Middleware Solutions for Self-organizing Multi-hop Multi-path Internet Connectivity Based on Bluetooth

Paolo Bellavista
Carlo Giannelli

DEIS, Università degli Studi di Bologna, Viale Risorgimento, 2 - 40136 Bologna Italy
carlo.giannelli@unibo.it
From traditional homogeneous to novel heterogeneous wireless scenarios
- several communication technologies
- infrastructure and peer points of access

Multi-hop Multi-path Heterogeneous Connectivity (MMHC) middleware for context-aware dynamic connectivity in heterogeneous environments
- context information related to path reliability and throughput
- manage interface, platform, driver heterogeneity
- efficiency required to gather context information, evaluate connectivity opportunities, and perform connections
The Wireless Scenario

- **Client node**: node requiring connectivity, e.g., user PDA
- **Connectors**: nodes providing connectivity, e.g., UMTS base station
- **Channel**: active client-connector IP connection, e.g., Bluetooth pairing and DHCP configuration

- **Handover procedure**
  - a client node changes current connector while moving
- **Evaluation process**
  - **context gathering**: which information is important?
  - **metric application**: which is the most suitable connector?
**Homogeneous Wireless Scenario**

- **One communication interface at a time**
  - the client node does not change wireless interface

- **Horizontal handover**
  - infrastructure connectors only
  - origin and destination connectors based on the same wireless technology

- **IEEE 802.11**
  - connectors are IEEE 802.11 Access Points (APs)
  - metric based on Received Signal Strength Indication (RSSI) and Signal to Noise Ratio (SNR), usually embedded in interface firmware
Heterogeneous Wireless Scenario

- **Heterogeneous interfaces**
  - the client node exploits **multiple wireless** interfaces, even simultaneously

- **Heterogeneous connectors**
  - **infrastructure** or **peer** nodes
  - fixed or **mobile** peers
  - single-/**multi-hop paths**

- **Connectivity management**
  - managing interfaces/connectors/channels/paths considering **several context data** to take advantage of the **many networking opportunities**

- **Wireless heterogeneity increases** client node **capabilities**:
  - heterogeneous connectors enable the **most suitable** form of connectivity
    - Bluetooth to limit power consumption, IEEE 802.11 to get larger bandwidth
    - peer connectors **extend connectivity** opportunities via multi-hop **paths**
      - UMTS link accessed via Bluetooth through a peer connector
MMHC: Multi-hop Multi-path Heterogeneous Connectivity

- Novel metric considering a wide set of information at different abstraction levels
  - traditional RSSI/SNR based evaluation processes are not enough

- Evaluation metric specifically designed for heterogeneous wireless scenarios
  - client node and peer mobility (based on RSSI) to provide reliability
  - wireless technology and path characteristics, e.g., bandwidth and number of clients at each hop, to provide sufficient throughput
  - residual battery level to ensure path long-term durability
MMHC Architecture

- **Network Interface Provider**
  - homogeneous access to heterogeneous interfaces on different operating systems

- **Connector Manager**
  - single-hop connections based on node mobility

- **Routing Manager**
  - context information remote distribution
  - multi-hop paths based on estimated connectivity availability and throughput
Network Interface Provider (NIP) provides a homogeneous access to heterogeneous interfaces.

**Features**: set of capabilities common to interfaces
- **get available connectors**, to get available connectors list and related information such as RSSI
- **perform as peer connector**, to offer connectivity in a peer-to-peer fashion
- **connect to a connector**, to perform a connection with a given connector

**IEEE 802.11 standard**
- **scan** of available ESSID (Extended Service Set ID)
- connectivity provisioning via **IBSS** (Independent Basic Service Set) (Independent Basic Service Set)
- **association** to a given AP
Connector Manager

- Connector Manager (CM) establishes **single-hop channels** with remote devices
  1) connectors discovery via **any interface**
  2) **connectors evaluation** based on **mobility degree**
  3) requires **layer2 connections** with most suitable connector of each interface
  4) activates layer3 configuration via **DHCP client**

- CM does not interact with remote nodes
  - mobility degree achieved **locally** in a lightweight manner
  - requirements in terms of maximum node-connector mutual mobility
Routing Manager

- Routing Manager (RM) handles **multi-hop paths**
  1) **context information exchange** with one-hop distant nodes
  2) single-hop distant **links evaluation**
  3) **routing rule modification** to provide suitable multi-hop paths

- RM has a **wider perspective**
  - mobility, throughput and energy of paths
  - **discard unreliable paths** due to mobility, then achieve a trade-off among **throughput and energy**
Performance Considerations (1)

- Operating system and wireless interface independent tasks
  - little impact on achieved performance
  - e.g., CM spends about 120ms for the dynamic evaluation of 5 connectors

- Operating system dependent tasks
  - different implementations on different platforms
  - e.g., RM performs multi-hop path creation via `iptables` on Linux and `route` on MS Windows
Wireless interface dependent tasks

