ANA – Autonomic Network Architecture
[www.ana-project.org]

Martin.May@technicolor.com, Technicolor Research Labs Paris
Christian.Tschudin@unibas.ch, University of Basel
« Future Internet Assembly », Valencia meeting, Apr 2010
Fact Sheet

ANA is funded by the European Union

- 4 years: Jan 2006 to Apr 2010
- 10 European partners, 1 Canadian partner

FET project, part of the “Situated and Autonomic Communications (SAC)” initiative:

- New paradigms for communication/networking systems.
- 4 projects: ANA, BIONETS, Haggle, Cascadas.
ANA Mission

Goal: To identify fundamental autonomic networking principles and to demonstrate autonomic steering.

Tenet: Let a network scale in functionality, not only size.

Need to separate “mechanics” from “networking logic”

- Stack becomes a tool box and framework (common denominator)
- This playground populated by constantly changing protocol logic as well as autonomic steering logic (adaptivity, evolvability)

A future-proof communication core:

- We identify “universal” communication abstractions, this relates to “Axiomatic Basis of Communication” work
Polymorphism Support

Tangible abstractions that encapsulate key networking components:

- **Functional Block** (FB): data processing entity
- **Information Dispatch Point** (IDP): indirection/*start-points* instead of end-points
- **Compartment**: "wrappers" for networks and admin domains.
- + “technology agnostic” communication API
  *e.g.: addresses are not part of the arch*
Visual Coding of Abstractions

Architecture is about communicating among human engineers, needs common language and understanding. Color codes used:

- **Compartment/CT**, recursive
- **Information Dispatch Point/IDP**
- **Information Channel/IC**, recursive
- **Functional Block/FB**, recursive
Expressing NW personalities in ANA, evolving them

Layers can still be expressed, but more composition flexibility: recursiveness, dynamic rewiring

re-binding of IDP 'c' is transparent for users of 'c' (function f2 here)

re-binding is a simple change in dispatch table
Nodes serve as „common ground“, behave like a compartment, strictly do „packet switching inside the node“ (instead of the IPC metaphor), are natural place to do virtualization (rsrc isolation)
Work on Autonomic Steering

Beyond „stack mechanics“, research is done on:

- Turfnet addressing/routing + Fisheye routing (NEC)
- Content Centric Routing (UPMC)
- Functional Composition (ULanc)
- Service Discovery + Placement + Migration (NKUA)
- Virtual Coordinate Systems (ULiège)
- IP independent Transport Layer (ETHZ)
- Integrated Monitoring
ANA and Future Internet

- ANA as a meta-architecture for networks
  - extracts and refactors core networking concepts,
  - supports heterogenous addressing and naming styles:
    - ANA is not address or protocol centric
    - future-proof systems are about evolvability and dynamic reconfiguration.

- ANA as a driver to promote disruptive NW research.
  - no IP-backward-compatibility constraints.
  - allows new ideas and experiments to emerge and interact.
  - "Let 1000 networks bloom" concept (see the 4WARD project)
ANA = tangible software

Available for Linux, incl. smartphones (Android, N810), and a partial Erlang version:

- **ANA minmex** ("micro-kernel") and various components:
  - compartment implementations (eth, IP, Turfnet)
  - functional composition framework
  - protocols: content centric routing, service discovery, transport etc

- **Configuration and visualization tools**
  - to explore node internals
  - sub-system for creating and managing virtual links.

- **APIs** are important! C, Lua, JNI, and Erlang wrappers.

