RX bytes:69 acl:0 sco:0 events:8 errors:0
        TX bytes:27 acl:0 sco:0 commands:7 errors:0

Typed ‘‘modprobe bnep’’ got:
/lib/modules/2.4.19-kb11/kernel/drivers/bluetooth # modprobe bnep
Using /lib/modules/2.4.19-kb11/kernel/net/bluetooth/l2cap.o
BlueZ L2CAP ver 2.3 Copyright (C) 2000,2001 Qualcomm Inc
Written 2000,2001 by Maxim Krasnyansky <maxk@qualcomm.com>
Using /lib/modules/2.4.19-kb11/kernel/net/bluetooth/bnep/bnep.o
BlueZ BNEP ver 1.2
Copyright (C) 2001,2002 Inventel Systemes
Written 2001,2002 by Clement Moreau <clement.moreau@inventel.fr>
Written 2001,2002 by David Libault <david.libault@inventel.fr>
Copyright (C) 2002 Maxim Krasnyanskiy <maxk@qualcomm.com>

Typed ‘‘hcitool dev’’ got
/lib/modules/2.4.19-kb11/kernel/drivers/bluetooth # hcitool dev
Devices:
        hci0    00:02:72:B0:C5:FB
Typed ‘‘hcitool dev’’ got
/lib/modules/2.4.19-kb11/kernel/drivers/bluetooth # hcitool scan
Scanning ...
        00:0A:3A:66:B8:B6       Possibility-0

Did device discovery on Possibility. Got Trippin device address \
00:02:72:B0:C5:FB.

It seems that Linux support for usb devices is more mature than that of \
pcmcia cards.

Next to do PAND

28/05/06
--------
--------

Swapped devices for usb bluetooth. Worked fine.

In /etc/bluetooth/hcid.conf changed script so default link mode now lm \
accept,master.

Could not get connection to the net. Will get it asap.
```

ken.

D.4 Personal Area Network HOWTO

See http://bluez.sourceforge.net/contrib/HOWTO-PAN

Appendix E

Java Virtual Machine

This appendix covers the porting of a Java virtual machine to the Korebot which runs an ARM processor.

E.1 Background

Until quite recently there were not many freely available implementations for Java virtual machines that ran on ARM processors. The most widely known proprietary Java virtual machine was IBM's java virtual machine.

E.2 SUN JVM

Since the open sourcing of the Sun Java virtual machine it is hoped that the porting process has been simplified. The author attempted before the open sourcing of the code to port the J2ME CDC version. This version is necessary for dynamic class loading, which is required for loading skills into the virtual machine. It was found to be almost impossible. From perusal of the Sun website it now appears there is at least one port of the J2ME CDC version to a Linux/ARM machine. The CLDC JVM was almost ported but ran foul of the United States export restrictions laws in respect of the secure socket layer (SSL).

E.3 JamVM

E.3.1 Making JamVM and GNU Classpath for the Korebot

To make the JamVM one requires the Korebot arm-linux-gcc 2.95.3 toolchain installed on your host machine. Next one needs to fetch the GNU Classpath Generics 0.91 library available from the GNU Classpath CVS repository. Run the following commands:

```
# CC=arm-linux-gcc ./configure --prefix=/mnt/hda/sw \
--disable-gtk-peer --host=arm-linux
# make
# make install
```

The highest JamVM version that can be used on the Korebot when compiling with the 2.95.3 toolchain is 1.4.3.

There is a problem with lines 54 and 58 of callNative.S. The gcc 2.95.3 toolchaine does not tolerate ' characters in code comments. These need to be removed. Run the following commands:

```
# CC=arm-linux-gcc ./configure --host=arm-linux \
--prefix=/mnt/hda/sw --with-classpath-install-dir=/mnt/hda/sw
# make
# make install
```

Zip up the sw directory and transfer it to the Korebot. Put the sw.zip in "/mnt/hda/sw" and unzip it. Then unzip glib.zip. You should have a working virtual machine.

E.3.2 Diary Excerpt JamVM 1.4.3 / GCP 0.91 Generics

```
13/07/07
--------
--------

...

Discovered error message relates to using 1.4 versus 1.5 java. Started \
trying to build it from source and ran into the same problem. Will now \
need to build using classpath-generics

Had problems with classpath-generics building. Will try later 0.95

Found problem in java/util/Collections.java at line 6579 should be:
      (Class<Map.Entry<K,V>>) (Class) Map.Entry.class;
Double cast is required because of Make.log

1598. ERROR in ../java/util/Collections.java (at line 6579)
(Class<Map.Entry<K,V>>) Map.Entry.class;
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
Cannot cast from Class<Map.Entry> to Class<Map.Entry<K,V>>


Compared 0.95 to 0.91 found need to cast it as Class first.
Installed to /usr/local/classpath

Now used eclipse java compiler
Confirmed in debian javac defaults to ecj in usr/bin/ecj

Had problems with pom.xml file found it worked just needed fork set to \
true in file
Got it working with generics.

Now to cross compile generics to target computer, Korebot.

16/07/07
--------
--------

zip -r sw sw

This zipped up all the files jamvm and classpath in sw

redid classpath and jamvm with /mnt/hda/sw

CC=arm-linux-gcc ./configure --prefix=/mnt/hda/sw --disable-gtk-peer \
--host=arm-linux
CC=arm-linux-gcc ./configure --host=arm-linux --prefix=/mnt/hda/sw \
--with-classpath-install-dir=/mnt/hda/sw

Able to Helloworld and the Helloworld using generics
```

```
18/07/07
--------
--------
Discovered ArrayBlockingQueue from jsr166 not in. In from 0.92 (bugger).\
Tried to make 0.95

CC=arm-linux-gcc ./configure --prefix=/home/krobinson/Desktop/sw/tmp/0.95\
 --disable-gtk-peer --disable-jni --disable-plugin --host=arm-linux

Read comment of 1.4.5. Need this jamvm for jsr166 collections to work.

CC=arm-linux-gcc ./configure --host=arm-linux --prefix=/mnt/hda/sw2 \
--with-classpath-install-dir=/mnt/hda/sw2

Problem with callNative.S 54 and 58

Proble again with 2.95.3 compiler did not like void return types. make \
return types which was throw away setDefaultInitArgs. This was done in jam.h,\
jam.c init.c

Did not work ran across this problem

 # /mnt/hda/sw2/bin/jamvm -verbose HWG
[Loaded java/lang/Object from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/Object]
[Loaded java/io/Serializable from /mnt/hda/sw2/share/classpath]
[Linking class java/io/Serializable]
[Loaded java/lang/reflect/Type from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/reflect/Type]
[Loaded java/lang/reflect/AnnotatedElement from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/reflect/AnnotatedElement]
[Loaded java/lang/reflect/GenericDeclaration from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/reflect/GenericDeclaration]
[Loaded java/lang/Class from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/Class]
[Loaded java/lang/Runnable from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/Runnable]
[Loaded java/lang/Thread from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/Thread]
[Loaded java/lang/VMThread from /mnt/hda/sw2/share/jamvm/classes]
[Linking class java/lang/VMThread]
[Loaded java/lang/Comparable from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/Comparable]
[Loaded java/lang/CharSequence from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/CharSequence]
[Loaded java/lang/String from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/String]
[Loaded java/lang/Cloneable from /mnt/hda/sw2/share/classpath]
[Linking class java/lang/Cloneable]
[Created primitive class char]
[Created array class [C]

21/07/07
--------
--------

CC=arm-linux-gcc ./configure --prefix= /mnt/hda/sw3 --disable-gtk-peer \
--disable-jni --disable-plugin --host=arm-linux


CC=arm-linux-gcc ./configure --host=arm-linux --prefix=/mnt/hda/sw3 \
--with-classpath-install-dir=/mnt/hda/sw3
```

```
./configure --disable-gtk-peer --disable-jni --disable-plugin

./configure

Still hung at the same spot. Did follow up post to Robert Lougher. Now\
trying to use gdb to find source of fault.

CC=arm-linux-gcc ./configure --prefix= /mnt/hda/sw4 --host=arm-linux

Tried cacoa 0.98

CC=arm-linux-gcc ./configure --host=arm-linux --prefix=/mnt/hda/sw3 \
--with-classpath-install-dir=/mnt/hda/sw3
Found it was picking up too many things in the host system thinking \
they were part of the cross compile environment.

Tried classpath 0.93 generics

For target.
----------

CC=arm-linux-gcc ./configure --prefix=/mnt/hda/sw5 --disable-gtk-peer \
--disable-gconf-peer --disable-plugin --host=arm-linux --target=arm-linux \
--disable-Werror

Why did this come up?
config.status: linking ./include/jni_md-x86-linux-gnu.h to include/jni_md.h

CC=arm-linux-gcc ./configure --host=arm-linux \
--prefix=/mnt/hda/sw-jamvm1_4_6-pre \
--with-classpath-install-dir=/mnt/hda/sw5 --target=arm-linux

For host
--------

./configure --prefix=/usr/local/sw --disable-gtk-peer --disable-gconf-peer\
--disable-plugin --disable-Werror

./configure --prefix=/usr/local/sw --with-classpath-install-dir=/usr/local/sw5
```

E.3.3 Email Trail later versions of JamVM and GCP

```
Gmail  Ken Robinson <kenrobinsonster@gmail.com>
jamvm 1.4.4 hangs
21 messages
Ken Robinson <kenrobinsonster@gmail.com>  Wed, Nov 15, 2006 at 12:50 PM
To: jamvm-general@lists.sourceforge.net
Hi,
I'm trying to get javm 1.4.4 running on an arm processor
Classpath 0.92
Finally compiled classpath-0.92 with following configure options:
CC=arm-linux-gcc ./configure --prefix=/sw --disable-gtk-peer
--disable-gconf-peer --disable-plugin --host=arm-linux
Did ``make'' and ``make install''

Jamvm 1.4.4
CC=arm-linux-gcc ./configure --host=arm-linux --prefix=/sw
--with-classpath-install-dir=/sw

Note my previous post where I had to eliminate single quotes ( ' )
```

from comments in *.S files and give a return type of an int instead of
void (this was to a function which set defaults for later use). This
is so I could cross-compile with a 2.95.3 toolchain.

Anyways I do a simple Hello World program which hangs.
When I do it with -verbose it seems to get up to
Loading C Class Array (I haven't got the exact message with me).

Any Ideas?
ken.
Michael Koch <konqueror@gmx.de> Wed, Nov 15, 2006 at 4:48 PM
To: Ken Robinson <kenrobinsonster@gmail.com>
Cc: jamvm-general@lists.sourceforge.net
[Quoted text hidden]
Can you attach gdb to see where it actually hangs?

Michael
--
http://www.worldforge.org/
Robert Lougher <rob.lougher@gmail.com> Wed, Nov 15, 2006 at 9:59 PM
To: Ken Robinson <kenrobinsonster@gmail.com>
Cc: jamvm-general@lists.sourceforge.net
Hi,

To help you I'll need some more details, e.g. ARM processor, Linux
version, and whether you're using glibc or uclibc. The _exact_ output
from running JamVM with -verbose:gc -verbose:class -verbose:jni would
also be useful.

Attaching gdb would be ideal, but I'm assuming you have no development
tools on the target. Depending on the details above, I may need to
ask you to compile in more tracing information.

Rob.
[Quoted text hidden]

    [Quoted text hidden]

    Take Surveys. Earn Cash. Influence the Future of IT
    Join SourceForge.net's Techsay panel and you'll get the chance to share your
    opinions on IT & business topics through brief surveys - and earn cash
    http://www.techsay.com/default.php?page=join.php&p=sourceforge&CID=DEVDEV
    ____
    Jamvm-general mailing list
    Jamvm-general@lists.sourceforge.net
    https://lists.sourceforge.net/lists/listinfo/jamvm-general

Ken Robinson <kenrobinsonster@gmail.com> Thu, Nov 16, 2006 at 12:47 AM
To: Robert Lougher <rob.lougher@gmail.com>
Hi Rob,
As requested:
1. XScale arm processor found on korebot
2. linux 2.4.19
3. glibc
4. Output on class
~ # /mnt/hda/sw/bin/jamvm -verbose HelloWorld
[Loaded java/lang/Object from /mnt/hda/sw/share/classpath]
[Linking class java/lang/Object]
[Loaded java/io/Serializable from /mnt/hda/sw/share/classpath]
[Linking class java/io/Serializable]
[Loaded java/lang/reflect/Type from /mnt/hda/sw/share/classpath]

```
[Linking class java/lang/reflect/Type]
[Loaded java/lang/reflect/AnnotatedElement from /mnt/hda/sw/share/classpath]
[Linking class java/lang/reflect/AnnotatedElement]
[Loaded java/lang/reflect/GenericDeclaration from /mnt/hda/sw/share/classpath]
[Linking class java/lang/reflect/GenericDeclaration]
[Loaded java/lang/Class from /mnt/hda/sw/share/classpath]
[Linking class java/lang/Class]
[Loaded java/lang/Runnable from /mnt/hda/sw/share/classpath]
[Linking class java/lang/Runnable]
[Loaded java/lang/Thread from /mnt/hda/sw/share/classpath]
[Linking class java/lang/Thread]
[Loaded java/lang/VMThread from /mnt/hda/sw/share/jamvm/classes]
[Linking class java/lang/VMThread]
[Loaded java/lang/Comparable from /mnt/hda/sw/share/classpath]
[Linking class java/lang/Comparable]
[Loaded java/lang/CharSequence from /mnt/hda/sw/share/classpath]
[Linking class java/lang/CharSequence]
[Loaded java/lang/String from /mnt/hda/sw/share/classpath]
[Linking class java/lang/String]
[Loaded java/lang/Cloneable from /mnt/hda/sw/share/classpath]
[Linking class java/lang/Cloneable]
[Created primitive class char]
[Created array class [C]

5. Output on gc and jnci gives nothing
ken
[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com>  Thu, Nov 16, 2006 at 3:21 AM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi,

Thanks for the info.  I'll work out where in the boot sequence it's
getting to, and see if I can see anything wrong...
[Quoted text hidden]
Ken Robinson <kenrobinsonster@gmail.com>  Sat, Jul 21, 2007 at 2:13 PM
To: Robert Lougher <rob.lougher@gmail.com>
Hi Robert,
I was wondering whether you had a chance to look at this problem.
When you didn't answer I went back to java 1.4.3
The difficulty is I now need to use generics and enumerations and so
have to use  classpath 0.95 and jam 1.4.5

I have run into the same problem.

