A Mechanism for Host Mobility Management supporting Application Awareness

Arjan Peddemors, Hans Zandbelt, Mortaza Bargh

Telematica Instituut

Enschede, The Netherlands

MobiSys'04, June 9, 2004

Telematica Instituut

Introduction and Motivation

- Ability to adapt to changing resources is broadly recognized as an important aspect for mobile applications
- Variation in available network resources important to *connected* mobile applications
- Cause of variations in available network resources
  - Mobile host typically connects to *wireless* networks: more dynamic characteristics than their fixed counterparts
  - Mobile host dynamically connects to multiple access networks at the same time: even becoming the case for mass market devices such as “Smart Phones”
  - Increasing heterogeneity in terms of wireless network technologies

Introduction and Motivation

- Dynamic availability of access networks requires handovers to sustain current end-to-end connections / sessions
- Fortunately, various protocols and approaches exist that support connection handover (“seamless mobility”), each with own benefits
- Mobile host may offer multiple of these mobility management facilities and allow mobile applications to choose between them
  - Applications need be aware of mobility process
- Mobility Process (definition)
  - Process of mobile host network activation and deactivation, and updates in the state of mobility management components to adjust to new network situation

Objectives

Define a system-level mobility management mechanism for mobile hosts in a heterogeneous network environment which

- Provides means to *inform applications* on the mobile host about the state and events of the *mobility process* and characteristics of *available networks*
- Cooperates and interacts with *existing protocols and mechanisms* that accommodate *node mobility* within the Internet

Outline

- Application Classification
- Guiding Principles
- Mechanism Architecture
- Implementation
- Experiments
- Evaluation and Discussion
- Conclusions and Future Work

Application Classification

- Two characteristics used for application classification
  - Whether or not the application is aware of the mobility process
  - Whether or not the application itself is dealing with the management of its connections and sessions in the event of changing connected networks
- Three types of applications

	Aware of mobility process	Managing own connection / sessions
Type I application	no	no
Type II application	yes	no
Type III application	yes	yes

Guiding Principles

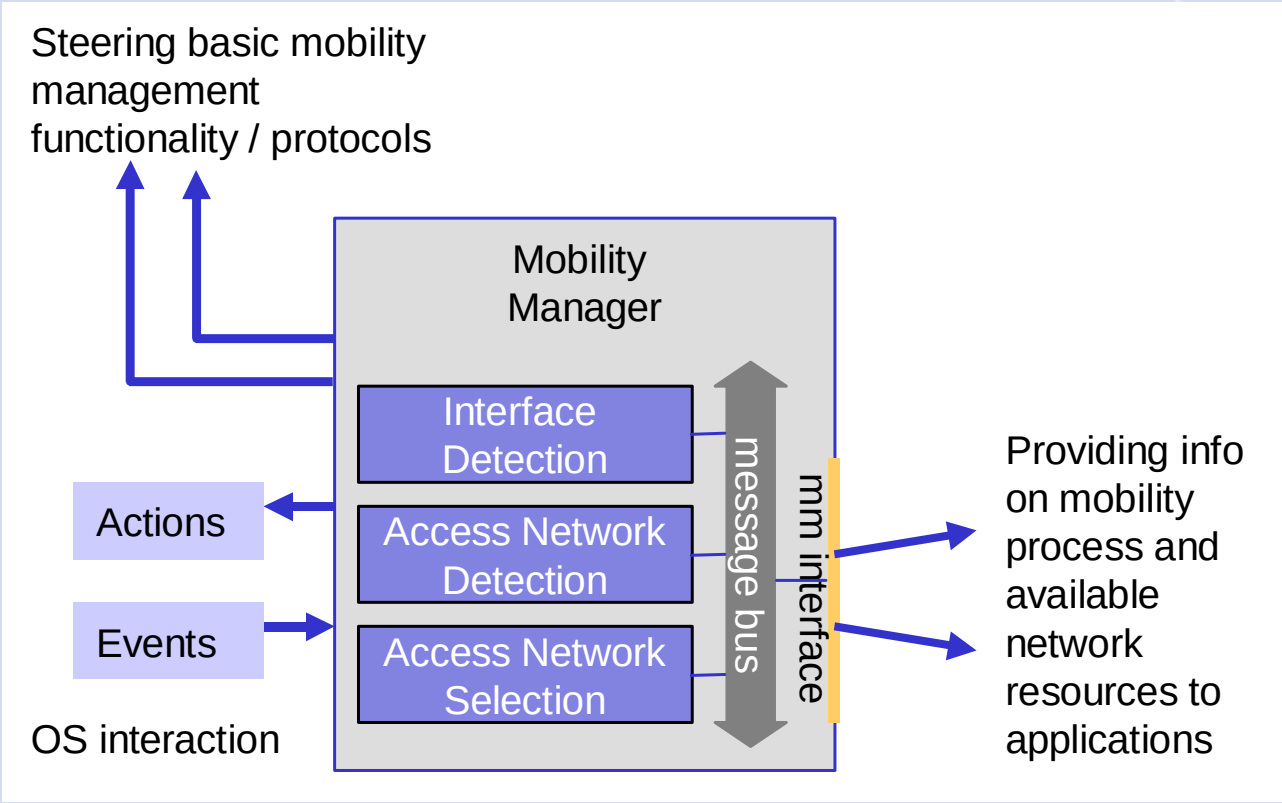
- Control is at the end-point
  - I.e. network entities do not influence the decision on access network selection
- Where possible, allow applications to keep using common communication libraries (Sockets API)
- Assuming best-effort IP-based access networks (so, not considering QoS conservation aspects in handover situation)
- No focus on real-time aspects
  - I.e. fast handovers
- No explicit attention to power management