GUI apps being developed for final ANA review:

- Transport agnostic chat on Android (Ethernet, email, SMS, twitter)
- Video-on-demand streaming application on Nokia N810.
Assessment of Uptake

ABC - Axiomatic Basis for Communication (UWaterloo):
  • so far first steps only (ANA fits well)
  • but we still lack a GRUNT="grand unified networking theory"

The 4WARD project is about “let 1000 networks bloom”, i.e. a meta-arch.
  • Compartiment: copied as-is
  • “generic path”/GP = “beefed up and distributed information channels
    all having a common management interface”
  • “mediation points”/MP = special rewiring places for GPs

General (personal) notes:
  • architecture discussions are surprisingly religious in style
  • old school thinking hard to overcome (layers, protocol entities, addr, end-points)
  • some „contributions“ more confusing than helpful („Patterns in net arch“)
Thank you

Questions?
Partners

- ETH Zurich (CH)
- University of Basel (CH)
- NEC (DE)
- Lancaster University (UK)
- Fraunhofer Fokus (DE)
- University of Liege (BE)
- University Pierre et Marie Curie (FR)
- NKUA (GR)
- University of Oslo (NO)
- Telekom Austria (AU)
- University of Waterloo (CA)
Project organization

- Project is articulated around 2 prototyping cycles.
  - "Learn by doing": design, test/validate, refine.

<table>
<thead>
<tr>
<th>2006</th>
<th>2007</th>
<th>2008</th>
<th>2009</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design phase</td>
<td>First prototyping phase</td>
<td>Intensive testing phase</td>
<td>2nd prototyping phase</td>
</tr>
<tr>
<td>First &quot;Blueprint&quot; (architectural model)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2nd design phase</td>
<td>Mature &quot;Blueprint&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final evaluation</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Networking Abstractions (!)

Goal: To demonstrate the feasibility of autonomic networking.
- Identify fundamental autonomic networking principles.
- Design and build an autonomic network architecture.

ANA in a blink:
- Network must not only *scale* in size but also *in functionality*.
- Evolving network: variability at all levels of the architecture.
- ANA = framework for function (re-)composition.
- Dynamic adaptation and re-organization of network.

Networks have to work: doing *research through prototypes*.
- Early build an experimental network architecture.
- Prototype used as feedback to refine architectural models.
Some background motivation

- Fundamental networking research is not possible on the Internet.
  - Variability in the Internet is above and below IP:
    it's the "hour-glass" model.

Changing/更新 the Internet core is difficult or impossible!
(e.g. IPv6, Multicast, MIP, QoS, …)

www, email, ftp, ssh, DNS,
peer-to-peer (eMule, BitTorrent)
VoIP (Skype), VoD, grid, …

Ethernet, wifi (802.11), ATM,
SONET/SDH, FrameRelay,
modem, ADSL, Cable, …
ANA != "one-size-fits-all"

- ANA does not want to propose another "one-size-fits-all network waist".
  - ANA is a framework to **host, interconnect, and federate multiple heterogeneous networks**.
  - ANA introduces the core concept of "network compartments."

Multiple "network compartments" can co-exist
ANA: a meta-architecture

- ANA does not impose how network compartments should work internally: the ANA framework specifies how networks interact.

ANA specifies interfaces and interactions with network compartment.

Internal operation is not imposed leading to multiple and heterogeneous compartments but generic interaction.

ANA framework
**Network compartments**

- A (network) compartment implements the operational rules and administrative policies for a given communication context. It defines:
  - How to **join and leave** a compartment: member registration, trust model, authentication, etc.
  - How to **reach** (communicate with) another member: peer resolution, addressing, routing, etc.
  - The compartment-wide policies: **interaction rules** with "external world", the compartment boundaries (administrative or technical), peerings with other compartments, etc.

Compartments decompose communication systems and networks into smaller and easier manageable units.
Network compartments (cont.)

- In addition, the compartment abstraction serves as the unit for the federation of networks into global-scale communication systems.

- Compartments can be overlaid, i.e. compartments can use the communication services of other compartments (and vice versa).
Network compartments (cont.)

- Registration and resolution are key functionalities of compartments.
  - Conceptually, each compartment maintains a membership database.
  - Registration: explicit membership is required ("default-off" model).
  - Resolution: explicit request before sending ("no sending in the void").
What about addresses and names?

- Addressing and naming are left to compartments.
  - Each compartment is free to use any addressing and naming schemes (or is free to not use addresses, for example in sensor networks).