Any ideas?
[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com>  Sun, Jul 22, 2007 at 5:53 AM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi Ken,

Yes, I did a major re-working of the threading code between 1.4.3 and
1.4.4 which appears to have introduced a number of regressions.  These
weren't reported in time for 1.4.5, so they're currently only
available via CVS.  Instructions for downloading can be found here:

http://developer.berlios.de/cvs/?group_id=6545

Let me know if this fixes things for you.

Rob.

P.S.  I'm not one of those developers who change perfectly good code
```

for the fun of it! The threading code was reliable, but its structure
was much the same since 1.0.0. I had to change it to support the JNI
invocation API and 1.5 stuff.
[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com> Sun, Jul 22, 2007 at 6:02 AM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi,

Yes, you reported the problem to me, but I couldn't track it down
without gdb traces... Did you see the long thread on threading
issues/mutexes, etc. on jamvm-general? If you didn't I wouldn't
bother going back and reading it, it got a bit nasty!

Rob.
[Quoted text hidden]
Ken Robinson <kenrobinsonster@gmail.com> Sun, Jul 22, 2007 at 7:12 AM
To: Robert Lougher <rob.lougher@gmail.com>
Thanks very much Rob,
I will give it a try. I tried to make gdb using 2.95.3 but ran into
even more problems with that.

Thanks for all your help.
ken.
[Quoted text hidden]
Ken Robinson <kenrobinsonster@gmail.com> Sun, Jul 22, 2007 at 7:17 AM
To: Robert Lougher <rob.lougher@gmail.com>
Sorry to trouble you once more. Does this work with classpath 0.95 or
do I need the classpath-generics 0.93 (I have it checked out already
and built)?
Also do I have to make sure jni is not disabled? I saw that in an earlier post
ken.
[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com> Sun, Jul 22, 2007 at 8:10 AM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi Ken,

It should work with both classpath-generics 0.93, and classpath-0.95
(classpath-0.95 is based on what was the generics release). Yes, you
need JNI support for Classpath...

Hope this helps,

Rob.
[Quoted text hidden]
Ken Robinson <kenrobinsonster@gmail.com> Sun, Jul 22, 2007 at 9:17 AM
To: Robert Lougher <rob.lougher@gmail.com>
Hi Rob,
It worked on my host box (ubuntu x86 2.6.20 gcc 4.1.2) but not on the
target (korebot arm 2.4.18 gcc 2.95.3). The verbose switch produced
the same problem that I've seen since jamvm 1.4.4 (See output below).

Does jamvm 1.4.3 work with the jsr166 classes? Failing that I'll leave
this for now and try a version of the library I'm using that does not
use jsr166 stuff. Thanks for all your help. Perhaps I'll rebuild the
whole of the korebot using a later toolchain.

ken.

Configuration
========
My configuration for gnu classpath 0.95 and jamvm cvs 1.4.6-pre was as follows:
CC=arm-linux-gcc ./configure --prefix=/mnt/hda/sw5 --disable-gtk-peer

```
--disable-gconf-peer --disable-plugin --host=arm-linux
--target=arm-linux --disable-Werror
Why did this come up?
config.status: linking ./include/jni_md-x86-linux-gnu.h to include/jni_md.h

CC=arm-linux-gcc ./configure --host=arm-linux --prefix=/mnt/hda/sw5
--with-classpath-install-dir=/mnt/hda/sw4 --target=arm-linux

Output
====
~ # /mnt/hda/sw5/bin/jamvm -verbose HWG
[Loaded java/lang/Object from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Object]
[Loaded java/io/Serializable from /mnt/hda/sw5/share/classpath]
[Linking class java/io/Serializable]
[Loaded java/lang/reflect/Type from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/reflect/Type]
[Loaded java/lang/reflect/AnnotatedElement from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/reflect/AnnotatedElement]
[Loaded java/lang/reflect/GenericDeclaration from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/reflect/GenericDeclaration]
[Loaded java/lang/Class from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Class]
[Loaded java/lang/Runnable from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Runnable]
[Loaded java/lang/Thread from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Thread]
[Loaded java/lang/VMThread from /mnt/hda/sw5/share/jamvm/classes]
[Linking class java/lang/VMThread]
[Loaded java/lang/Comparable from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Comparable]
[Loaded java/lang/CharSequence from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/CharSequence]
[Loaded java/lang/String from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/String]
[Loaded java/lang/Cloneable from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Cloneable]
[Created primitive class char]
[Created array class [C]

[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com>  Sun, Jul 22, 2007 at 10:10 AM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi,

No, 1.4.3 won't work with jsr166, as it needs sun.misc.Unsafe which I
only added n JamVM 1.4.5...  I'll go back to the original plan and see
if I can work out where in the startup it's hanging.

Could you do one experiment?  In thread.c, comment out all calls to
createVMThread (or make createVMThread an empty function) and then
rerun with -verbose.

This will ensure it is running single-threaded during startup...

Thanks,

Rob.
[Quoted text hidden]
Ken Robinson <kenrobinsonster@gmail.com>  Sun, Jul 22, 2007 at 10:38 AM
To: Robert Lougher <rob.lougher@gmail.com>
Hi Rob,
```

Here is the output. Still the same. Ran it with the gc and jni options
and it output nothing.

Thanks again for all your help. I remember trying to port the sun
version to the arm processor. Not good. I then used yours and it
compiled straight out of the box. Very good.

ken.

~ # /mnt/hda/sw-jamvm1_4_6-pre/bin/jamvm -verbose HWG
[Loaded java/lang/Object from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Object]
[Loaded java/io/Serializable from /mnt/hda/sw5/share/classpath]
[Linking class java/io/Serializable]
[Loaded java/lang/reflect/Type from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/reflect/Type]
[Loaded java/lang/reflect/AnnotatedElement from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/reflect/AnnotatedElement]
[Loaded java/lang/reflect/GenericDeclaration from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/reflect/GenericDeclaration]
[Loaded java/lang/Class from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Class]
[Loaded java/lang/Runnable from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Runnable]
[Loaded java/lang/Thread from /mnt/hda/sw5/share/classpath]
[Linking class java/lang/Thread]
[Loaded java/lang/VMThread from /mnt/hda/sw-jamvm1_4_6-pre/share/jamvm/classes]
[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com> Sun, Jul 22, 2007 at 4:48 PM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi Ken,

To help pinpoint where the hang is occurring, could you replace
interp.c with the attached and rerun? Also, it would be useful
turning on thread and lock tracing (either use configure options
--enable-tracethread --enable-tracelock or edit config.h).
[Quoted text hidden]

interp.c
73K
Robert Lougher <rob.lougher@gmail.com> Sun, Jul 22, 2007 at 7:00 PM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi Ken,

Sorry, a further point. Could you also add a fflush(stream) to the
end of jam_fprintf in hooks.c?   This will make sure you get all the
trace output.

Thanks,

Rob.
[Quoted text hidden]
Ken Robinson <kenrobinsonster@gmail.com> Sun, Jul 22, 2007 at 9:19 PM
To: Robert Lougher <rob.lougher@gmail.com>
Hi Rob,
I did all you suggested.

Here is the output with -verbose switch:

~ # /mnt/hda/sw-jamvm1_4_6-pre/bin/jamvm -verbose HWG
[Loaded java/lang/Object from /mnt/hda/sw5/share/classpath]

```
Thread 0x368bc lock on obj 0x402191d0...
[Linking class java/lang/Object]
Thread 0x368bc unlock on obj 0x402191d0...
[Loaded java/io/Serializable from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219138...
[Linking class java/io/Serializable]
Thread 0x368bc unlock on obj 0x40219138...
[Loaded java/lang/reflect/Type from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219268...
[Linking class java/lang/reflect/Type]
Thread 0x368bc unlock on obj 0x40219268...
[Loaded java/lang/reflect/AnnotatedElement from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219300...
[Linking class java/lang/reflect/AnnotatedElement]
Thread 0x368bc unlock on obj 0x40219300...
[Loaded java/lang/reflect/GenericDeclaration from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219398...
[Linking class java/lang/reflect/GenericDeclaration]
Thread 0x368bc unlock on obj 0x40219398...
[Loaded java/lang/Class from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x402190a0...
[Linking class java/lang/Class]
Thread 0x368bc unlock on obj 0x402190a0...
[Loaded java/lang/Runnable from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219430...
[Linking class java/lang/Runnable]
Thread 0x368bc unlock on obj 0x40219430...
[Loaded java/lang/Thread from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219008...
[Linking class java/lang/Thread]
Thread 0x368bc unlock on obj 0x40219008...
[Loaded java/lang/VMThread from /mnt/hda/sw-jamvm1_4_6-pre/share/jamvm/classes]
Thread 0x368bc lock on obj 0x402194c8...
[Linking class java/lang/VMThread]
Thread 0x368bc unlock on obj 0x402194c8...
[Loaded java/lang/Comparable from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219660...
[Linking class java/lang/Comparable]
Thread 0x368bc unlock on obj 0x40219660...
[Loaded java/lang/CharSequence from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x402196f8...
[Linking class java/lang/CharSequence]
Thread 0x368bc unlock on obj 0x402196f8...
[Loaded java/lang/String from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x402195c8...
[Linking class java/lang/String]
Thread 0x368bc unlock on obj 0x402195c8...
Thread 0x368bc lock on obj 0x402191d0...
Thread 0x368bc unlock on obj 0x402191d0...
Thread 0x368bc lock on obj 0x402191d0...
Thread 0x368bc NotifyAll on obj 0x402191d0...
Thread 0x368bc unlock on obj 0x402191d0...
[Loaded java/lang/Cloneable from /mnt/hda/sw5/share/classpath]
Thread 0x368bc lock on obj 0x40219828...
[Linking class java/lang/Cloneable]
Thread 0x368bc unlock on obj 0x40219828...
Thread 0x368bc lock on obj 0x40219828...
Thread 0x368bc unlock on obj 0x40219828...
Thread 0x368bc lock on obj 0x40219828...
Thread 0x368bc NotifyAll on obj 0x40219828...
Thread 0x368bc unlock on obj 0x40219828...
Thread 0x368bc lock on obj 0x40219138...
```

```
Thread 0x368bc unlock on obj 0x40219138...
Thread 0x368bc lock on obj 0x40219138...
Thread 0x368bc NotifyAll on obj 0x40219138...
Thread 0x368bc unlock on obj 0x40219138...
[Created primitive class char]
[Created array class [C]

            java/lang/Thread.<init>(Ljava/lang/VMThread;Ljava/lang/String;IZ)V

[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com>  Mon, Jul 23, 2007 at 1:20 AM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi Ken,

OK, it's hanging as soon as the VM tries to execute Java code.  Very,
very odd.  Could you replace interp-direct.h with the attached and
rerun?  To do this you need to define DIRECT_DEBUG.  Easiest way is to
edit config.h and add a line at the top:

#define DIRECT_DEBUG 1

Thanks,

Rob.

P.S. Sorry to do this to you, but without gdb this is the only way I
can track down where it is hanging...
[Quoted text hidden]

interp-direct.h
6K
Ken Robinson <kenrobinsonster@gmail.com>  Mon, Jul 23, 2007 at 11:06 AM
To: Robert Lougher <rob.lougher@gmail.com>
Hi Rob,
I did as you asked and got the following output. The last words "1 :
REDISPATCH 183 1 1"  repeats indefinitely.
ken.