Mobility Manager

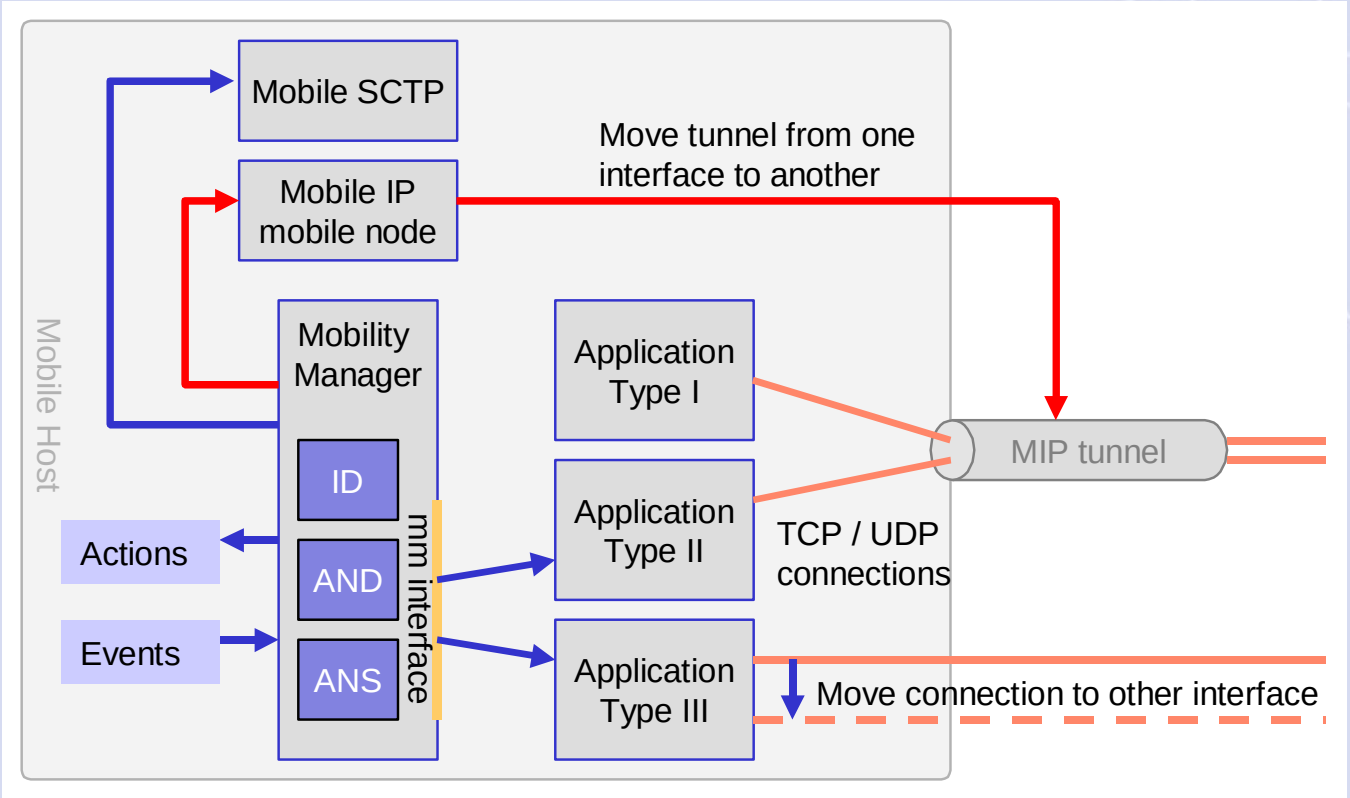
- Mobility Manager: main architectural component
  - System level service residing on the mobile host as background process
  - Responsible for the management of the host's network mobility
- Listens for events from OS about
  - Changes in network interfaces
  - Availability of network scan results
- Takes actions to
  - Set IP configuration for network interfaces
  - Manipulate the host's routing table
  - Initiate scanning on wireless network interfaces
  - Control network and transport layer mobility management components

Mobility Manager Components

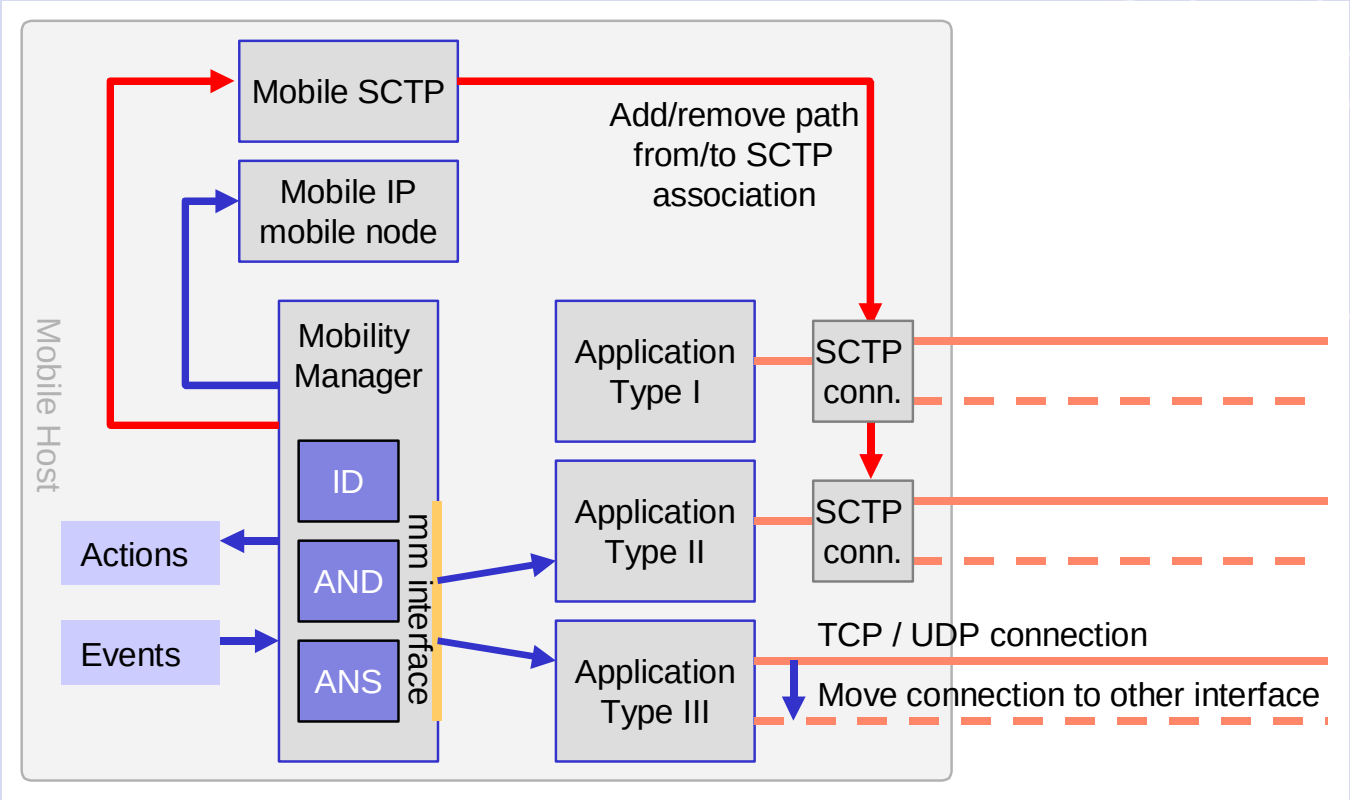
- Communication between components
  - Message based
  - Every component defines own set of outgoing and incoming messages