- The main advantages are:
  - No need to manage a unique global addressing scheme.
  - No need to impose a unique way to resolve names.
  - ANA is open to future addressing and naming schemes.

- The main drawbacks are:
  - Global routing becomes something similar to searching.
  (if communicating parties are not all members of a given compartment).
Local labels for handling (global) addresses

- "Resolution of members" results in a local label
  - Addresses (if any) and names (if any) limited as input for resolution
  - Applications send data to labels (which stands for a communication entry point)

Properties of local labels:
- Size of labels can change from device to device.
- Labels' lifetime = communication lifetime (like sockets).
- No need to manage a unique global addressing scheme.
- ANA is open to future addressing and naming schemes (via resolution).
Information channels (ICs)

- Resolution process returns access to an "information channel" that can be used to reach the target member(s).

- Various types of information channels.

unicast  multicast  anycast  concast
Information dispatch points (IDPs)

- Startpoints instead of endpoints.
  - In ANA communication is always towards a startpoint, or information dispatch point (IDP).
    - Ability to bind to destinations in an address agnostic way.
    - This is important to support many flavors of compartments that can use different types of addresses and names.
    - Useful decoupling between identifiers and means to address them.

![Diagram showing data sent to IDP which has state to reach destination]
**Functional blocks (FBs)**

- Code and state that can process data packets.

- Protocols and algorithms are represented as FBs.
- Access to FBs is also via information dispatch points (IDPs).
- FBs can have multiple input and output IDPs.
- FB internally selects output IDP(s) to which data is sent.

![Diagram of FBs and IDPs](image_url)
How ICs, FBs, and IDPs fit together
What about addresses and names? (cont.)

Main challenges for ANA are:
- How to gateway heterogeneous compartments.
- How to federate and overlay compartments.
- How to access compartment services in a generic manner.
- How to recursively resolve a name through the collection of compartments.

How does ANA support heterogeneous addressing and naming schemes?
- "Clients" of ANA interact with IDPs (i.e. local labels/numbers).
- Packet dispatch in the ANA core is also based on IDPs.
- Access to compartments is via generic functions/primitives.

Key concept is to cleanly separate compartment internals from abstractions exposed to users of compartment services.
What about addresses and names? (cont.)

Why is this useful?

- "Separation of concerns" is fundamental.
  - e.g. client wants to reach www.switch.ch and does not care whether this resolves to 130.59.138.34 or 2001:620:0:1b::b or whatever else.

- Flexibility to add new addressing and naming schemes.
  - Internet transition to IPv6 has required almost all applications to be modified to support IPv6: changes in network layer affect application layer!
  - A network architecture should not be centered around its addressing scheme.

- Flexibility for each compartment to autonomously handle its addressing and naming schemes.
  - Which relates to the popularity of NAT in today's Internet.
What about addresses and names? (cont.)

Why is this useful?

- Emerging network paradigms are not always centered around names and addresses.
  - Sensor networks: core "routing" block is data (fetching, aggregation, etc).
  - Content routing: routing is based on set of keywords (e.g. printer, 2\textsuperscript{nd} floor).
  - Delay tolerant networks: intermittent connectivity, no traditional addresses.
  - MANETs: spontaneous networks with no topological organization.

- Names are just a means to single out an entity inside a set.
  - What one cares about is the entity, not its name.

- Addresses are just a means to carry data to some destination(s).
  - What one cares about is the path, not how it is labeled.
What about addresses and names? (cont.)

- fine … but do IDPs become the new invariant?

  - Yes but how IDPs are implemented is a local issue.
    - Each ANA node could have its own way of representing IDPs.
      - (e.g. char, unsigned int, MD5 hashed values, etc …).
    - Think of file descriptors in Unix filesystems: there are many different filesystems, but a generic way to access files.

- okay … but compartments become opaque "black boxes"?