~ # /mnt/hda/sw-jamvm1_4_6-pre/bin/jamvm -verbose HWG
[Loaded java/lang/Object from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x402191d0...
[Linking class java/lang/Object]
Thread 0x3e40c unlock on obj 0x402191d0...
[Loaded java/io/Serializable from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219138...
[Linking class java/io/Serializable]
Thread 0x3e40c unlock on obj 0x40219138...
[Loaded java/lang/reflect/Type from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219268...
[Linking class java/lang/reflect/Type]
Thread 0x3e40c unlock on obj 0x40219268...
[Loaded java/lang/reflect/AnnotatedElement from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219300...
[Linking class java/lang/reflect/AnnotatedElement]
Thread 0x3e40c unlock on obj 0x40219300...
[Loaded java/lang/reflect/GenericDeclaration from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219398...
[Linking class java/lang/reflect/GenericDeclaration]
Thread 0x3e40c unlock on obj 0x40219398...
[Loaded java/lang/Class from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x402190a0...
[Linking class java/lang/Class]
```

```
Thread 0x3e40c unlock on obj 0x402190a0...
[Loaded java/lang/Runnable from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219430...
[Linking class java/lang/Runnable]
Thread 0x3e40c unlock on obj 0x40219430...
[Loaded java/lang/Thread from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219008...
[Linking class java/lang/Thread]
Thread 0x3e40c unlock on obj 0x40219008...
[Loaded java/lang/VMThread from /mnt/hda/sw-jamvm1_4_6-pre/share/jamvm/classes]
Thread 0x3e40c lock on obj 0x402194c8...
[Linking class java/lang/VMThread]
Thread 0x3e40c unlock on obj 0x402194c8...
[Loaded java/lang/Comparable from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219660...
[Linking class java/lang/Comparable]
Thread 0x3e40c unlock on obj 0x40219660...
[Loaded java/lang/CharSequence from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x402196f8...
[Linking class java/lang/CharSequence]
Thread 0x3e40c unlock on obj 0x402196f8...
[Loaded java/lang/String from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x402195c8...
[Linking class java/lang/String]
Thread 0x3e40c unlock on obj 0x402195c8...
Thread 0x3e40c lock on obj 0x402191d0...
Thread 0x3e40c unlock on obj 0x402191d0...
Thread 0x3e40c lock on obj 0x402191d0...
Thread 0x3e40c NotifyAll on obj 0x402191d0...
Thread 0x3e40c unlock on obj 0x402191d0...
[Loaded java/lang/Cloneable from /mnt/hda/sw5/share/classpath]
Thread 0x3e40c lock on obj 0x40219828...
[Linking class java/lang/Cloneable]
Thread 0x3e40c unlock on obj 0x40219828...
Thread 0x3e40c lock on obj 0x40219828...
Thread 0x3e40c unlock on obj 0x40219828...
Thread 0x3e40c lock on obj 0x40219828...
Thread 0x3e40c NotifyAll on obj 0x40219828...
Thread 0x3e40c unlock on obj 0x40219828...
Thread 0x3e40c lock on obj 0x40219138...
Thread 0x3e40c unlock on obj 0x40219138...
Thread 0x3e40c lock on obj 0x40219138...
Thread 0x3e40c NotifyAll on obj 0x40219138...
Thread 0x3e40c unlock on obj 0x40219138...
[Created primitive class char]
[Created array class [C]

            java/lang/Thread.<init>(Ljava/lang/VMThread;Ljava/lang/String;IZ)V

0 : REDISPATCH 42 0 0
0 : DISPATCH 42 0 0
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
1 : REDISPATCH 183 1 1
[Quoted text hidden]
```

Ken Robinson <kenrobinsonster@gmail.com> Wed, Jul 25, 2007 at 12:47 AM
To: Robert Lougher <rob.lougher@gmail.com>
Sorry Robert. Did you get this post?
ken.
[Quoted text hidden]
Robert Lougher <rob.lougher@gmail.com> Wed, Jul 25, 2007 at 1:56 AM
To: Ken Robinson <kenrobinsonster@gmail.com>
Hi Ken,

Just a quick email to let you know I did get it. I spent a while
analysing where it was hanging, and I don't really understand it
(yet). I'll write a longer email with my analysis/questions, later,
as I'm just going out the door...

Thanks,

Rob.
[Quoted text hidden]

Appendix F

Jadabs

F.1 Email Trail

```
Gmail  Ken Robinson <kenrobinsonster@gmail.com>
Jadabs
5 messages
Ken Robinson <kenrobinsonster@gmail.com>  Wed, Nov 2, 2005 at 1:03 PM
To: frei@inf.ethz.ch
Hi Andreas,
I am presently doing a research masters part-time in Brisbane, Queensland.
I am looking at having robots form mobile ad hoc networks and then \
communicate; discovering services, transferring code, messaging and \
the like. I was originally looking at JXTA but through it mailing \
lists was lead to jadabs.

Jadabs has everything that I need (dynamic loading of jar files, \
weaving of new code, communication using JXTA over bluetooth).

I've downloaded jadabs both minimal and full, but cannot get it to \
work. I get the below error? Is there anything that I should be doing?

regards,
Ken Robinson

$ bash jadabs.sh
Knopflerfish OSGi framework, version <no version found>
Copyright 2003-2004 Knopflerfish. All Rights Reserved.

See http://www.knopflerfish.org for more information.
Loading xargs file c:\SW\jadabs\jadabs-1.0.0-SNAPSHOT\init.xargs
Installed and started: file:repository/log4j/jars
/log4j-1.2.8-osgi.jar (id#1)
Installed: file:repository/kobjects/jars/kxml2-2.1.9-osgi.jar (id#2)
Installed and started: file:repository/jadabs/jars/bundleloader-1.0.0-SNAPSHOT.j
ar (id#3)
Installed and started: file:repository/jadabs/jars/pluginloader-1.0.0-SNAPSHOT.j
ar (id#4)
0 [main] INFO  ch.ethz.jadabs.bundleLoader.BundleLoaderImpl - BundleLoader start
ing ...
1594 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - jadabs.starter
(The system cannot find the file specified)
java.io.FileNotFoundException: jadabs.starter (The system cannot find the file s
pecified)
        at java.io.FileInputStream.open(Native Method)
        at java.io.FileInputStream.<init>(Unknown Source)
        at java.io.FileInputStream.<init>(Unknown Source)
```

```
        at java.io.FileReader.<init>(Unknown Source)
        at ch.ethz.jadabs.pluginLoader.PluginLoaderImpl.init(PluginLoaderImpl.ja
va:127)
        at ch.ethz.jadabs.pluginLoader.PluginLoaderActivator.start(PluginLoaderA
ctivator.java:98)
        at org.knopflerfish.framework.BundleImpl$1.run(BundleImpl.java:279)
        at java.security.AccessController.doPrivileged(Native Method)
        at org.knopflerfish.framework.BundleImpl.start(BundleImpl.java:253)
        at org.knopflerfish.framework.Framework.launch(Framework.java:358)
        at org.knopflerfish.framework.Main.handleArgs(Main.java:283)
        at org.knopflerfish.framework.Main.main(Main.java:190)
1609 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - Loading of Plug
in  failed.
Framework launched

Andreas Frei <frei@inf.ethz.ch>  Thu, Nov 3, 2005 at 6:37 AM
To: Ken Robinson <kenrobinsonster@gmail.com>

Hi Ken



A File jadabs.starter File is needed, if it is not in the bin \
directory download the sources from CVS at berlios.de. An example\
 of the jadabs.starter is in the bin directory.



Cheers,

Andreas



Von: Ken Robinson [mailto:kenrobinsonster@gmail.com]
Gesendet: Mittwoch, 2. November 2005 04:03
An: frei@inf.ethz.ch
Betreff: Jadabs



Hi Andreas,
I am presently doing a research masters part-time in Brisbane, Queensland.
I am looking at having robots form mobile ad hoc networks and then \
communicate; discovering services, transferring code, messaging and \
the like. I was originally looking at JXTA but through it mailing \
lists was lead to jadabs.

Jadabs has everything that I need (dynamic loading of jar files, \
weaving of new code, communication using JXTA over bluetooth).

I've downloaded jadabs both minimal and full, but cannot get it to work. \
I get the below error? Is there anything that I should be doing?

regards,
Ken Robinson

$ bash jadabs.sh
Knopflerfish OSGi framework, version <no version found>
Copyright 2003-2004 Knopflerfish. All Rights Reserved.

See http://www.knopflerfish.org for more information.
```

```
Loading xargs file c:\SW\jadabs\jadabs-1.0.0-SNAPSHOT\init.xargs
Installed and started: file:repository/log4j/jars

/log4j-1.2.8-osgi.jar (id#1)
Installed: file:repository/kobjects/jars/kxml2-2.1.9-osgi.jar (id#2)
Installed and started: file:repository/jadabs/jars/bundleloader-1.0.0-SNAPSHOT.j
ar (id#3)
Installed and started: file:repository/jadabs/jars/pluginloader-1.0.0-SNAPSHOT.j
ar (id#4)
0 [main] INFO  ch.ethz.jadabs.bundleLoader.BundleLoaderImpl - BundleLoader start
ing ...
1594 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - jadabs.starter
(The system cannot find the file specified)
java.io.FileNotFoundException: jadabs.starter (The system cannot find the file s
pecified)
        at java.io.FileInputStream.open(Native Method)
        at java.io.FileInputStream.<init>(Unknown Source)
        at java.io.FileInputStream.<init>(Unknown Source)
        at java.io.FileReader.<init>(Unknown Source)
        at ch.ethz.jadabs.pluginLoader.PluginLoaderImpl.init(PluginLoaderImpl.ja
va:127)
        at ch.ethz.jadabs.pluginLoader.PluginLoaderActivator.start(PluginLoaderA
ctivator.java:98)
        at org.knopflerfish.framework.BundleImpl$1.run(BundleImpl.java:279)
        at java.security.AccessController.doPrivileged(Native Method)
        at org.knopflerfish.framework.BundleImpl.start(BundleImpl.java:253)
        at org.knopflerfish.framework.Framework.launch(Framework.java:358)
        at org.knopflerfish.framework.Main.handleArgs(Main.java:283)
        at org.knopflerfish.framework.Main.main(Main.java:190)
1609 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - Loading of Plug
in  failed.
Framework launched


Ken Robinson <kenrobinsonster@gmail.com>  Tue, Nov 8, 2005 at 10:26 PM
To: Andreas Frei <frei@inf.ethz.ch>
Hi Andreas,
I've tried the jadabs.sh from the cvs bin directory and get the same result

I then tried to build using maven but get the following error. This is
after I copy the plugins from
http://osgirepo.berlios.de/repository/maven/jars/ and put it in the
plugin directory in maven. Is there something basic I am doing wrong?
Do I need to all find and install those jar files mentioned below?
Should they go into the .maven/repository?

regards,
Ken Robnson


 __  __
|  \/  |__ _Apache__ ___
| |\/| / _` \ V / -_) ' \  ~ intelligent projects ~
|_|  |_\__,_|\_/\___|_||_|  v. 1.0.2

The build cannot continue because of the following unsatisfied dependencies:

commons-net-1.1.0.jar
jsch-0.1.5.jar
commons-jelly-tags-velocity-20030303.205659.jar
velocity-1.4-dev.jar
> On 11/2/05, Andreas Frei <frei@inf.ethz.ch> wrote:
> Hi Ken
```

>
>
>
> A File jadabs.starter File is needed, if it is not in the bin directory
> download the sources from CVS at berlios.de. An example of the
> jadabs.starter is in the bin directory.
>
>
>
> Cheers,
>
> Andreas
>
>
>
>   -----
>
> Von: Ken Robinson [mailto:kenrobinsonster@gmail.com]
> Gesendet: Mittwoch, 2. November 2005 04:03
> An: frei@inf.ethz.ch
> Betreff: Jadabs
>
>
>
> Hi Andreas,
> I am presently doing a research masters part-time in Brisbane, Queensland.
> I am looking at having robots form mobile ad hoc networks and then
> communicate; discovering services, transferring code, messaging and the
> like.
> I was originally looking at JXTA but through it mailing lists was lead to
> jadabs.
>
> Jadabs has everything that I need (dynamic loading of jar files, weaving of
> new code, communication using JXTA over bluetooth).
>
> I've downloaded jadabs both minimal and full, but cannot get it to work. I
> get the below error? Is there anything that I should be doing?
>
> regards,
> Ken Robinson
>
> $ bash jadabs.sh
> Knopflerfish OSGi framework, version <no version found>
> Copyright 2003-2004 Knopflerfish. All Rights Reserved.
>
> See http://www.knopflerfish.org <http://www.knopflerfish.org/> for more
> information.
> Loading xargs file c:\SW\jadabs\jadabs-1.0.0-SNAPSHOT\init.xargs
> Installed and started: file:repository/log4j/jars
>
> /log4j-1.2.8-osgi.jar (id#1)
> Installed: file:repository/kobjects/jars/kxml2-2.1.9-osgi.jar (id#2)
> Installed and started:
> file:repository/jadabs/jars/bundleloader-1.0.0-SNAPSHOT.j
> ar (id#3)
> Installed and started:
> file:repository/jadabs/jars/pluginloader-1.0.0-SNAPSHOT.j
> ar (id#4)
> 0 [main] INFO ch.ethz.jadabs.bundleLoader.BundleLoaderImpl - BundleLoader
> start
> ing ...
> 1594 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl -

```
> jadabs.starter
> (The system cannot find the file specified)
> java.io.FileNotFoundException: jadabs.starter (The system cannot find the
> file s
> pecified)
>         at java.io.FileInputStream.open(Native Method)
>         at java.io.FileInputStream.<init>(Unknown Source)
>         at java.io.FileInputStream.<init>(Unknown Source)
>         at java.io.FileReader.<init>(Unknown Source)
>         at
> ch.ethz.jadabs.pluginLoader.PluginLoaderImpl.init(PluginLoaderImpl.ja
> va:127)
>         at
> ch.ethz.jadabs.pluginLoader.PluginLoaderActivator.start(PluginLoaderA
> ctivator.java:98)
>         at org.knopflerfish.framework.BundleImpl$1.run(BundleImpl.java:279)
>         at java.security.AccessController.doPrivileged(Native Method)
>         at org.knopflerfish.framework.BundleImpl.start(BundleImpl.java:253)
>         at org.knopflerfish.framework.Framework.launch(Framework.java:358)
>         at org.knopflerfish.framework.Main.handleArgs(Main.java:283)
>         at org.knopflerfish.framework.Main.main(Main.java:190)
> 1609 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - Loading of
> Plug
> in  failed.
> Framework launched
>
>
>
>
>
Ken Robinson <kenrobinsonster@gmail.com>  Tue, Nov 8, 2005 at 10:34 PM
To: jadabs-dev@lists.berlios.de
I've tried the jadabs.sh from the cvs bin directory and get the same result

I then tried to build using maven but get the following error. This is
after I copy the plugins from
http://osgirepo.berlios.de/repository/maven/jars/ and put it in the
plugin directory in maven. Is there something basic I am doing wrong?
Do I need to all find and install those jar files mentioned below?
Should they go into the .maven/repository?

regards,
Ken Robnson


 __  __
|  \/  |__ _Apache__ ___
| |\/| / _` \ V / -_) ' \  ~ intelligent projects ~
|_|  |_\__,_|\_/\___|_||_|  v. 1.0.2

The build cannot continue because of the following unsatisfied dependencies:

commons-net-1.1.0.jar
jsch-0.1.5.jar
commons-jelly-tags-velocity-20030303.205659.jar
velocity-1.4-dev.jar
> On 11/2/05, Andreas Frei <frei@inf.ethz.ch> wrote:
> Hi Ken
>
>
>
> A File jadabs.starter File is needed, if it is not in the bin directory
> download the sources from CVS at berlios.de. An example of the
```

> jadabs.starter is in the bin directory.
>
>
>
> Cheers,
>
> Andreas
>
>
>
> -----
>
> Von: Ken Robinson [mailto:kenrobinsonster@gmail.com]
> Gesendet: Mittwoch, 2. November 2005 04:03
> An: frei@inf.ethz.ch
> Betreff: Jadabs
>
>
>
> Hi Andreas,
> I am presently doing a research masters part-time in Brisbane, Queensland.
> I am looking at having robots form mobile ad hoc networks and then
> communicate; discovering services, transferring code, messaging and the
> like.
> I was originally looking at JXTA but through it mailing lists was lead to
> jadabs.
>
> Jadabs has everything that I need (dynamic loading of jar files, weaving of
> new code, communication using JXTA over bluetooth).
>
> I've downloaded jadabs both minimal and full, but cannot get it to work. I
> get the below error? Is there anything that I should be doing?
>
> regards,
> Ken Robinson
>
> $ bash jadabs.sh
> Knopflerfish OSGi framework, version <no version found>
> Copyright 2003-2004 Knopflerfish. All Rights Reserved.
>
> See http://www.knopflerfish.org <http://www.knopflerfish.org/> for more
> information.
> Loading xargs file c:\SW\jadabs\jadabs-1.0.0-SNAPSHOT\init.xargs
> Installed and started: file:repository/log4j/jars
>
> /log4j-1.2.8-osgi.jar (id#1)
> Installed: file:repository/kobjects/jars/kxml2-2.1.9-osgi.jar (id#2)
> Installed and started:
> file:repository/jadabs/jars/bundleloader-1.0.0-SNAPSHOT.j
> ar (id#3)
> Installed and started:
> file:repository/jadabs/jars/pluginloader-1.0.0-SNAPSHOT.j
> ar (id#4)
> 0 [main] INFO ch.ethz.jadabs.bundleLoader.BundleLoaderImpl - BundleLoader
> start
> ing ...
> 1594 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl -
> jadabs.starter
> (The system cannot find the file specified)
> java.io.FileNotFoundException: jadabs.starter (The system cannot find the
> file s
> pecified)

```
>         at java.io.FileInputStream.open(Native Method)
>         at java.io.FileInputStream.<init>(Unknown Source)
>         at java.io.FileInputStream.<init>(Unknown Source)
>         at java.io.FileReader.<init>(Unknown Source)
>         at
> ch.ethz.jadabs.pluginLoader.PluginLoaderImpl.init(PluginLoaderImpl.ja
> va:127)
>         at
> ch.ethz.jadabs.pluginLoader.PluginLoaderActivator.start(PluginLoaderA
> ctivator.java:98)
>         at org.knopflerfish.framework.BundleImpl$1.run(BundleImpl.java:279)
>         at java.security.AccessController.doPrivileged(Native Method)
>         at org.knopflerfish.framework.BundleImpl.start(BundleImpl.java:253)
>         at org.knopflerfish.framework.Framework.launch(Framework.java:358)
>         at org.knopflerfish.framework.Main.handleArgs(Main.java:283)
>         at org.knopflerfish.framework.Main.main(Main.java:190)
> 1609 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - Loading of
> Plug
> in  failed.
> Framework launched
>
>
>
>
>
Ken Robinson <kenrobinsonster@gmail.com>  Mon, Nov 28, 2005 at 12:29 PM
To: jadabs-dev@lists.berlios.de
Hi Andreas,
I am now trying to build jadabs using maven. I get a compilation error as detailed \
below, not knowing how to make osgi:install. Can you help?

jadabs:buildbundles:
multiproject:projects-init:
Gathering project list
Starting the reactor...
Our processing order:
Benchmark Service
Jadabs-Concurrent
RemoteFramework API
Log4j-CDC
JXME-OSGi
JXME-Services API
RemoteFramework Impl
jadabsgui
PPC-Admin
TestMain
JXME-Services-Impl
JXME-TCP
Jadabs-JXME-TCP-Test
Jadabs-JXME-UDP
Jadabs-JXME-UDP-Test
Messenger
Messenger-Test
MobileServices-SMTPGW
MobileServices-SMTPGW-Test
MobileServices-SMTPGW-Test-TCP
Shell
+----------------------------------------
| Gathering project list Benchmark Service
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
```

```
| Gathering project list Jadabs-Concurrent
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list RemoteFramework API
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Log4j-CDC
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list JXME-OSGi
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list JXME-Services API
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list RemoteFramework Impl
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list jadabsgui
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list PPC-Admin
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list TestMain
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list JXME-Services-Impl
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list JXME-TCP
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Jadabs-JXME-TCP-Test
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Jadabs-JXME-UDP
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Jadabs-JXME-UDP-Test
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Messenger
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Messenger-Test
| Memory: 4M/5M
```

```
+----------------------------------------
+----------------------------------------
| Gathering project list MobileServices-SMTPGW
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list MobileServices-SMTPGW-Test
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list MobileServices-SMTPGW-Test-TCP
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Shell
| Memory: 4M/5M
+----------------------------------------

multiproject:goal:
Starting the reactor...
Our processing order:
Benchmark Service
Jadabs-Concurrent
RemoteFramework API
Log4j-CDC
JXME-OSGi
JXME-Services API
RemoteFramework Impl
jadabsgui
PPC-Admin
TestMain
JXME-Services-Impl
JXME-TCP
Jadabs-JXME-TCP-Test
Jadabs-JXME-UDP
Jadabs-JXME-UDP-Test
Messenger
Messenger-Test
MobileServices-SMTPGW
MobileServices-SMTPGW-Test
MobileServices-SMTPGW-Test-TCP
Shell
+----------------------------------------
| Executing osgi:install Benchmark Service
| Memory: 4M/5M
+----------------------------------------
DEPRECATED: the default goal should be specified in the <build> section of \
project.xml instead of maven.xml

BUILD FAILED
File..... /Users/kp_robinson/.maven/cache/maven-multiproject-plugin-1.4.1/plugin.jelly
Element... maven:reactor
Line..... 218
Column.... -1
Unknown goal "osgi:install"
Total time   : 14 seconds
Finished at  : Sunday, November 27, 2005 9:55:37 PM EST


Compilation exited abnormally with code 70 at Sun Nov 27 21:55:37

On 11/8/05, Ken Robinson <kenrobinsonster@gmail.com > wrote:
```

I've tried the jadabs.sh from the cvs bin directory and get the same result

I then tried to build using maven but get the following error. This is
after I copy the plugins from
http://osgirepo.berlios.de/repository/maven/jars/ and put it in the
plugin directory in maven. Is there something basic I am doing wrong?
Do I need to all find and install those jar files mentioned below?
Should they go into the .maven/repository?

regards,
Ken Robnson

```
__  __
|  \/  |__ _Apache__ ___
| |\/| / _` \ V / -_) ' \  ~ intelligent projects ~
|_|  |_\__,_|\_/\___|_||_|  v. 1.0.2
```

The build cannot continue because of the following unsatisfied dependencies:

commons-net-1.1.0.jar
jsch-0.1.5.jar
commons-jelly-tags-velocity-20030303.205659.jar
velocity-1.4-dev.jar
> On 11/2/05, Andreas Frei <frei@inf.ethz.ch> wrote:
> Hi Ken
>
>
>
> A File jadabs.starter File is needed, if it is not in the bin directory
> download the sources from CVS at berlios.de. An example of the
> jadabs.starter is in the bin directory.
>
>
>
> Cheers,
>
> Andreas
>
>
>
>   -----
>
> Von: Ken Robinson [mailto: kenrobinsonster@gmail.com]
> Gesendet: Mittwoch, 2. November 2005 04:03
> An: frei@inf.ethz.ch
> Betreff: Jadabs
>
>
>
> Hi Andreas,
> I am presently doing a research masters part-time in Brisbane, Queensland.
> I am looking at having robots form mobile ad hoc networks and then
> communicate; discovering services, transferring code, messaging and the
> like.
> I was originally looking at JXTA but through it mailing lists was lead to
> jadabs.
>
> Jadabs has everything that I need (dynamic loading of jar files, weaving of
> new code, communication using JXTA over bluetooth).
>
> I've downloaded jadabs both minimal and full, but cannot get it to work. I

```
> get the below error? Is there anything that I should be doing?
>
> regards,
> Ken Robinson
>
> $ bash jadabs.sh
> Knopflerfish OSGi framework, version <no version found>
> Copyright 2003-2004 Knopflerfish. All Rights Reserved.
>
> See http://www.knopflerfish.org <http://www.knopflerfish.org/ >  for more
> information.
> Loading xargs file c:\SW\jadabs\jadabs-1.0.0-SNAPSHOT\init.xargs
> Installed and started: file:repository/log4j/jars
>
> /log4j-1.2.8-osgi.jar (id#1)
> Installed: file:repository/kobjects/jars/kxml2- 2.1.9-osgi.jar (id#2)
> Installed and started:
> file:repository/jadabs/jars/bundleloader-1.0.0-SNAPSHOT.j
> ar (id#3)
> Installed and started:
> file:repository/jadabs/jars/pluginloader-1.0.0-SNAPSHOT.j
> ar (id#4)
> 0 [main] INFO  ch.ethz.jadabs.bundleLoader.BundleLoaderImpl - BundleLoader
> start
> ing ...
> 1594 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl -
> jadabs.starter
> (The system cannot find the file specified)
> java.io.FileNotFoundException: jadabs.starter (The system cannot find the
> file s
> pecified)
>         at java.io.FileInputStream.open(Native Method)
>         at java.io.FileInputStream.<init>(Unknown Source)
>         at java.io.FileInputStream.<init>(Unknown Source)
>         at java.io.FileReader.<init>(Unknown Source)
>         at
> ch.ethz.jadabs.pluginLoader.PluginLoaderImpl.init(PluginLoaderImpl.ja
> va:127)
>         at
> ch.ethz.jadabs.pluginLoader.PluginLoaderActivator.start(PluginLoaderA
> ctivator.java:98)
>         at org.knopflerfish.framework.BundleImpl$1.run(BundleImpl.java:279)
>         at java.security.AccessController.doPrivileged(Native Method)
>         at org.knopflerfish.framework.BundleImpl.start (BundleImpl.java:253)
>         at org.knopflerfish.framework.Framework.launch(Framework.java:358)
>         at org.knopflerfish.framework.Main.handleArgs(Main.java:283)
>         at org.knopflerfish.framework.Main.main (Main.java:190)
> 1609 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - Loading of
> Plug
> in  failed.
> Framework launched
>
>
>
>
>
```

F.2 Diary excerpt re Jadabs

```
15/09/05
--------
```

```
--------

Tested out Jadabs.
Done by using JamVM 1.3.3, GNU_Classpath 0.18 and jikes 1.22

without-gtk-peers with-jikes

Got up to starting of jadabs after changing of java to jamvm.
Hung.