Mobility Manager with Mobile IP

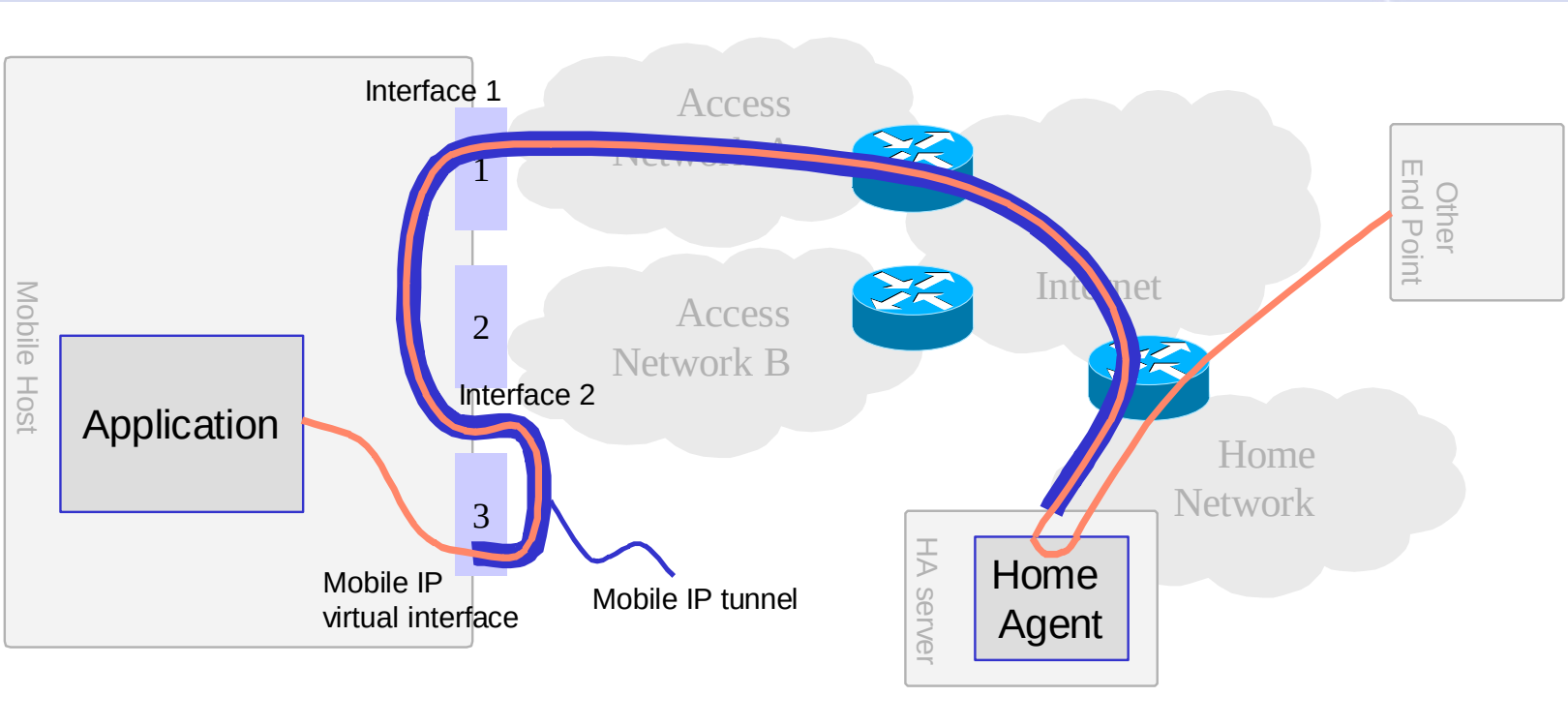


Mobility Manager with Mobile SCTP



Path Selection

- Applications can select outgoing interface by binding their connections to the IP number of the preferred interface
- Interface choice may determine usage of mobility management facility (e.g. Mobile IP virtual interface)



Interface Detection

- Supplies information about creation and deletion of network interfaces in the OS, and interface parameters
- Indicates interface type (abstraction from OS interface type)
  - `eth` (Ethernet)
  - `wlan` (Wireless LAN)
  - `dialup` (PPP dialup)

ID message	description
`NEW_EVT`	A new interface is available in the OS
`CHG_EVT`	An existing interface has changed: up or down, IP address change
`DEL_EVT`	An interface is removed from the OS
`LIST_REQ`	Request a list of current interface
`LIST_EVT`	A list of current interfaces (including characteristics)

Access Network Detection

- Responsible for scanning of available access networks
  - Some interface types (e.g. `wlan`) may allow to choose access network
- Translates access network quality parameters (e.g. signal and noise) into qualitative indication of network quality

AND message	description
`NEW_EVT`	A new network is available
`CHG_EVT`	An existing network has changed: link quality change
`DEL_EVT`	A network is no longer available
`SCAN_START_REQ`	Start scanning on a specific interface (in addition to automatic scans)
`SCAN_READY_EVT`	Scan is ready for a specific interface
`LIST_REQ`	Request a list of available networks
`LIST_EVT`	A list of currently available networks (including characteristics)

Access Network Selection

- Most of Mobility Manager's control functionality resides in Access Network Selection component
- Activates for each network interface preferred network at link layer
- Obtains and sets IP parameters for each network interface
- Responsible for controlling the below-application layer mobility management facilities
  - For Mobile IP, select the interface/network combination that provides the care-of address (CoA)
- Controls which network interface is used as the default route
  - Influences default mobility management behavior

Access Network Selection

- ANS messages

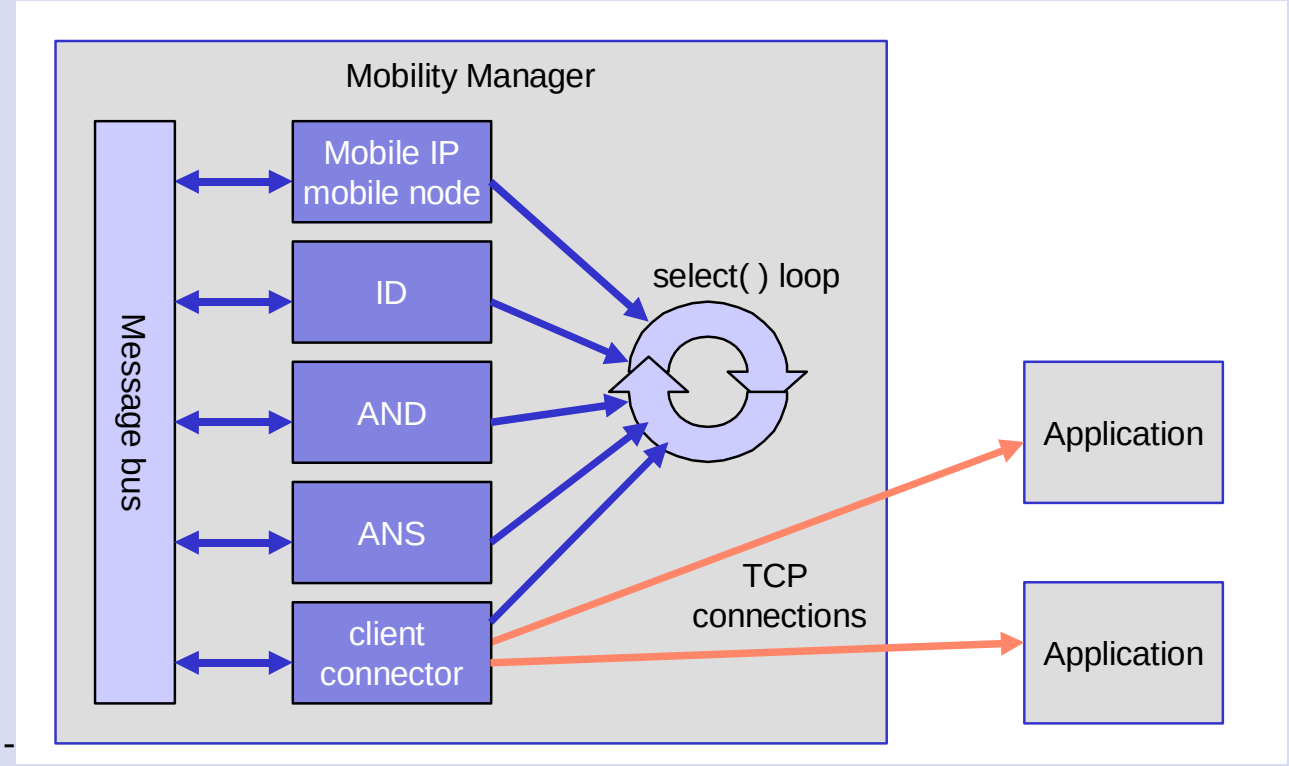
ANS message	description
AN_SELECT	A network is selected for a certain interface
AN_DESELECT	A network is deselected for a certain interface
DEFAULT_IF_SELECT	An interface is selected as default interface (default route)
MIP_COA_SELECT	An interface is selected to supply the Mobile IP CoA
MIP_COA_DESELECT	An interface is deselected to supply the Mobile IP CoA
STATUS_REQ	Request the current ANS status
STATUS_EVT	Supply the current ANS status (network selection per interface, selected default route, selected CoA supplier)