  - Yes and no … each compartment is free to "expose" internal details via dedicated functions.
    - E.g. each compartment can provide extra information available for each IDP via a `get_idpinfo()` function.
The "glue" of ANA: the API

- Network compartments are free to internally run whatever addressing/naming schemes, routing protocols, etc.

- The "glue" for all interactions in ANA is the compartment API.

- All network compartments must support the API in order to allow all possible interactions between compartments.
The API offers 6 fundamental primitives.

\[
\begin{align*}
\text{IDP}_p & \text{ publish(IDP}_c, \text{ CONTEXT, SERVICE)} \\
\text{int unpublish(IDP}_p, \text{ IDP}_p, \text{ SERVICE)} \\
\text{IDP}_r & \text{ resolve(IDP}_c, \text{ CONTEXT, SERVICE)} \\
\text{int release(IDP}_r) \\
\text{void* lookup(IDP}_c, \text{ CONTEXT, SERVICE)} \\
\text{int send(IDP}_r, \text{ DATA)}
\end{align*}
\]
CONTEXTS and SERVICES

- The SERVICE argument is typically **what** is being published or looked up.
  - e.g., an address, a name, a file, a video stream, a printing service, etc.

- The CONTEXT defines some **scope** inside a compartment.
  - IPv4: 1.2.3.4, 224.0.0.1, 10.1.2.255.
  - IPv6: 2001::1, FF02::1, ::1.
  - DNS: A, AAAA, MX.
  - eMule: Madonna, Pink Floyd, Blade Runner.
CONTEXTS and SERVICES

- We have currently specified two "well-known" CONTEXT values.
  - "." → node-local
  - "*" → largest possible scope as interpreted by the compartment

Examples:

- **IPv4 compartment:**
  - "." ~ 255.255.255.255
  - "." ~ 127.0.0.1

- **IPv6 compartment:**
  - "*" ~ FF02::1
  - "." ~ ::1
Using the API: some examples

Publishing an IPv4 address in the Ethernet compartment.

\[
z \leftarrow \text{publish}(y, "\star", "10.1.2.3")
\]
Using the API: some examples

Resolving an IPv4 address in the Ethernet compartment.

\[
s \leftarrow \text{resolve}(e, "\ast", "10.1.2.3")
\]
Using the API: some examples

Sending data.

\[
\text{send}(s, \text{DATA})
\]
The Node compartment

- It provides a local and private "execution environment" for each "client" of the ANA framework.
  - A "client" is an entity (application, FB, or compartment) using ANA.
- It is the "compartment" startpoint of any communication.
The Node compartment

- The Node compartment is the context in which FBs and IDPs exist.
- In contrast, ICs only "live" in network compartments.

This shows that there is really just one IDP "mapped" in the different views.

Underlaying Hardware

"ANA world"

Send to medium
(Ethernet, wifi, etc)

Listen to medium
(Ethernet, wifi, etc)

IC abstracts service
provided by
underlying hardware

exported view

structural view

imported view

data in

data out
"Chains" of functions are setup on-demand in a dynamic way.

Packet dispatching in ANA is based on IDPs.

Re-binding of IDP 'c' is not visible to users of 'c' (function f2 here).

Re-binding is a simple change in dispatch table.
Overlay scenario with compartments
Overlay scenario with compartments (cont.)

- Same figure but only with exported views of L* compartments.
Overlay scenario with compartments (cont.)

- Figure just showing export view of compartment N.

Which could also be drawn like that (just showing the export view).
Prototyping in ANA

- Prototyping is an essential activity in ANA.
  - Architecture is refined during/after each prototyping cycle.
  - Feedback from testing phase is critical.

- Project has developed an ANA node prototype.
  - All functionalities of ANA + an abstraction layer to run ANA on different operating systems or on dedicated hardware (e.g. network processors).

- Still in developments:
  - The system is being ported to handheld devices like the Android, iPhone, Nokia N810.
  - A scripting facility (with Lua) for quick developments of components.
  - Demonstration of the architecture with concrete applications.