19/09/05
--------
--------

Tried to run jadabs 1.0.0 version both minimal and prose version using bash \
jadabs.sh and bash jadabs.sh. Got launch of framework but
following error message:
27060 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - Loading of \
Plugin  failed.

Will now try building it from stratch.

Downloaded Maven 1.0.2
Install instructions http://maven.apache.org/start/install.html
- Set MAVEN_HOME in set-maven-environment

ToDo - Check classpath for GNU_Classpath. May already be covered by JamVM.

20/09/05
--------
--------

Tried to compile. Problem with maven compiling it. Should now try compiling it with java from sun.

Problem with /usr/local not having bin, jre and lib

22/09/05
--------
Decided to go with SDK development kit but ran into configure problems. libtdl is missing.

27/09/05
--------

Can't use jdk 1.3.1. Can't use 1.5.0. Need to download 1.4.2

29/09/05
--------
Installed 1.5.0. Can use it. downloaded jadabs as per jadabs website instructions.\
 started installing maven. Need dependencies.

03/10/05
--------

Used instructions from berlios jadabas website to grab cvs jadabs source tree.\
 Copied TBD jar file. cd'ed into jadabs tree. Typed ``maven jadabs''. \
Did not work. Need all of them need to be fixed up.

31/10/05
--------

$ bash jadabs.sh
Knopflerfish OSGi framework, version <no version found>
```

```
Copyright 2003-2004 Knopflerfish. All Rights Reserved.

See http://www.knopflerfish.org for more information.
Loading xargs file c:\SW\jadabs\jadabs-1.0.0-SNAPSHOT\init.xargs
Installed and started: file:repository/log4j/jars/log4j-1.2.8-osgi.jar (id#1)
Installed: file:repository/kobjects/jars/kxml2-2.1.9-osgi.jar (id#2)
Installed and started: file:repository/jadabs/jars/bundleloader-1.0.0-SNAPSHOT.j
ar (id#3)
Installed and started: file:repository/jadabs/jars/pluginloader-1.0.0-SNAPSHOT.j
ar (id#4)
0 [main] INFO  ch.ethz.jadabs.bundleLoader.BundleLoaderImpl - BundleLoader start
ing ...
1594 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - jadabs.starter
(The system cannot find the file specified)
java.io.FileNotFoundException: jadabs.starter (The system cannot find the file s
pecified)
        at java.io.FileInputStream.open(Native Method)
        at java.io.FileInputStream.<init>(Unknown Source)
        at java.io.FileInputStream.<init>(Unknown Source)
        at java.io.FileReader.<init>(Unknown Source)
        at ch.ethz.jadabs.pluginLoader.PluginLoaderImpl.init(PluginLoaderImpl.ja
va:127)
        at ch.ethz.jadabs.pluginLoader.PluginLoaderActivator.start(PluginLoaderA
ctivator.java:98)
        at org.knopflerfish.framework.BundleImpl$1.run(BundleImpl.java:279)
        at java.security.AccessController.doPrivileged(Native Method)
        at org.knopflerfish.framework.BundleImpl.start(BundleImpl.java:253)
        at org.knopflerfish.framework.Framework.launch(Framework.java:358)
        at org.knopflerfish.framework.Main.handleArgs(Main.java:283)
        at org.knopflerfish.framework.Main.main(Main.java:190)
1609 [main] ERROR ch.ethz.jadabs.pluginLoader.PluginLoaderImpl - Loading of Plug
in  failed.
Framework launched

Got this error. Looked at mailing list. Found nothing. Will need to email \
project people.

09/11/05
--------
Tried to build using maven. Appears to be some problem with debian not \
resolving names or downloading files from remote repository.

17/11/05
--------
Started building using maven. Running into problem using Linux where \
download does not occur from the remote repository. Going through compile.

27/11/05
--------

maven -E
 __  __
|  \/  |__ _Apache__ ___
| |\/| / _` \ V / -_) ' \  ~ intelligent projects ~
|_|  |_\__,_|\_/\___|_||_|  v. 1.1-beta-2

DEPRECATED: the default goal should be specified in the <build> section \
of project.xml instead of maven.xml
Starting the reactor...
Our processing order:

+----------------------------------------
```

```
| Dependency download null
| Memory: 2M/3M
+----------------------------------------
DEPRECATED: the default goal should be specified in the <build> section \
of project.xml instead of maven.xml
build:start:

jadabs:buildall:
deps:download:
build:start:

java:prepare-filesystem:

java:compile:
Compiling to /Users/kp_robinson/Desktop/SRC/play/jadabs/jadabs/target/classes
No java source files to compile.

build:end:


jadabs:buildbundles:
multiproject:projects-init:
Gathering project list
Starting the reactor...
Our processing order:
Benchmark Service
Jadabs-Concurrent
RemoteFramework API
Log4j-CDC
JXME-OSGi
JXME-Services API
RemoteFramework Impl
jadabsgui
PPC-Admin
TestMain
JXME-Services-Impl
JXME-TCP
Jadabs-JXME-TCP-Test
Jadabs-JXME-UDP
Jadabs-JXME-UDP-Test
Messenger
Messenger-Test
MobileServices-SMTPGW
MobileServices-SMTPGW-Test
MobileServices-SMTPGW-Test-TCP
Shell
+----------------------------------------
| Gathering project list Benchmark Service
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Jadabs-Concurrent
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list RemoteFramework API
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Log4j-CDC
| Memory: 4M/5M
+----------------------------------------
```

```
+---------------------------------------
| Gathering project list JXME-OSGi
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list JXME-Services API
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list RemoteFramework Impl
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list jadabsgui
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list PPC-Admin
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list TestMain
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list JXME-Services-Impl
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list JXME-TCP
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list Jadabs-JXME-TCP-Test
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list Jadabs-JXME-UDP
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list Jadabs-JXME-UDP-Test
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list Messenger
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list Messenger-Test
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list MobileServices-SMTPGW
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list MobileServices-SMTPGW-Test
| Memory: 4M/5M
+---------------------------------------
+---------------------------------------
| Gathering project list MobileServices-SMTPGW-Test-TCP
```

```
| Memory: 4M/5M
+----------------------------------------
+----------------------------------------
| Gathering project list Shell
| Memory: 4M/5M
+----------------------------------------

multiproject:goal:
Starting the reactor...
Our processing order:
Benchmark Service
Jadabs-Concurrent
RemoteFramework API
Log4j-CDC
JXME-OSGi
JXME-Services API
RemoteFramework Impl
jadabsgui
PPC-Admin
TestMain
JXME-Services-Impl
JXME-TCP
Jadabs-JXME-TCP-Test
Jadabs-JXME-UDP
Jadabs-JXME-UDP-Test
Messenger
Messenger-Test
MobileServices-SMTPGW
MobileServices-SMTPGW-Test
MobileServices-SMTPGW-Test-TCP
Shell
+----------------------------------------
| Executing osgi:install Benchmark Service
| Memory: 4M/5M
+----------------------------------------
DEPRECATED: the default goal should be specified in the <build> section of \
project.xml instead of maven.xml

BUILD FAILED
File...... /Users/kp_robinson/.maven/cache/maven-multiproject-plugin-1.4.1/\
plugin.jelly
Element... maven:reactor
Line...... 218
Column.... -1
Unknown goal "osgi:install"
Total time   : 14 seconds
Finished at  : Sunday, November 27, 2005 9:55:37 PM EST
```

Appendix G

Jython

```
23/10/06
--------
--------

...

Downloaded Jython lastest version jython_Release_2_2alpha.jar
Transferred.

Installed jython using
/sw/bin/jamvm -jar jython_Release_2_2alpha.jar
Installed to /sw/jython

Changed jython script to call jamvm instead of java

Got
Jython 2.2a1 on java1.4.2 (JIT: )
pc : [<0000cad8>]    lr : [<00019adc>]    Not tainted
sp : bffff048  ip : bffff058  fp : bffff068
r10: 0003a500  r9 : 00000000  r8 : 00000000
r7 : 0003a4e8  r6 : 0003a4ec  r5 : 000ffcd0  r4 : 0003a4e8
r3 : 00000108  r2 : 00000000  r1 : 00000000  r0 : 00000000
Flags: nzcv  IRQs on  FIQs on  Mode USER_32  Segment user
Control: 397F  Table: A3DD8000  DAC: 00000015
SIGSEGV

So whenever any user level program recevies segfault , kernel prints
Register dump at the time of Segfault having values like PC etc.
And program terminates.

[Linking class site$py]
Segmentation fault

Done when used debian jamvm and jython on host box

Similar thing when using debian jam and downloaded jython on host box
```

Did not happen with java from sun.

May have to use jythonc until bug fixed

25/10/06

Compiled jamvm 1.4.2 and GNU Classpath 0.20. Did not work with \
jython2.2.alpha1 at all.

Changed property of registry within jython so compiler javac pointer to \
symbolic link in /usr/bin/javac which was gcj

Got compiler to do print ‘‘hello world’’

Had trouble with class HelloWorld.

Question whether it would be better to use good ide as code is unreadable

Appendix H

OSGi

There are many OSGi implementations. The one we have chosen was the Apache Felix implementation. This was for a number of reasons. The first is it uses version 1.4.2 of the Java programming language. Version 1.4.2 (also known as J2ME) does not need a computer with as much memory and processing power as later versions. It is ideal for computers with some memory and processing power; the Korebot falls into this class. The second reason is the Apache Felix implementation does not require a GUI framework to interact with the framework, a textual user interface (TUI) is used. The Korebot has only a serial terminal display and so requires a TUI.

To install the OSGi framework go to the Apache Felix website. Download the latest implementation. Unzip the directory. Go into the unzipped directory. Enter the command "java -jar bin/felix.jar". This starts up the Apache Felix OSGi framework

```
# java -jar bin/felix.jar
-> ps
START LEVEL 1
   ID   State         Level  Name
[   0] [Active     ] [    0] System Bundle (1.0.0)
[   1] [Active     ] [    1] Apache Felix Shell Service (1.0.0)
[   2] [Active     ] [    1] Apache Felix Shell TUI (1.0.0)
[   3] [Active     ] [    1] Apache Felix Bundle Repository (1.0.0)
[   4] [Active     ] [    1] Bluetooth Service (0.1)
[   5] [Active     ] [    1] JXTA Service (0.1)
[   6] [Resolved   ] [    1] RANET Application (0.1)
```

Each row represents a bundle along with its present status. Each bundle is made using maven, an open source build tool (http://maven.apache.org website). The code and any dependencies are generated using an XML file called pom.xml. The one shown is a fairly detailed pom.xml, referring to dependencies, the OSGi plugin and an inlined artifact.

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <packaging>osgi-bundle</packaging>
  <name>JXSE MC service</name>
  <groupId>au.edu.qut.ranet</groupId>
  <artifactId>ranet.jxseMC.service</artifactId>
  <version>0.1</version>
  <description>JXSE MC service</description>
  <dependencies>
    <dependency>
      <groupId>org.apache.felix</groupId>
      <artifactId>org.osgi.core</artifactId>
      <version>1.0.0</version>
```

```
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>au.edu.qut.ranet</groupId>
    <artifactId>ranet.jxseMC.impl</artifactId>
    <version>0.1</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>au.edu.qut.ranet</groupId>
    <artifactId>ranet.jxta.swixml</artifactId>
    <version>2.3.6</version>
    <scope>runtime</scope>
  </dependency>
  <dependency>
    <groupId>au.edu.qut.ranet</groupId>
    <artifactId>ranet.jxse.bcprov-jdk14</artifactId>
    <version>2.5_rc3</version>
    <scope>runtime</scope>
  </dependency>
  <dependency>
    <groupId>au.edu.qut.ranet</groupId>
    <artifactId>ranet.jxse.javax.servlet</artifactId>
    <version>2.5_rc3</version>
    <scope>runtime</scope>
  </dependency>
  <dependency>
    <groupId>au.edu.qut.ranet</groupId>
    <artifactId>ranet.jxse</artifactId>
    <version>2.5_rc3</version>
    <scope>runtime</scope>
  </dependency>
  <dependency>
    <groupId>au.edu.qut.ranet</groupId>
    <artifactId>ranet.jxse.org.mortbay.jetty</artifactId>
    <version>2.5_rc3</version>
    <scope>runtime</scope>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
    <scope>test</scope>
  </dependency>
</dependencies>
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.felix.plugins</groupId>
      <artifactId>maven-osgi-plugin</artifactId>
      <extensions>true</extensions>
      <version>0.9.0-SNAPSHOT</version>
      <configuration>
         <inlinedArtifacts>
            <inlinedArtifact>ranet.jxseMC.impl</inlinedArtifact>
         </inlinedArtifacts>
        <osgiManifest>
          <bundleName>JXSE MC service</bundleName>
          <bundleVendor>Ken Robinson</bundleVendor>
          <bundleDescription>JXSE MC service</bundleDescription>
          <bundleActivator>
            au.edu.qut.ranet.jxseMC.service.impl.Activator
```

```
            </bundleActivator>
            <ignorePackage>junit.textui, junit.framework</ignorePackage>
            <exportPackage>
              au.edu.qut.ranet.jxseMC.service
            </exportPackage>
          </osgiManifest>
        </configuration>
      </plugin>
    </plugins>
     <resources>
       <resource>
         <directory>lib</directory>
       </resource>
     </resources>
  </build>
</project>
```

Maven has goals which cover part or whole of the life cycle. For instance, by typing "mvn compile" you can compile the project. Other goals include "package", "install" and "deploy".