Client Interfacing: info to applications

- Path (local network) followed by IP packets for a connection
- The available below-application layer mobility management facilities
- The system default behavior for mobility management
- The current interface parameters (type, IP settings)
- The quality of available access networks

Prototype Implementation

- Mobility Manager
  - Implemented in C as user space daemon on the Linux OS
  - Target platform: HP iPAQ with Familiar Linux
  - Support for Ethernet, 802.11b, PPP/dialup

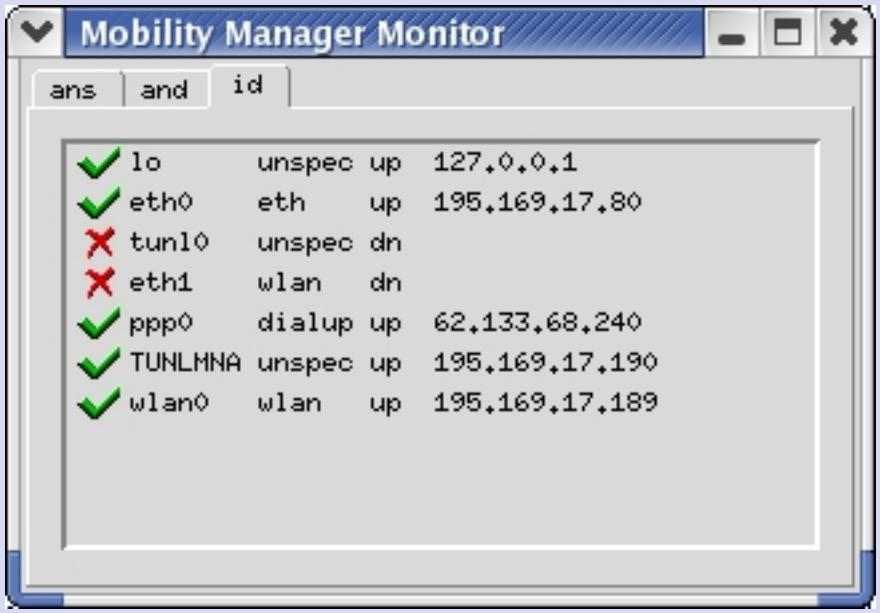


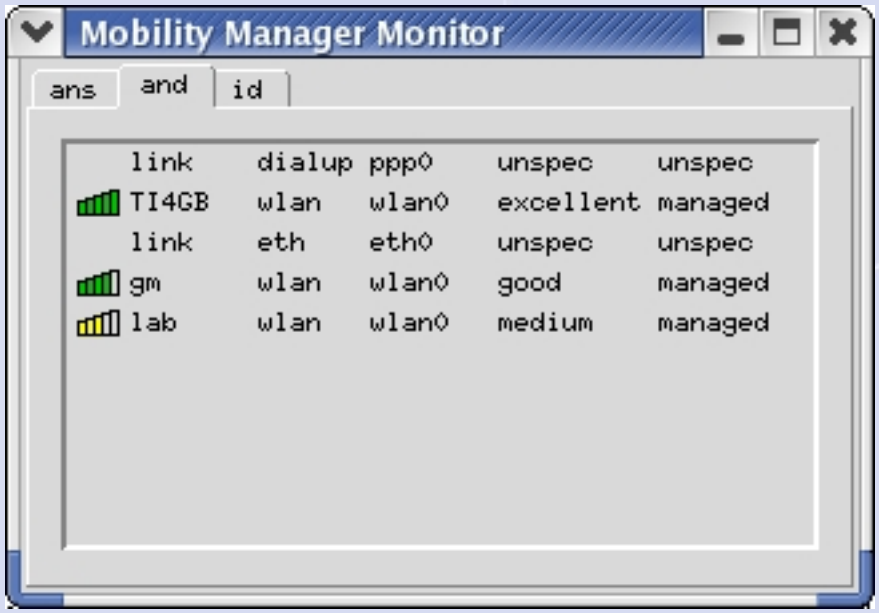
Prototype Implementation

- Interface Detection
  - Using Linux `rtnetlink` to detect interface changes
- Mobile IP
  - Stripped down Dynamics/HUT mobile host functionality
- Access Network Detection
  - Using Linux Wireless Extension + `hostap_cs` and `orinoco_cs` drivers
  - Scanning at regular intervals (every 10 seconds)
- Access Network Selection
  - Manipulate routing tables with Linux `rtnetlink`
  - Using dhcpcd dhcp client software (external executable)

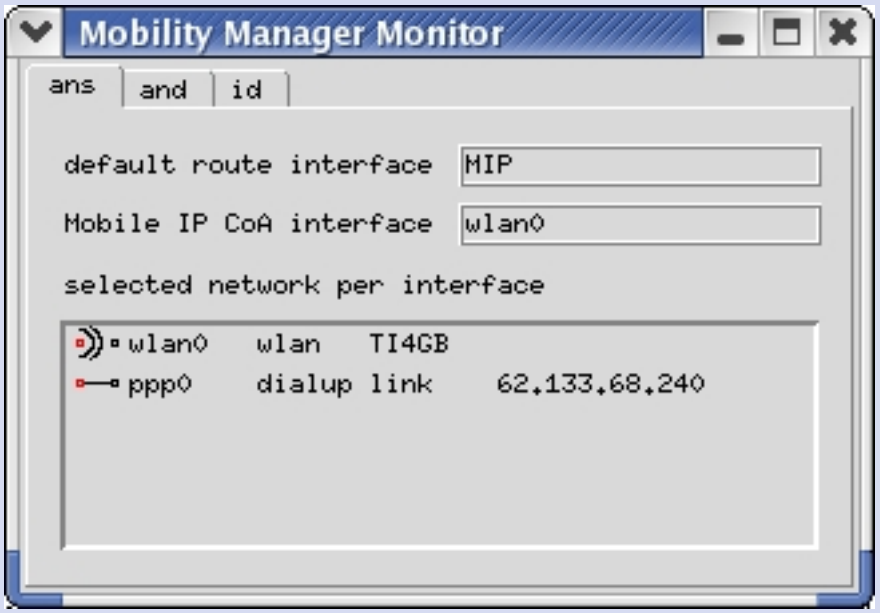
Mobility Manager Monitor

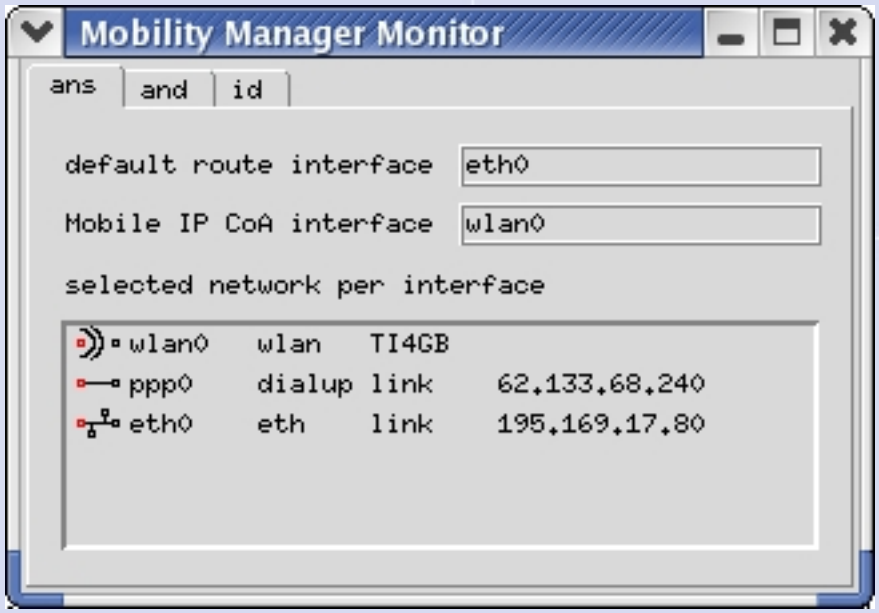
- GUI application reflecting all Mobility Manager messages