- different implementations on different platforms and different drivers
  - IEEE 802.11: Wireless Extensions on Linux, NDIS drivers on Windows XP/Vista
  - Bluetooth: BlueZ on Linux, Widcomm, BlueSoleil and Microsoft Bluetooth Stack on Windows XP/Vista

- great impact on performance
  - IEEE 802.11 AP scan and association last about 1-3s, in relation to underlying devices
  - Bluetooth inquiry procedure and pairing last more than 12s
Bluetooth usually exploited only to connect remote devices, e.g., wireless keyboard/mouse
- manual connection via platform specific user interfaces
- relatively long pairing procedure
- limited coverage range

Bluetooth suitable also for **connectivity provisioning**
- much **lower power consumption** than IEEE 802.11
  - 1-35 mA instead of 100-350 mA
- **sufficient bandwidth** for many applications
  - 0.7 Mbps for Bluetooth 1.2, 3 Mbps for Bluetooth 2.0 EDR
NIP and Bluetooth

- NIP implementation for Bluetooth
  - get available connectors, **inquiry procedure** to discover Bluetooth devices
  - perform as peer connector, **PAN** (Personal Area Network) provisioning and **DHCP server** instantiation
  - connect to a connector, **PAN** connection and **DHCP client**

- Bluetooth-specific issues:
  - manage driver **heterogeneity**
  - increase inquiry **efficiency**

- **JSR-82** exploitation to provide a **platform independent efficient implementation** of NIP features for Bluetooth devices
Pros
- multi-platform standard
- applications can reduce the inquiry period

Cons
- do not provide RSSI values, required to evaluate connectors mobility
- do not support BNEP (Bluetooth Networking Encapsulation Protocol), required to easily provide IP-based PAN connectivity

Note: NIP and JSR-82 act at different layers
- NIP provides homogeneous access to heterogeneous interfaces, e.g., IEEE 802.11 and Bluetooth
- JSR-82 provides homogeneous access to heterogeneous drivers for Bluetooth, e.g., BlueZ and MS Bluetooth Stack
Adaptable Inquiry Procedure

- **JSR-82 DiscoveryAgent class**: `startInquiry` and `cancelInquiry` methods
  - full inquiry procedure of 10.24s discovers 100% devices
  - halved inquiry procedure of 5.12s discovers **99% devices** [Peterson et al.]

- **Connector Manager (CM)** modifies inquiry procedure length in relation to the current context
  - short inquiry at system startup and at connectivity disruption
  - full inquiry otherwise
Bluetooth provides RSSI only of connected devices
1) inquiry procedure to discover devices
2) baseband connection
3) RSSI gathering

Linux BlueZ
- hcitool -i hciX cc remote_addr
- hcitool -i hciX rssi remote_addr

Windows Widcomm
- BOND RETURN_CODE Bond(BD_ADDR bda, BT_CHAR *pin_code)
- BOOL GetConnectionStats(BD_ADDR bda, tBT_CONN_STATS *p_conn_stats)
additional BNEPConnector class: server() method
- setup a Group Ad-hoc Network (GN) PAN: a device behaves as gateway, the others as clients
- instantiate a DHCP server: the node acts as gateway

Linux BlueZ
- `pand -i hciX --listen --role GN --devup ./devup.sh --master`
- `devup.sh` instantiates the DHCP server at connection establishment, as soon as bnepx interface is available

Windows Widcomm
- no PAN setup, no DHCP server command (available on MS Windows Server 2008)
JSR-82 Extension for Connectivity Provisioning (2)

1) NIP exploits BNEPConnector.client(remote_addr) to connect to a GN PAN as PANU (PAN User)
   - Linux BlueZ: pand -i hciX --connect remote_addr
     --role PANU --service GN
   - Windows Widcomm: CreateConnection(remote_addr, SERVCLASS_GN)

2) Then NIP activates a DHCP client
   - Linux: dhclient on bnepX interface
   - Windows: ipconfig /renew *
Conclusions & Ongoing Work

- MMHC supports **multi-hop multi-path** spontaneous connectivity exploiting off-the-shelf **heterogeneous equipment**
  - IEEE 802.11, Bluetooth, Ethernet
- Bluetooth effective exploitation via
  - **efficient inquiry** procedure to discover devices
  - **homogeneous access** to heterogeneous drivers
- **JSR-82 enhancement** to
  - gather RSSI, required to estimate node mobility
  - provide BNEP connections, to easily support IP connectivity

- Ongoing work
  - additional efforts to fully support MS Windows drivers
  - **QoS** issues: multi-hop connectivity **starvation avoidance** via dynamic and context-aware bandwidth reservation
  - **security** issues: peer mutual authentication, user incentives, dynamic level of trust management
Any question?

- Prototype code and implementation insights:
  - http://lia.deis.unibo.it/research/MAC/
  - http://lia.deis.unibo.it/research/MACHINE/
  - http://lia.deis.unibo.it/research/MMHC/
  - http://lia.deis.unibo.it/Staff/CarloGiannelli/