The first goal is to make a directory structure as follows:

```
# mvn archetype:create -DgroupId=au.edu.qut.ranet.bluetooth.service \
-DartifactId=BluetoothService -Dpackagename=au.edu.qut.ranet.bluetooth.service
```

At the head of the created directory structure there is now a pom.xml (as per Apache Felix website http://felix.apache.org/site/index.html). This creates a normal jar file by default. This needs to be altered so a "bundle" jar file is created. Alter the pom.xml as per the documentation on the Apache Felix website. Add the source code and the type the command "mvn compile". When the code has been tested using "mvn test" do a "mvn install".

Copy the generated bundle jar file into the unzipped OSGi framework.

After the framework begins it prompts you for the name of the profile you wish to use. Where the name of the profile is not preexisting a new profile is created. Profiles in OSGi record the bundles that are installed and their state. So enter some profile name.

Now "install ¡file¿" where ¡file¿ is the path to the jar file (known in Apache Felix as the bundle) to be installed.

```
# install file:ranet.appMC.nobt-0.1.jar
```

If you type "ps" the bundle should appear with the other existing bundles, showing its status as inactive. Enter the command "start ¡ bundle id number ¿ " and type "ps" again. The bundle status should come up as active.

Appendix I

JXTA

I.1 JXME 2.1.3

```
24/06/07
--------
--------

At stage can search for local ads and remote ads for PeerGroups. Where \
none found can create Group. Found through inspection of code base cache\
is updated upon discovery events being received in a synchronous fashion.\
After peer group done. Look at advertising skills. First skill message.\
Second skill code of hello world. Add pipes for these skills.

26/06/07
--------
--------

It appears we need to pay more attention to ChatDemo and GroupDemo.\
ConfigurationFactory is required and it seems we may need to call\
explicitly those things which push ads to that cache

30/06/07
--------
--------

svn co file:////home/svn/svnreposranet for checkout.

02/07/07
--------
--------

Found that code for JXME does not support RendezvousService \
startRendezvousService. Have sent off log to jxta forum for answer. \
Appear to be in Endpoint propogate. Howevery none of the error message\
come up. Appears to have no value in propogate at 496 of ResolverServiceImpl

if(null != rendezvous) {
    // Walk the message
    rendezvous.walk((Message) queryMsg.clone(), handlerName, outQueName,\
 RendezVousService.DEFAULT_TTL);

    // propagate to local net as well
    rendezvous.propagateToNeighbors(queryMsg, handlerName, outQueName, 2);
} else {
    endpoint.propagate(queryMsg, handlerName, outQueName);
```

```
}


-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running au.edu.qut.ranet.jxme.impl.JXMEImplTest
Jul 2, 2007 12:33:01 AM au.edu.qut.ranet.jxme.impl.JXMEImpl <init>
INFO: about to make new Net Peer Group
Creating a new group advertisement
<FATAL 2007-07-02 00:33:03,896 PeerView::run:2380> Uncaught Throwable in \
thread : PeerView Timer for urn:jxta:uuid-9E992C1CE00345F68981E5A63D04E66A02
java.lang.NullPointerException
   at net.jxta.impl.resolver.ResolverServiceImpl.sendQuery(\
ResolverServiceImpl.java:496)
   at net.jxta.impl.discovery.DiscoveryServiceImpl.getRemoteAdvertisements(\
DiscoveryServiceImpl.java:302)
   at net.jxta.impl.discovery.DiscoveryServiceImpl.getRemoteAdvertisements(\
DiscoveryServiceImpl.java:253)
   at net.jxta.impl.rendezvous.PeerView.discoverRdvAdverisements(\
PeerView.java:708)
   at net.jxta.impl.rendezvous.PeerView.seed(PeerView.java:1937)
   at net.jxta.impl.rendezvous.PeerView.kick(PeerView.java:966)
   at net.jxta.impl.rendezvous.PeerView.access$900(PeerView.java:138)
   at net.jxta.impl.rendezvous.PeerView$KickerTask.run(PeerView.java:2377)
   at java.util.Timer$Scheduler.run(Timer.java:399)
   at java.lang.Thread.run(Thread.java:710)

 Group = BlueSoccerTeam
 Group ID = urn:jxta:uuid-9E992C1CE00345F68981E5A63D04E66A02
Jul 2, 2007 12:33:04 AM au.edu.qut.ranet.jxme.impl.JXMEImpl publishGroup
INFO: Group BlueSoccerTeam published successfully.
Jul 2, 2007 12:33:04 AM au.edu.qut.ranet.jxme.impl.JXMEImpl statusRemoteGroups
INFO: trying to get remote advertisements for peer groups
In JXTA test loop
In JXTA test loop
```

I.2 JXSE 2.3.6

```
11/07/07
--------
--------

...

Looking at 0.91 classpath and JamVM to run JXSE 2.4.1. Problem is generics\
 and enumerations in 1.5
```

I.3 P2PS

```
09/09/07
========

...

Then added all the p2ps and jdom and apache.log4j classes to the classpath. \
Found GNU Classpath did not implement the udp multicast.
```

```
[Linking class p2ps/group/UnknownAuthorizationMechanismException]
[Loaded p2ps/group/AuthorizationException from /mnt/hda/sw/share/classpath]
[Linking class p2ps/group/AuthorizationException]
org.osgi.framework.BundleException: Activator start error.
   at org.apache.felix.framework.Felix._startBundle(Felix.java:1305)
   at org.apache.felix.framework.Felix.startBundle(Felix.java:1201)
   at org.apache.felix.framework.BundleImpl.start(BundleImpl.java:345)
   at org.apache.felix.shell.impl.StartCommandImpl.execute(StartCommandImpl.jav)
   at org.apache.felix.shell.impl.Activator$ShellServiceImpl.executeCommand(Act)
   at org.apache.felix.shell.tui.Activator$ShellTuiRunnable.run(Activator.java:)
   at java.lang.Thread.run(Thread.java:712)
Caused by: java.lang.InternalError: PlainDatagramSocketImpl::joinGroup is not id
   at gnu.java.net.VMPlainDatagramSocketImpl.joinGroup(VMPlainDatagramSocketImp)
   at gnu.java.net.PlainDatagramSocketImpl.joinGroup(PlainDatagramSocketImpl.ja)
   at java.net.MulticastSocket.joinGroup(MulticastSocket.java:411)
   at p2ps.imp.endpoint.UDP.UDPMulticastEndpoint.joinGroup(Unknown Source)
   at p2ps.imp.endpoint.UDP.UDPMulticastEndpoint.<init>(Unknown Source)
   at p2ps.imp.endpoint.UDP.UDPResolver.createDefaultInputEndpoint(Unknown Sour)
   at p2ps.imp.pipe.PipeServiceImp.getEndpoints(Unknown Source)
   at p2ps.imp.pipe.PipeServiceImp.createInputPipe(Unknown Source)
   at p2ps.imp.discovery.DiscoveryServiceImp.createDiscoveryPipe(Unknown Source)
   at p2ps.imp.discovery.DiscoveryServiceImp.init(Unknown Source)
   at p2ps.imp.peer.PeerImp.init(Unknown Source)
   at p2ps.imp.peer.PeerImp.init(Unknown Source)
   at au.edu.qut.ranet.p2ps.impl.PeerClient.<init>(PeerClient.java:33)
   at au.edu.qut.ranet.p2ps.impl.PeerClient.<init>(PeerClient.java:23)
   at au.edu.qut.ranet.p2ps.service.impl.Activator$P2PSServiceImpl.testP2PS(Act)
   at au.edu.qut.ranet.p2ps.service.itest.Activator.start(Activator.java:39)
   at org.apache.felix.framework.util.SecureAction.startActivator(SecureAction.)
   at org.apache.felix.framework.Felix._startBundle(Felix.java:1260)
   ...6 more
java.lang.InternalError: PlainDatagramSocketImpl::joinGroup is not implemented
-> [Loaded p2ps/endpoint/StreamEndpoint from /mnt/hda/sw/share/classpath]
[Linking class p2ps/endpoint/StreamEndpoint]
[Loaded p2ps/imp/endpoint/TCP/TCPEndpoint from /mnt/hda/sw/share/classpath]
[Linking class p2ps/imp/endpoint/TCP/TCPEndpoint]
[Loaded java/net/Socket from /mnt/hda/sw/share/classpath]
[Linking class java/net/Socket]
[Loaded java/io/InterruptedIOException from /mnt/hda/sw/share/classpath]
[Linking class java/io/InterruptedIOException]
[Loaded java/net/SocketTimeoutException from /mnt/hda/sw/share/classpath]
[Linking class java/net/SocketTimeoutException]

...
```

I.4 Using JXTA with Maven and OSGI

To install JXTA within the Apache Felix OSGi framework first download the JXTA 2.5 release candidate version 3 from the JXTA website (www.jxta.org). The jar files making up this release need to be made available to the maven build utility so install the JXTA jar file by entering the following commands.

```
# mvn install:install-file -DartifactId=ranet.jxse.bcprov-jdk14 \
-Dfile=bcprov-jdk14.jar -DgroupId=au.edu.qut.ranet -Dpackaging=jar \
-Dversion=2.5_rc3 -DgeneratePom=true
# mvn install:install-file -DartifactId=ranet.jxse.javax.servlet \
-Dfile=javax.servlet.jar -DgroupId=au.edu.qut.ranet -Dpackaging=jar \
-Dversion=2.5_rc3 -DgeneratePom=true
# mvn install:install-file -DartifactId=ranet.jxse -Dfile=jxta.jar \
-DgroupId=au.edu.qut.ranet -Dpackaging=jar -Dversion=2.5_rc3 \
```

```
-DgeneratePom=true
# mvn install:install-file -DartifactId=ranet.jxse.org.mortbay.jetty \
-Dfile=org.mortbay.jetty.jar -DgroupId=au.edu.qut.ranet -Dpackaging=jar \
-Dversion=2.5_rc3 -DgeneratePom=true
```

For each of the above commands a status message like the following should come up.

```
krobinson@Possibility:~/Desktop/sw/jxta-lib-2.4.1$ mvn install:install-file \
-DartifactId=ranet.jxta.swixml.jar.cots -Dfile=swixml.jar \
-DgroupId=au.edu.qut.ranet -Dpackaging=jar -Dversion=2.4.1 -DgeneratePom=true
[INFO] Scanning for projects...
[INFO] Searching repository for plugin with prefix: 'install'.
[INFO] ------------------------------------------------------------------------\
-------
[INFO] Building Maven Default Project
[INFO]    task-segment: [install:install-file] (aggregator-style)
[INFO] ------------------------------------------------------------------------\
-------
[INFO] [install:install-file]
[INFO] Installing /home/krobinson/Desktop/sw/jxta-lib-2.4.1/swixml.jar to \
/home/krobinson/.m2/repository/au/edu/qut/ranet/ranet.jxta.swixml.jar.cots\
/2.4.1/ranet.jxta.swixml.jar.cots-2.4.1.jar
[INFO] ------------------------------------------------------------------------\
-----
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------\
-----
[INFO] Total time: 1 second
[INFO] Finished at: Wed Jul 04 22:05:40 EST 2007
[INFO] Final Memory: 2M/3M
[INFO] ------------------------------------------------------------------------\
-----
krobinson@Possibility:~/Desktop/sw/jxta-lib-2.4.1$
```

Part of the JXTA functionality that you must have is JXTA broadcasts. JXTA broadcasts require IP Multicast to work on the Bluetooth interface. If multicast is enabled when running the ifconfig command the Bluetooth interface should have the word "MULTICAST" as part of it status message.

```
bnep0     Link encap:Ethernet  HWaddr 00:0A:3A:66:B8:B6
          inet6 addr: fe80::20a:3aff:fe66:b8b6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:37 errors:0 dropped:0 overruns:0 frame:0
          TX packets:71 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:20447 (19.9 KiB)  TX bytes:21342 (20.8 KiB)
```