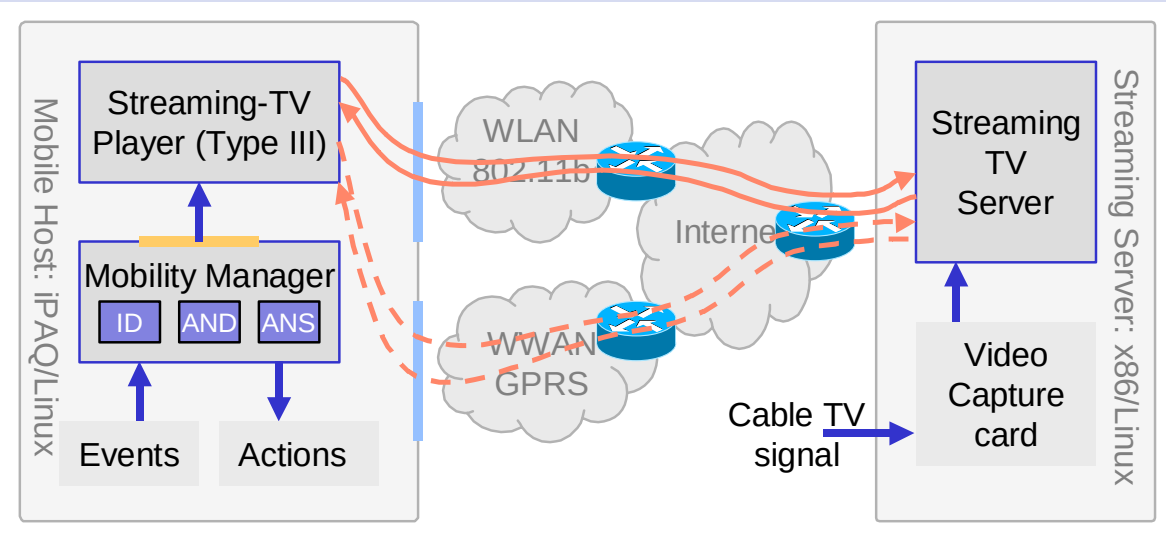
Mobility Manager Monitor





Experiments

- Streaming TV application
  - Server captures video and send out in quality as indicates by player
  - Player adapts quality and bandwidth consumption depending on available network resources on mobile host
  - Player type II: triggered by ANS `MIP_COA_SELECT` message
  - Player type III: triggered by ANS `DEFAULT_IF_SELECT` message



Evaluation and Discussion

- Multiple mobility management protocols
  - Experiments show that our mechanism works for both Mobile IP and a proprietary application-level mobility protocol (single application)
  - Would be good to use additional protocols (e.g. Mobile SCTP) to investigate modularity of our approach
- Interface abstraction
  - Interface provide by Mobility Manager to application hides many aspects and parameters from layer below the application layer
  - Still, application have to deal with much information that has not been abstracted away
  - Need more experiments with more different applications to determine “right” level of abstraction

Conclusions and Future Work

- Introduced a mechanism for host mobility management
  - For environment with heterogeneous networks
  - That cooperates and interacts with existing mobility management facilities
  - That makes applications on the mobile host aware of the mobility process and available network resources
- Proposed an application classification (from mobility management perspective)
- Experiments show that prototype implementation works with type II and type III adaptive streaming TV application
- Future work
  - Focus at level of abstraction of information towards applications
  - Capture this information in terms of *network context* and *communication context*