I.5 Problems Using JXTA on JamVM

These are diary excerpts showing the inability of the JamVM to process the security key used in JXTA because it lacked the necessary encryption algorithm.

```
300807
======
Problem with generation of private key.

INFO: about to make new Net Peer Group
log4j:WARN No appenders could be found for logger (net.jxta.util.ClassFactory).
log4j:WARN Please initialize the log4j system properly.
```

```
java.lang.IllegalStateException: Failed to process private key
   at net.jxta.impl.protocol.PSEConfigAdv.getEncryptedPrivKey(PSEConfigAdv.java\
:434)
   at net.jxta.impl.protocol.PSEConfigAdv.getDocument(PSEConfigAdv.java:612)
   at net.jxta.impl.peergroup.ConfigDialog.saveValues(ConfigDialog.java:2166)
   at net.jxta.impl.peergroup.ConfigDialog.access$700(ConfigDialog.java:130)
   at net.jxta.impl.peergroup.ConfigDialog$3.actionPerformed(ConfigDialog.java\
:1680)
   at java.awt.Button.processActionEvent(Button.java:409)
   at java.awt.Button.processEvent(Button.java:392)
   at java.awt.Button.dispatchEventImpl(Button.java:419)
   at java.awt.Component.dispatchEvent(Component.java:2670)
   at java.awt.EventQueue.dispatchEvent(EventQueue.java:474)
   at java.awt.EventDispatchThread.run(EventDispatchThread.java:85)
org.osgi.framework.BundleException: Activator start error.
   at org.apache.felix.framework.Felix._startBundle(Felix.java:1579)
   at org.apache.felix.framework.Felix.startBundle(Felix.java:1469)
   at org.apache.felix.framework.BundleImpl.start(BundleImpl.java:348)
   at org.apache.felix.shell.impl.StartCommandImpl.execute(StartCommandImpl.java\
:82)
   at org.apache.felix.shell.impl.Activator$ShellServiceImpl.executeCommand(Acti\
vator.java:265)
   at org.apache.felix.shell.tui.Activator$ShellTuiRunnable.run(Activator.java:1\
67)
   at java.lang.Thread.run(Thread.java:740)
Caused by: net.jxta.exception.JxtaError: Canceled during configuration
   at net.jxta.impl.peergroup.ConfigDialog.untilDone(ConfigDialog.java:1741)
   at net.jxta.impl.peergroup.DefaultConfigurator.getPlatformConfig\
(DefaultConfigurator.java:133)
   at net.jxta.impl.peergroup.Platform.initFirst(Platform.java:191)
   at net.jxta.impl.peergroup.GenericPeerGroup.init(GenericPeerGroup.java:901)
   at net.jxta.peergroup.PeerGroupFactory.newPlatform(PeerGroupFactory.java:421)
   at net.jxta.peergroup.PeerGroupFactory.newNetPeerGroup(PeerGroupFactory.java\
:489)
   at au.edu.qut.ranet.jxtaold.impl.JXTAImpl.startJXTA(JXTAImpl.java:99)
   at au.edu.qut.ranet.jxtaold.service.impl.Activator$JXTAServiceImpl.startJXTA\
(Activator.java:107)
   at au.edu.qut.ranet.app.Activator.start(Activator.java:75)
   at org.apache.felix.framework.util.SecureAction.startActivator(SecureAction.\
java:589)
   at org.apache.felix.framework.Felix._startBundle(Felix.java:1535)
   ...6 more
net.jxta.exception.JxtaError: Canceled during configuration
->

01/09/08 [sic]
========

...

[Loaded net/jxta/endpoint/EndpointAddress$UnmodifiableEndpointAddress]
[Linking class net/jxta/impl/membership/pse/PSESecurityEngineFactory]
[Loaded net/jxta/impl/membership/pse/PSESecurityEngineFactory]
[Linking class net/jxta/impl/membership/pse/PSESecurityEngineFactory\
$PSESecurityEngineDefaultFactory]
[Loaded net/jxta/impl/membership/pse/PSESecurityEngineFactory\
$PSESecurityEngineDefaultFactory]
[Linking class net/jxta/impl/membership/pse/PSEPeerSecurityEngine]
[Loaded net/jxta/impl/membership/pse/PSEPeerSecurityEngine]
[Linking class net/jxta/impl/membership/pse/PSESecurityEngineFactory\
$PSEPeerSecurityEngineDefault]
[Loaded net/jxta/impl/membership/pse/PSESecurityEngineFactory\
```

```
$PSEPeerSecurityEngineDefault]
[Linking class net/jxta/impl/membership/pse/PSEAuthenticatorEngineFactory]
[Loaded net/jxta/impl/membership/pse/PSEAuthenticatorEngineFactory]
[Linking class net/jxta/impl/membership/pse/PSEAuthenticatorEngineFactory\
$PSEAuthenticatorEngineDefaultFactory]
[Loaded net/jxta/impl/membership/pse/PSEAuthenticatorEngineFactory\
$PSEAuthenticatorEngineDefaultFactory]
[Linking class net/jxta/impl/membership/pse/PSEKeyStoreManagerFactory]
[Loaded net/jxta/impl/membership/pse/PSEKeyStoreManagerFactory]
[Linking class net/jxta/impl/membership/pse/PSEKeyStoreManagerFactory\
$PSEKeyStoreManagerFactoryDefault]
[Loaded net/jxta/impl/membership/pse/PSEKeyStoreManagerFactory\
$PSEKeyStoreManagerFactoryDefault]
[Linking class net/jxta/impl/membership/pse/KeyStoreManager]
[Loaded net/jxta/impl/membership/pse/KeyStoreManager]
[Linking class net/jxta/impl/membership/pse/CMKeyStoreManager]
[Loaded net/jxta/impl/membership/pse/CMKeyStoreManager]
[Loaded java/security/KeyStore from /usr/share/classpath/glibj.zip]
[Linking class java/security/KeyStore]
[Loaded java/security/GeneralSecurityException from /usr/share/classpath/\
glibj.zip]
[Linking class java/security/GeneralSecurityException]
[Loaded java/security/NoSuchAlgorithmException from /usr/share/classpath/\
glibj.zip]
[Linking class java/security/NoSuchAlgorithmException]
[Loaded java/security/KeyStoreException from /usr/share/classpath/glibj.zip]
[Linking class java/security/KeyStoreException]
[Loaded java/security/NoSuchProviderException from /usr/share/classpath/\
glibj.zip]
[Linking class java/security/NoSuchProviderException]
[Linking class net/jxta/pipe/PipeService]
[Loaded net/jxta/pipe/PipeService]
[Linking class net/jxta/impl/pipe/PipeResolver$Listener]
[Loaded net/jxta/impl/pipe/PipeResolver$Listener]
[Linking class net/jxta/impl/pipe/PipeServiceImpl]
[Linking class net/jxta/exception/ServiceNotFoundException]
[Loaded net/jxta/exception/ServiceNotFoundException]
[Loaded java/util/Hashtable$3 from /usr/share/classpath/glibj.zip]
[Linking class java/util/Hashtable$3]
[Loaded java/util/Hashtable$ValueIterator from /usr/share/classpath/glibj.zip]
[Linking class java/util/Hashtable$ValueIterator]
[Loaded java/lang/ClassCastException from /usr/share/classpath/glibj.zip]
[Linking class java/lang/ClassCastException]
[Linking class net/jxta/impl/endpoint/QuotaIncomingMessageListener\
$MessageFromSource]
[Loaded net/jxta/impl/endpoint/QuotaIncomingMessageListener\
$MessageFromSource]
fatal error : group creation failure
[Loaded java/lang/Throwable$StaticData from /usr/share/classpath/glibj.zip]
[Linking class java/lang/Throwable$StaticData]
net.jxta.exception.PeerGroupException: Could not load group implementation
   at net.jxta.impl.peergroup.GenericPeerGroup.newGroup(GenericPeerGroup.java\
:1361)
   at net.jxta.impl.peergroup.PeerGroupInterface.newGroup(PeerGroupInterface\
.java:327)
   at net.jxta.peergroup.PeerGroupFactory.newNetPeerGroup(PeerGroupFactory\
.java:452)
   at net.jxta.peergroup.PeerGroupFactory.newNetPeerGroup(PeerGroupFactory\
.java:491)
   at au.edu.qut.ranet.jxtaold.impl.JXTAImpl.startJXTA(JXTAImpl.java:99)
   at au.edu.qut.ranet.jxtaold.service.impl.Activator$JXTAServiceImpl.startJXTA\
(Activator.java:107)
```

```
   at au.edu.qut.ranet.app.Activator.start(Activator.java:54)
   at org.apache.felix.framework.util.SecureAction.startActivator(SecureAction\
.java:589)
   at org.apache.felix.framework.Felix._startBundle(Felix.java:1535)
   at org.apache.felix.framework.Felix.startBundle(Felix.java:1469)
   at org.apache.felix.framework.Felix.setFrameworkStartLevel(Felix.java:1064)
   at org.apache.felix.framework.StartLevelImpl.run(StartLevelImpl.java:258)
   at java.lang.Thread.run(Thread.java:740)
Caused by: net.jxta.exception.PeerGroupException: Missing peer group service
   at net.jxta.impl.peergroup.StdPeerGroup.initFirst(StdPeerGroup.java:583)
   at net.jxta.impl.peergroup.ShadowPeerGroup.initFirst(ShadowPeerGroup.java:85)
   at net.jxta.impl.peergroup.GenericPeerGroup.init(GenericPeerGroup.java:901)
   at net.jxta.impl.peergroup.GenericPeerGroup.loadModule(GenericPeerGroup.java\
:719)
   at net.jxta.impl.peergroup.GenericPeerGroup.newGroup(GenericPeerGroup.java\
:1355)
   ...12 more
Caused by: net.jxta.exception.ServiceNotFoundException: urn:jxta\
:uuid-DEADBEEFDEAFBABAFEEDBABE0000000505
   at net.jxta.impl.peergroup.GenericPeerGroup.lookupService(GenericPeerGroup\
.java:519)
   at net.jxta.impl.peergroup.GenericPeerGroup.checkServices(GenericPeerGroup\
.java:561)
   at net.jxta.impl.peergroup.StdPeerGroup.initFirst(StdPeerGroup.java:580)
   ...16 more
root@krobinson-desktop:~/Desktop/work-tfr-area/vers1_0main/felix-1.0.0#
```

Appendix J

RANET Documents

You may wish to have a detailed view of the RANET code. With the electronic copy of this document is a zipped up "html" directory. Within that directory is an "index.html" file. Open that up in your favourite browser.

Bibliography

[1] A.M. Andersen and T. Torabi. A holistic framework for mobile environments. In *Software Engineering Conference, 2006. Australian*, pages 18– 21, April 2006.

[2] Frank-Uwe Andersen, Luca Caviglione, and Oliver Waldhorst. Overall research directions and problem classification (in preparation of a "problem statement" draft). World Wide Web - University of Maryland Computer Science website, 2007. URL http://www.cs.umd.edu/projects/p2prg/wireless-research.txt accessed on 04/12/07.

[3] Joe Armstrong. *Programming Erlang, Software for a Concurrent World*. The Pragmatic Programmers. Pragmatic Bookshelf, Raleigh, North Carolina, first edition, 2007.

[4] Mia Backlund Norberg and Terese Taaveniku. A web service architecture in mobile ad hoc networks. Master's thesis, Computer Science and Electrical Engineering / Computer Communication, Lulea University of Technology, Lulea University of Technology, 971 87 Lulea, Sweden, 2005. url http://epubl.ltu.se/1402-1617/2005/141/index-en.html accessed on 27/11/2007.

[5] James Bailey. Sensor model language. In *Sensor Networks*, Melbourne, Australia, October 2006. National ICT Australia.

[6] Nick Baker. Zigbee and bluetooth - strengths and weaknesses for industrial applications. *IEE Computing and Control Engineering*, pages 21–25, April/May 2005.

[7] Balazs Bakos, Gergely Csucs, Lorant Farkas, and Jukka K. Nurminen. Peer-to-peer protocol evaluation in topologies resembling wireless networks. an experiment with gnutella query engine. World Wide Web, 2002. Paper published by Nokia Research Centre P.O. Box 392, H-1461 Budapest Hungary, Budapest University of Technology and Economics P.O. Box 91 H-1521 Budapest Hungary and Nokia Research Centre P.O. Box 407 FIN-00045 Nokia Group Finland.

[8] S. Baset, H. Schulzrinne, and M. Matuszewski. Peer-to-peer protocol (p2pp) draft-baset-p2psip-p2pp-01.

[9] S. Bergbreiter and K.S.J. Pister. Cotsbots: an off-the-shelf platform for distributed robotics. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1632– 1637, October 2003.