Mobility Manager and Monitor code available at

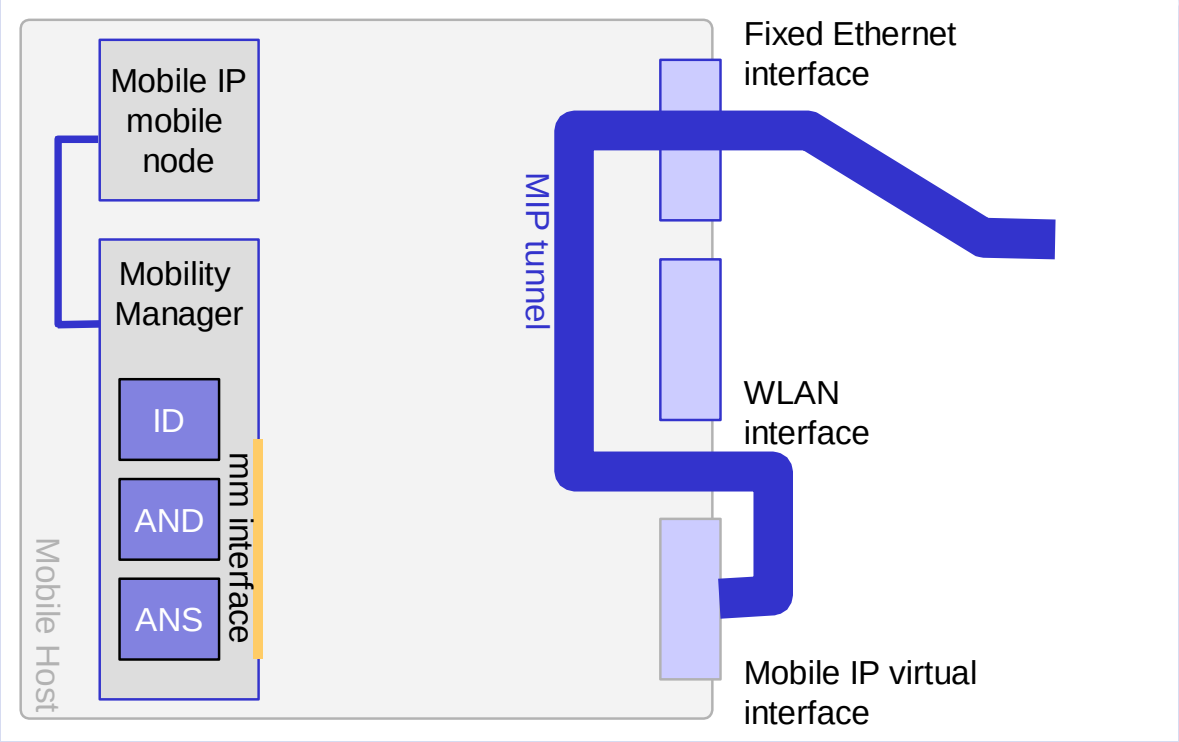
http://moma.telin.nl/

Contact: Arjan.Peddemors@telin.nl

Backup Slides

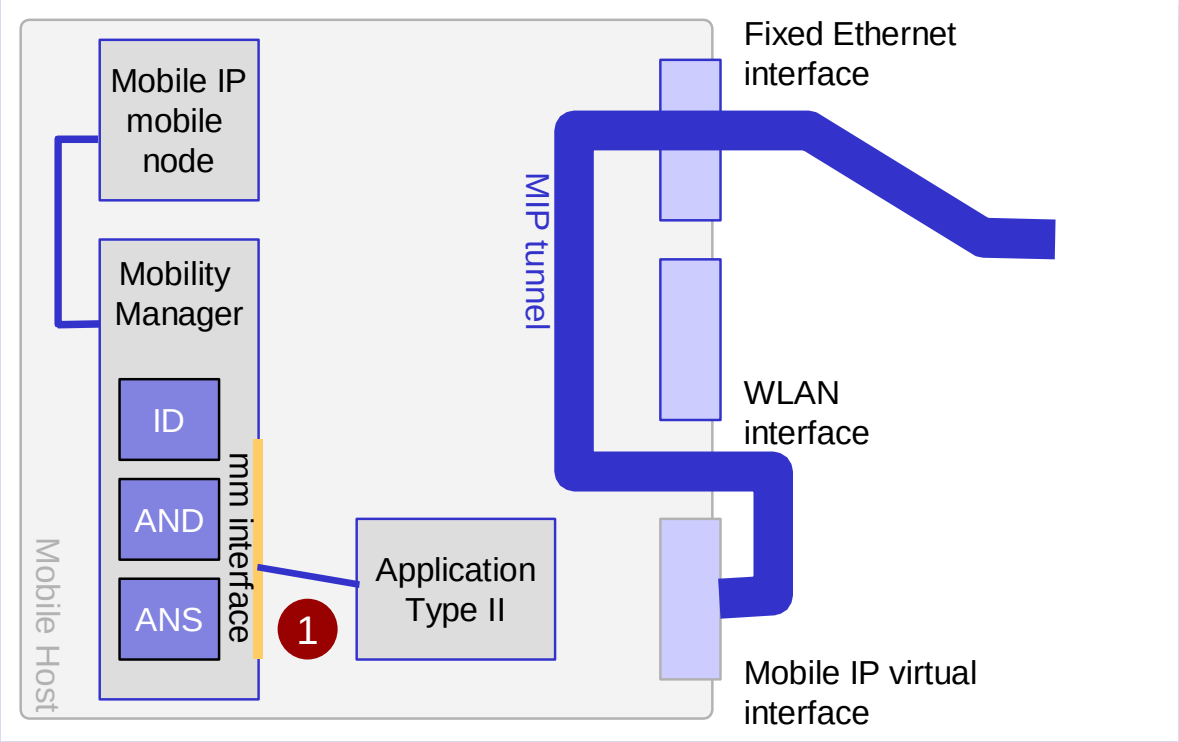
Interaction Example – Type II app

- Mobile host has access to a fixed Ethernet and an 802.11b WLAN
- Mobile host offers Mobile IP mobility management functionality
- Initially Mobile IP tunnel over fixed interface