[10] M. Beutel, J.and Dyer, L. Meier, and L. Thiele. Scalable topology control for deployment-support networks. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, April 2005.

[11] M Bisignano, G. Di Modica, and O. Tomarchio. Jmobipeer: a middleware for mobile peer-to-peer computing in manets. *Distributed Computing Systems Workshops, 2005. 25th IEEE International Conference on*, pages 785–791, 6-10 June 2005.

[12] Andrew Black, Stéphane Ducasse, Oscar Nierstrasz, and Damien Pollet. *Squeak by Example*. Square Bracket Associates, September 2007. Chapter 1 - A quick tour of squeak. Also has contributions by Damien Cassou and Marcus Denker. Online at http://SqueakByExample.org/index.html.

[13] Francis Brosnan Blazquez. Beep: The blocks extensible exchange protocol. World Wide Web - BEEP website, August 2007. http://www.beepcore.org accessed on 24/11/2007.

[14] BlueZ. Howto set up common pan scenarios with bluez's integrated pan support. World Wide Web - BlueZ website. http://bluez.sourceforge.net/contrib/HOWTO-PAN accessed on 09/08/2006.

[15] D. Bryan, P. Matthews, A. E. Shim, and D. Willis. Concepts and terminology for peer to peer sip draft-ietf-p2psip-concepts-01. World Wide Web - IETF website, November 2007. http://tools.ietf.org/html/draft-ietf-p2psip-concepts-01 accessed on 28/11/2007.

[16] Products, bug labs. World Wide Web - Bug Labs website, November 2007. http://www.buglabs.net/products accessed on 06/11/2007.

[17] Lennert Buytenhek. brctl. man page part of Linux distribution, November 2001. L. Buytenhek is the author of the brctl utility. He can be reached at buytenh@gnu.org.

[18] The gnutella protocol specification v0.4. World Wide Web - Clip 2 Website, July 2004.

[19] Bram Cohen. The bittorrent protocol specification. World Wide Web - Bittorrent website, 2008.

[20] Douglas E. Comer. *Internetworking with TCP/IP - Principles, Protocols and Architectures*. Prentice-Hall International, 2000.

[21] M. Connolly. Analysis of udp performance over bluetooth. In *Information Technology and Telecommunications 2003*, 2003.

[22] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G.S. Sukhatme. Robomote: enabling mobility in sensor networks. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 404 – 409, April 2005.

[23] Bruce Powel Douglass. *Doing hard time: developing real-time systems with UML, objects, frameworks, and patterns*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[24] Cameron Ross Dunne. Using mobile agents for network resource discovery in peer-to-peer networks. *SIGecom Exchanges - Newsletter of the ACM Special Interest Group on E-Commerce*, 2.3:1–9, 2001.

[25] The RoboCup Federation. Home page. World Wide Web - RoboCup website, 2005. http://www.robocup.org/02.html accessed on 20/02/2005.

[26] Home page. World Wide Web - Federation of International Robot-soccer Association website, 2005. http://www.fira.net/index.html accessed on 20/02/2005.

[27] Endless possibilities. Published on the gumstix website. http://gumstix.com/index.html accessed on 13/12/2007.

[28] Chen Hongyuan, T.V.L.N. Sivakumar, Leping Huang, and T. Kashima. Topology-controllable scatternet formation method and its implementation. In *Wireless Ad-Hoc Networks, 2004 International Workshop on*, pages 243– 248, June 2004.

[29] Brian Hrolenok, Jason Youngblood, and other. flockbots. World Wide Web - FlockBots website, 2005. http://cs.gmu.edu/ eclab/projects/robots/flockbots/pmwiki.php accessed on 11/09/2006.

[30] P. Ibach, N. Milanovic, J. Richling, V. Stantchev, A. Wiesner, and M. Malek. Cero: Ce robots community. *Software, IEE Proceedings*, 152(5):210–214, January 2005.

[31] Infrared data association. World Wide Web - Infrared Data Association, February 2009. URL http://www.irda.org/ accessed on 16/02/2009.

[32] Jadabs-cldc goals. World Wide Web - Swiss Federal Institute of Technology (ETH) Zurich - Information and Communication Systems Research Group - berliOS website, 10 November 2004. URL http://jadabs.berlios.de/jadabs-cldc accessed on 10/01/2005.

[33] Java client. World Wide Web - Java Client website, April 2008. URL http://javaplayer.sourceforge.net/index.php accessed on 25/04/2008.

[34] Jehn-Ruey Jiang, Bing-Rong Lin, and Yu-Chee Tseng. Analysis of bluetooth device discovery and some speedup mechanisms. World Wide Web - Home page of Yu-Chee Tseng at NCTU, Department of Computer Science, Taiwan, 2004. URL http://www.cs.nctu.edu.tw/ yctseng/papers.pub/mobile46-IJEE.pdf accessed on 22/02/2009.

[35] T.E. Jonvik, P. Engelstad, and D. van Tanh. Ad-hoc formation of bluetooth piconet and ip allocation in pan. In *Wireless Personal Multimedia Communications, 2002. The 5th International Symposium on*, volume 2, pages 489– 493, 2002.

[36] JXTA. Jxta java tm standard edition v2.5: Programmers guide. World Wide Web - JXTA website, September 2007.

[37] The jxta community. Published on the JXTA website, 2008. URL https://jxta.dev.java.net accessed on 13/01/2008.

[38] D. Kammer, G. McNutt, and B. Senese. *Bluetooth Application Developers Guide: The Short Range Interconnect Solution.* Syngress Publishing, Inc., 800 Hingham Street, Rockland, MA 02370, United States of America, first edition, 2002.

[39] Jorunn Kassin. Next generation of mobile terminals. Master's thesis, Faculty of Information Technology, Mathematics and Electrical Engineering, Department of Telematics, Norwegian University of Science and Technology, Norway, July 2002.

[40] knopflerfish. World Wide Web - Knopflerfish open source OSGi website, September 2006. URL http://www.knopflerfish.org/ accessed on 15/09/2006.

[41] Michael Koch. Jamvm. man page part of Linux distribution, September 2006. M. Koch is the author of the JamVM man page. He can be reached at konqueror@gmx.de.

[42] Tetsuo Kotoku. Openrtm-aist: A reference implementation of the robotic technology component specification. World Wide Web - Staff section of National Institute of Advanced Industrial Science and Technology (AIST), June 2007. URL http://staff.aist.go.jp/t.kotoku/omg/2007Brussels/robotics2007-06-10.pdf accessed on 12/11/2007.

[43] Korebot light edition board. Published on the K-Team website. URL http://www.k-team.com accessed on 13/12/2007.

[44] M.B. McMickell, B. Goodwine, and L.A. Montestruque. Micabot: a robotic platform for larg-scale distributed robotics. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 2, pages 1600–1605, September 2003.

[45] Boris Mejias and Kevin Glynn. A service-based architecture for deploying robust peer-to-peer applications. In *Self-Managed Networks, Systems, and Services (SELFMAN) 2006*, Dublin, Ireland, 2006. IEEE Computer Society's Task Force on Autonomous and Autonomic Systems (TFAAS). Slide presentation on P2PS and P2PKit presented at conference.

[46] Microsoft office communicator. World Wide Web - Microsoft web page, 2008.

[47] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing - technical report (2002). World Wide Web - Hewlett Packard Website, 2002.

[48] Dejan S. Milojicic, Vana Kalogeraki, Rajan Lukose, Kiran Nagaraja, Jim Pruyne, Bruno Richard, Sami Rollins, and Zhichen Xu. Peer-to-peer computing. Technical Report HPL-2002-57R1, HP Laboratories Palo Alto, California, United States of America, 2003. www.hpl.hp.com/techreports/2002/HPL-2002-57R1.pdf.

[49] J. Misic and V.B. Misic. Modeling bluetooth piconet performance. *Communications Letters, IEEE*, 7(1):18– 20, January 2003.

[50] P. Mohapatra, C. Gui, and J. Li. Group communications in mobile ad hoc networks. *Computer*, February 2004.

[51] R. Morrow. *Bluetooth Operation and Use*. McGraw-Hill, Two Penn Plaza, New York, NY 10121-2298, first edition, 2002.

[52] NICTA. Introduction. In *Sensor Networks*, Melbourne, Australia, October 2006. National ICT Australia.

[53] Faqs. Published on the OSGi organization website FAQS section, January 2005. URL http://www.osgi.org/about/faqs.asp?section= accessed on 10/01/2005.

[54] B. Patil, Y. Saifullah, S. Faccin, S. Sreemanthula, L. Aravamudhan, S. Sharma, and R. Mononen. *IP in Wireless Networks*. Prentice Hall Professional Technical Reference, 2003.

[55] pickle – python object serialization. World Wide Web - Python website, September 2006. http://docs.python.org/lib/module-pickle.html accessed on 15/01/2008.

[56] Himanshu Raj, Balasubramanian Seshasayee, Keith J. O'Hara, Ripal Nathuji, Karsten Schwan, and Tucker Balch. Spirits: Using virtualization and pervasiveness to manage mobile robot software systems. In *Self-Managed Networks, Systems, and Services (SELFMAN) 2006*, pages 116–129, Dublin, Ireland, 2006. IEEE Computer Society's Task Force on Autonomous and Autonomic Systems (TFAAS). http://www.springerlink.com/content/mw63815701312834 accessed on 05/12/2007.

[57] Kenneth Patrick Robinson and Joaquin Sitte. Wireless robotic ad-hoc network using commercial off the shelf (cots) products. In Ulrich Rückert, Joaquin Sitte, and Ulf Witkowski, editors, *Proceedings of the 4th International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE 2007)*, pages 181–188, Buenos Aires, October 2007.

[58] Robo markup language home page. Web page on Roboml Website, 2003. URL http://www.roboml.org/roboml.html accessed on 18/11/2007.

[59] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, November 2001.

[60] Roman Schmidt. P-grid system overview. World Wide Web - P-Grid Website, March 2007. http://www.p-grid.org/implementation/P-Grid Overview.pdf accessed on 25/11/2007.

[61] Peter Seibel. *Practical Common Lisp*. Apress, 2855 Telegraph Avenue, Suite 600 Berkeley, CA 94705, first edition, 2005.

[62] Russell Selwyn. Wireless peer to peer. World Wide Web - QUT OLT Website, 2004. Lecture TBD in subject ITN671 Wireless Networks.

[63] Gadi Shor. How bluetooth, uwb, and 802.11 stack up on power consumption. World Wide Web - Digital Home Design Line, April 2008. URL http://www.digitalhomedesignline.com accessed on 24/02/2009.

[64] Raul Siles. Wireless forensics: Tapping the air - part one. World Wide Web - Security Focus website, January 2007. URL http://www.securityfocus.com/infocus/1884 accessed on 24/02/2009.

[65] J. Sitte and P. Winzer. Mastering complexity in robot design. *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, 2:1815–1819 vol.2, 28 Sept.-2 Oct. 2004.

[66] Michael Somby. A review of robotics software platforms. World Wide Web - LinuxDevices.com, November 2007.

[67] Jtrs. World Wide Web - Space and Naval Warfare System Command website, September 2006. URL http://enterprise.spawar.navy.mil/index.cfm accessed on 15/09/2006.

[68] Lloyd Spencer. Robotics and the need for standards. World Wide Web - Robotic Trends website, May 2005.

[69] W.R. Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*, volume 1 of *Addison-Wesley Computing Series*. Addison-Wesley Publishing Company, One Jacob Way, Reading, Massachusetts, United States of America, first edition, 1994.

[70] Swarm-bots. Swarm-bots homepage. World Wide Web - Swarm-bots website, 2006. URL http://www.swarm-bots.org accessed on 14/09/2006.

[71] TinyOS. Tinyos homepage. World Wide Web - TinyOS website, 2006. URL http://www.tinyos.net/special/mission accessed on 29/08/2006.

[72] Project Jxta v2.0. Project jxta v2.0: Java programmer's guide. World Wide Web - Project JXTA website, 2003.

[73] Ashlee Vance. The call for robotics standards is on. World Wide Web - The Register website, June 2006. URL http://www.theregister.co.uk/2006/06/22/robotics standards accessed on 11/12/2007.

[74] H. Velayos and G. Karlsson. Techniques to reduce the ieee 802.11b handoff time. *Communications, 2004 IEEE International Conference on*, 7:3844–3848 Vol.7, June 2004.

[75] Ian Wang. P2PS (Peer-to-Peer Simplified). In *Proceedings of 13th Annual Mardi Gras Conference - Frontiers of Grid Applications and Technologies*, pages 54–59. Louisiana State University, February 2005.

[76] Wikipedia. Robot software. World Wide Web - Wikipedia website, November 2007. URL http://en.wikipedia.org/wiki/Usenet accessed on 12/11/2007.

[77] Xmpp standards foundation. World Wide Web - XMPP website, 2007. URL http://www.xmpp.org accessed on 27/11/2007.