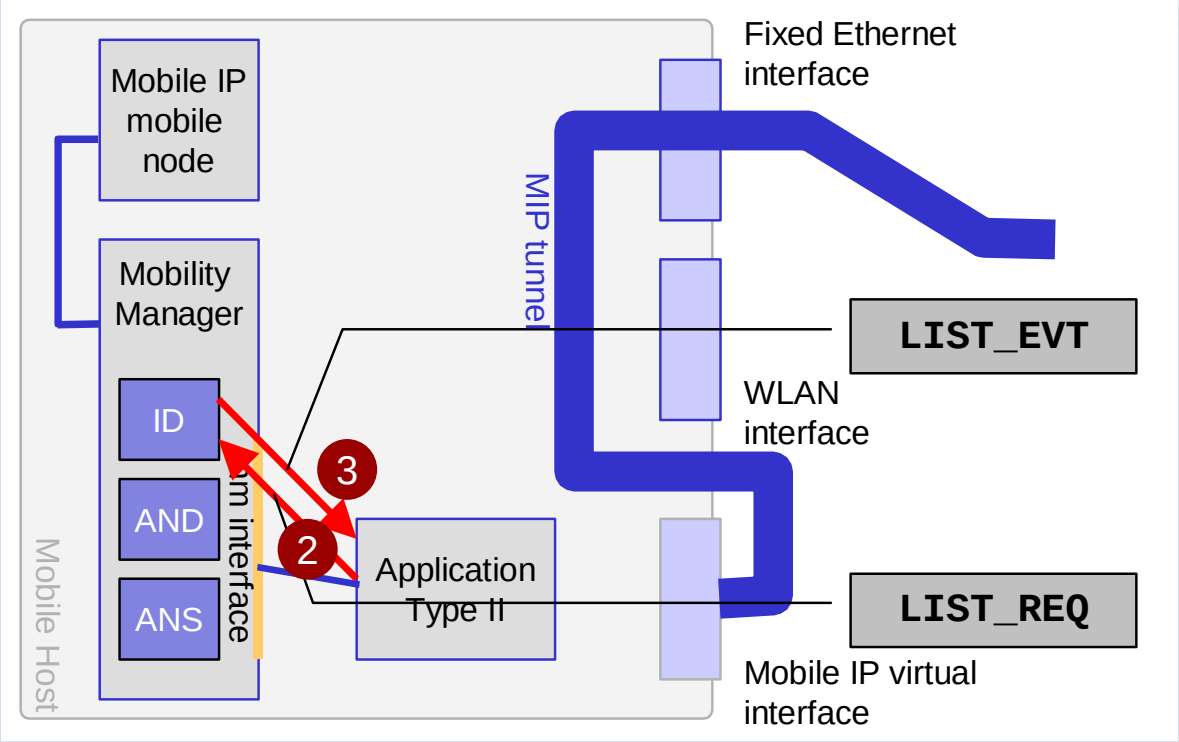
Interaction Example – Type II app

- Application starts and connects to mobility manager



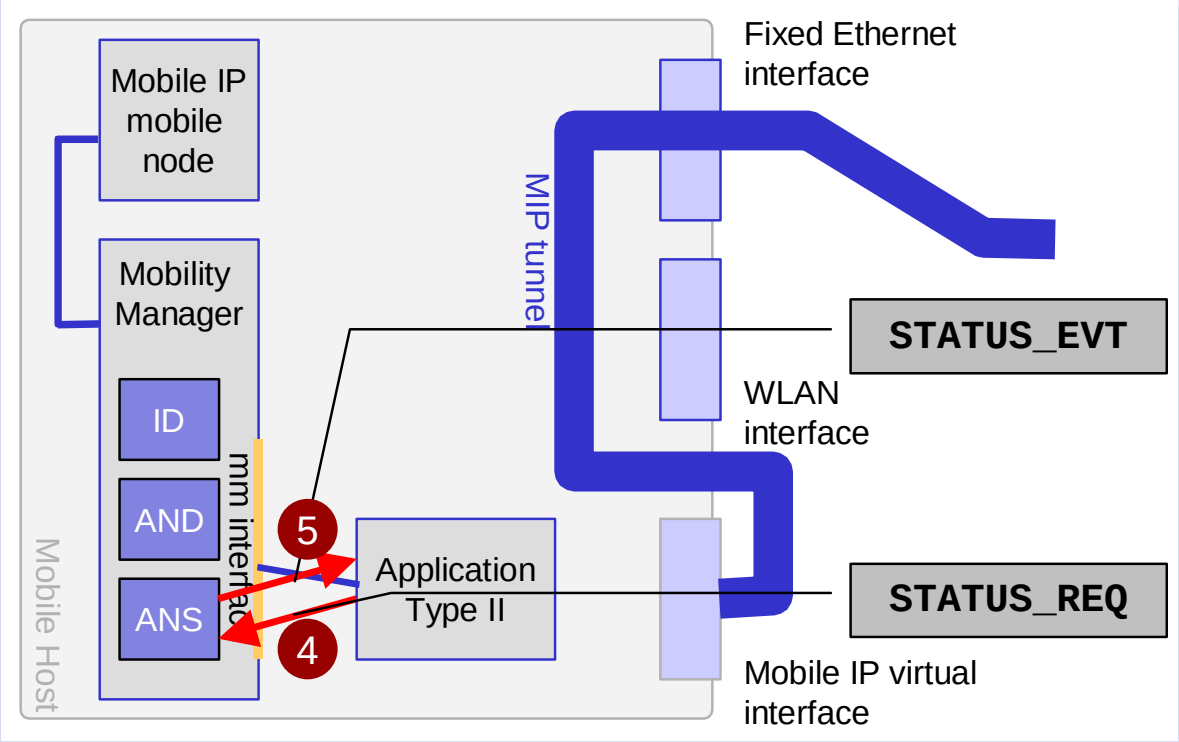
Interaction Example – Type II app

- Application sends ID `LIST_REQ` to obtain list of currently available interfaces including info on types, maximum available bandwidth, up/down states, IP parameters
- Reply with ID `LIST_EVT` (holding all this info)



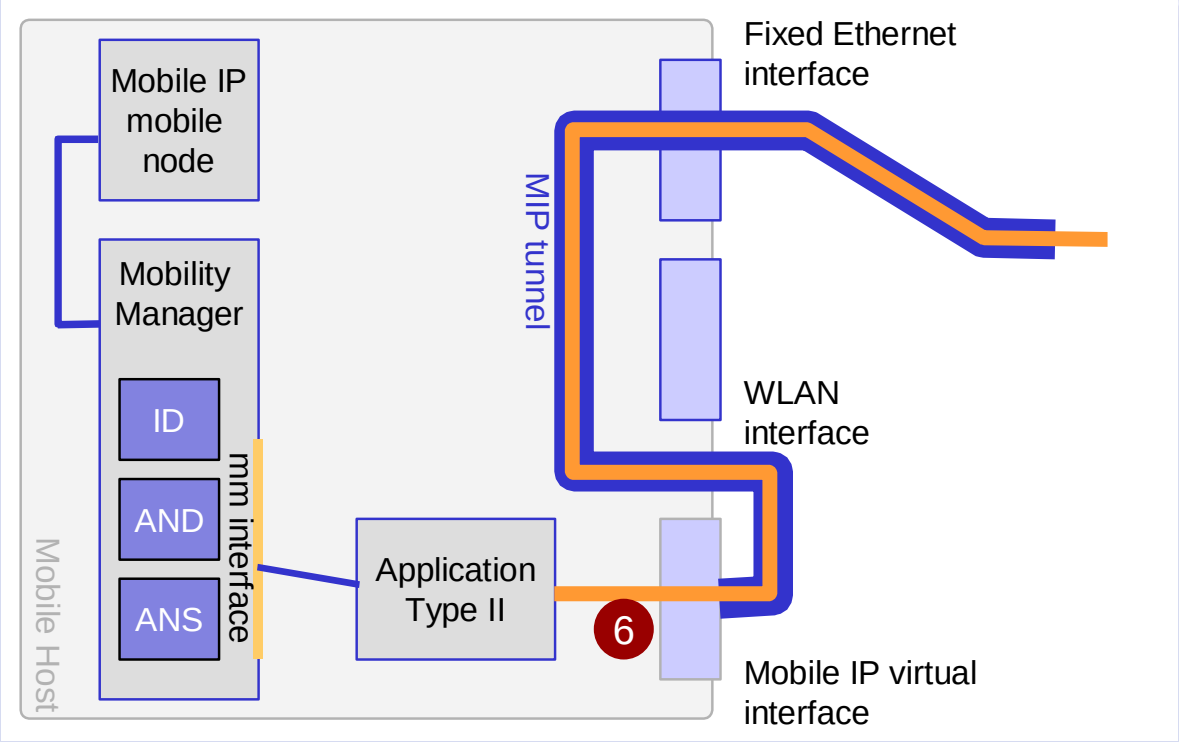
Interaction Example – Type II app

- Application sends ANS STATUS_REQ to get ANS status
- Reply with ANS STATUS_EVT indicating that fixed interface supplies Mobile IP CoA and default route is set to Mobile IP tunnel (app knows default connections are managed by Mobile IP)



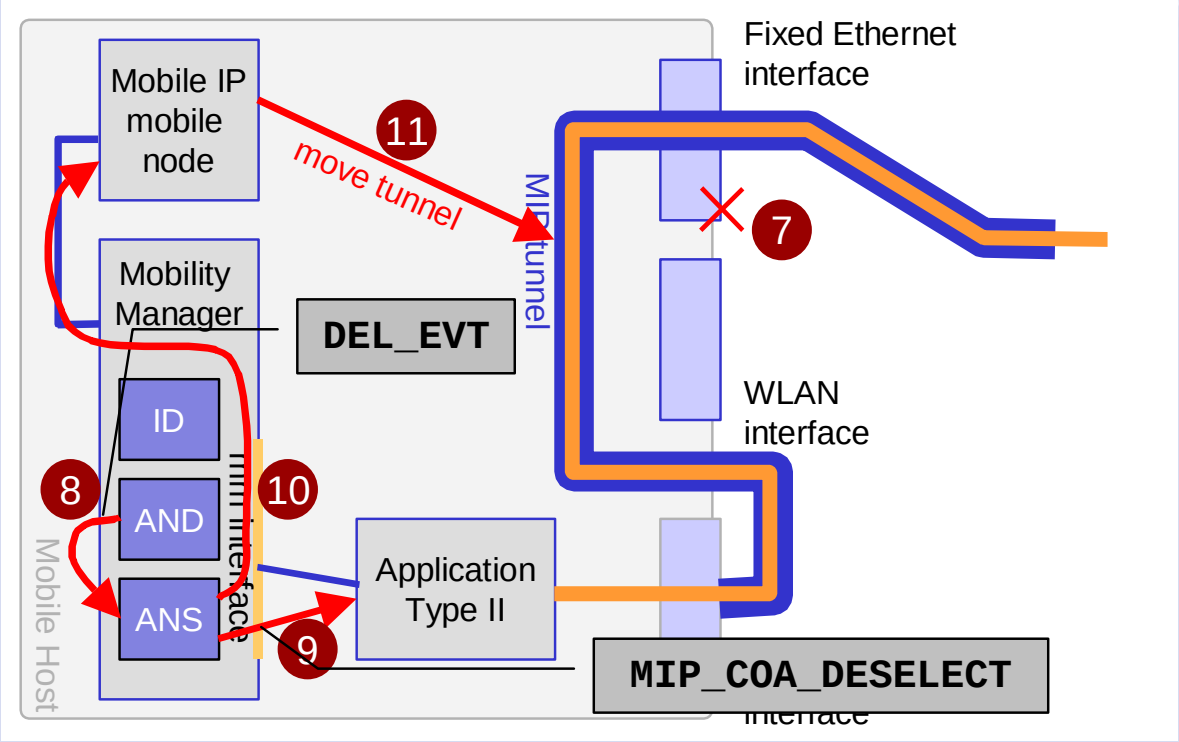
Interaction Example – Type II app

- Application initiates UDP connection with other end point using regular Sockets API: connection is managed by Mobile IP (MIP interface is default)
- Application consumes bandwidth according to `eth` interface characteristics



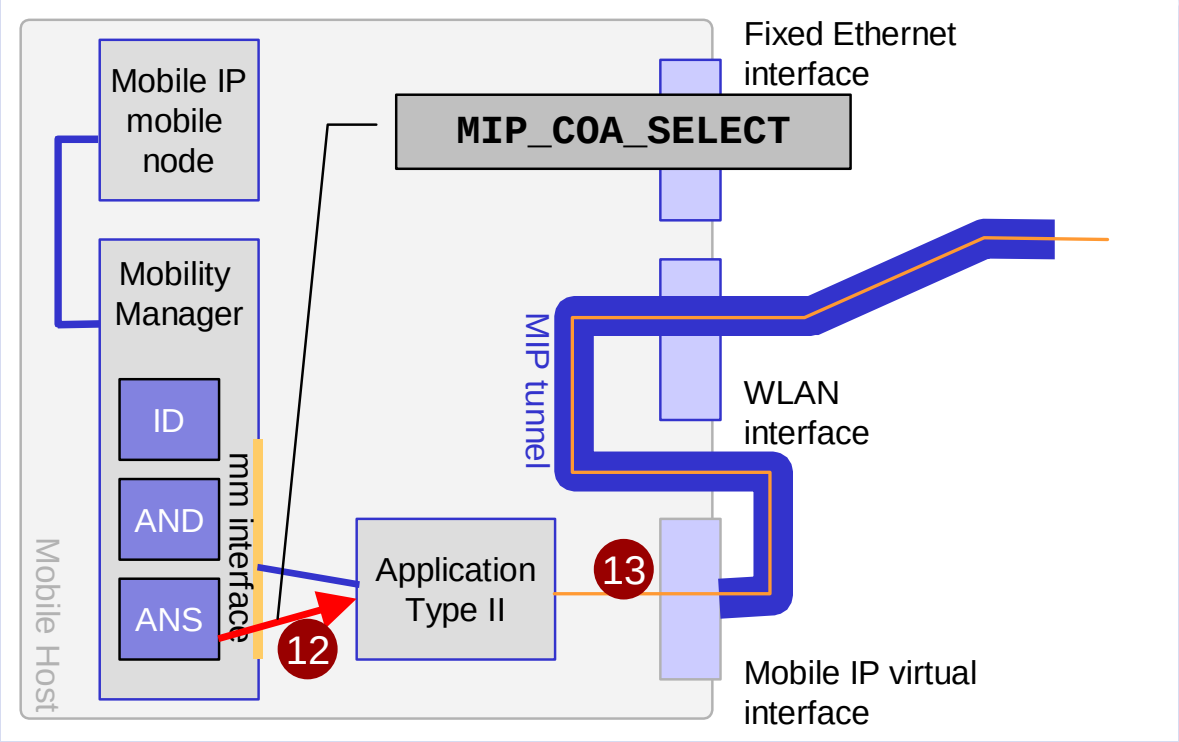
Interaction Example – Type II app

- Mobile host disconnects from fixed network; AND detects that link is down and sends `DEL_EVT`
- ANS reacts by sending out `MIP_COA_DESELECT` and instructs Mobile IP functionality to move the tunnel to the WLAN interface



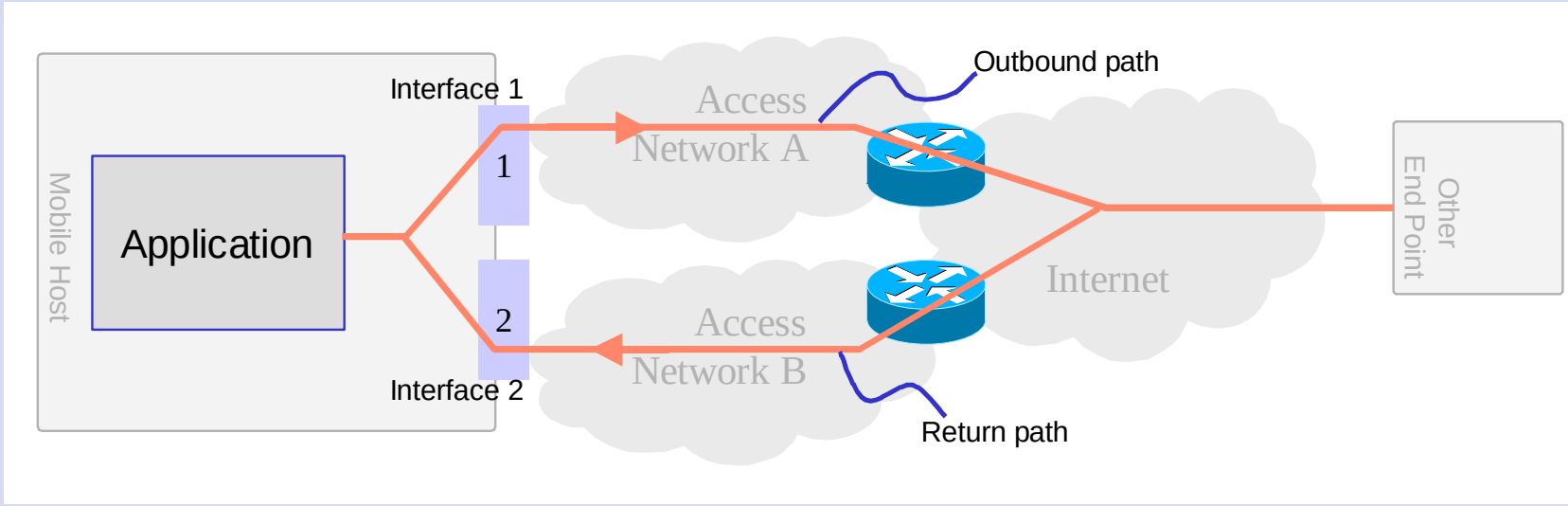
Interaction Example – Type II app

- After tunnel move, ANS immediately sends a MIP_COA_SELECT message to indicate that Mobile IP tunnel is over WLAN interface
- Application reacts by adapting its bandwidth consumption for the UDP connection



End-point default routing behavior

- Operating System selects the outgoing interface based on the destination address
- One network interface is the default interface
- When explicitly specifying the source address in a multi-homed situation, this may lead to difference in outbound and return path



Linux Policy Based Routing

- Supports configuration of multiple routing tables
- Kernel selects table based on number of attributes associated with the IP packet (e.g. source address)
  - Rules in the Routing Policy Database (RPDB)
- Mobility Manager uses this routing functionality
  - For every interface connected to network with a gateway, configure a dedicated routing table with default route to this gateway
  - Add a rule to the RPDB that prescribes that the dedicated routing table must be used when source address matches the address associated with the interface
  - Now, multiple default routes available (every interface's own routing table)
  - Application binds connection socket to IP number of preferred interface