

# We Got Trouble!

Right Here in River City  
With a capital T and that Rhymes with P  
and That Stands for *IP!*

John Day  
Dublin 2014

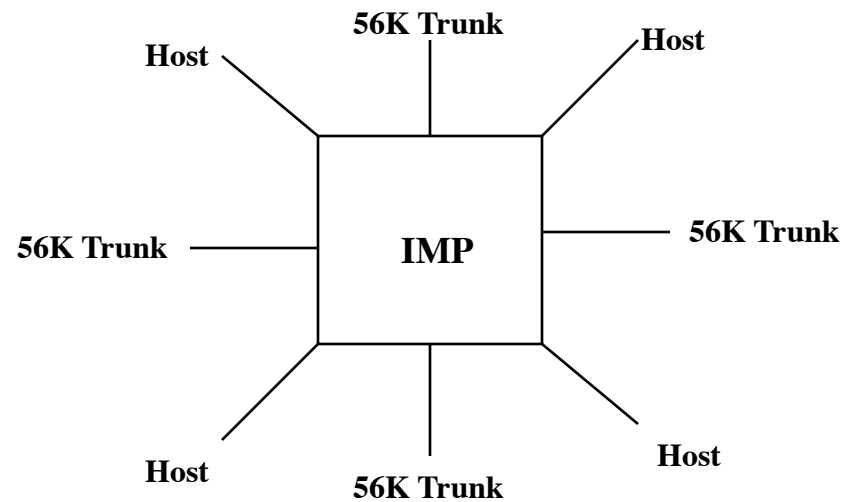
It doesn't have to make sense. It's religion!  
-Robbie Coltrane  
Nuns on the Run

## As the Song Says

- Most of our Troubles start with IP
- And the Lack of a Complete Addressing Structure
- To Understand Why this is the case, we need to go back in time, back to . . .

# Addressing in the ARPANET

## The Root Cause of our Problems



**Each ARPANet IMP (switch) had ports to support a maximum of 4 trunks and 4 hosts.  
Each IMP had a number. The host address (IP address) was the IMP # and the  
Host #, i.e. a port number. Maximum number of hosts was huge: 63.**

**So a host's address was its IMP Port Number.**

## Was There a Reason?

Sure

It was easy to build for an *experimental* network of this size.

Was there a lot of thought given to how addressing should work?

Not really.

We were doing good to do this much!  
There were many much bigger problems to overcome:  
Like just moving data

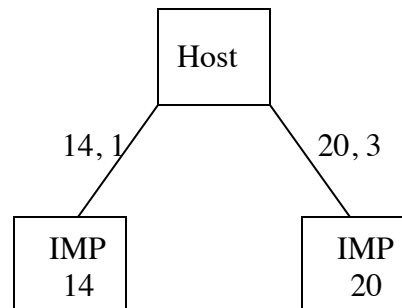
And addressing is a hard problem.

# Did it take long to realize there was a problem?

**Nope.**

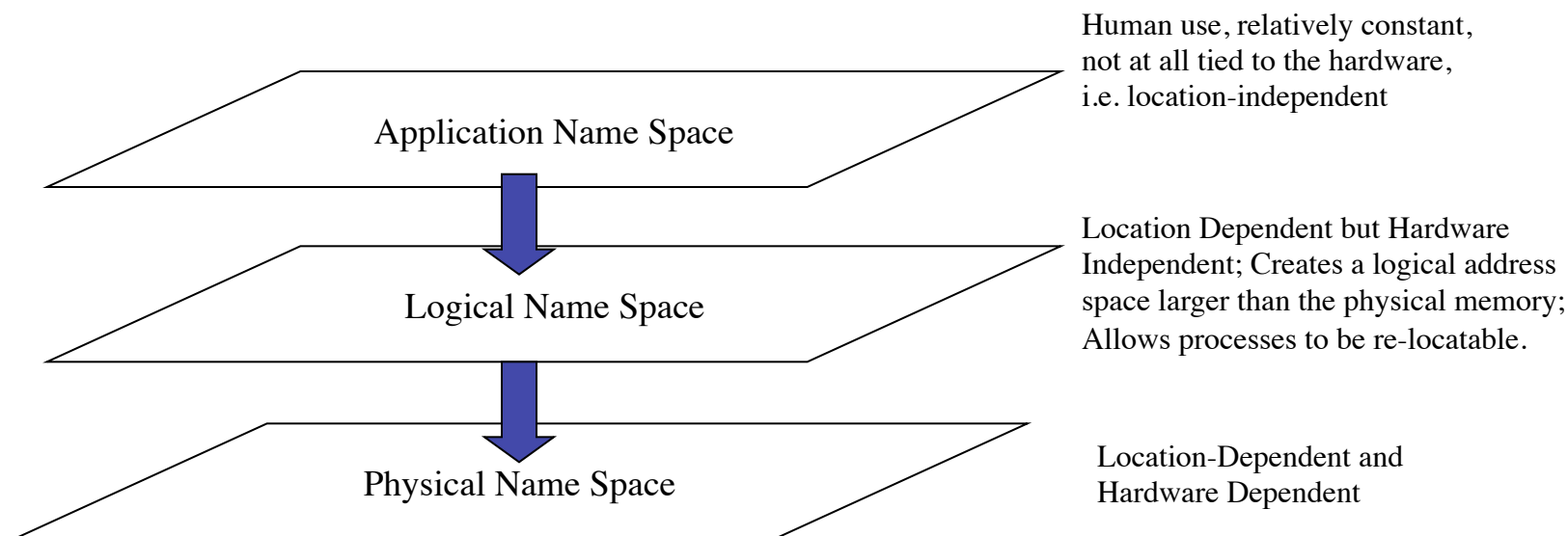
**First time (~ 1972) one of the Air Force bases took us at our word that the network was suppose to be survivable and asked for links to two different IMPs to connect its host to the Network.**

**Naming the hosts by the names of their interfaces meant that the two connections looked like two hosts to the Net.  
Still does.**



# Address Spaces in Operating Systems

(From my OS Course)



An name space is defined as a set of identifiers with a given scope.

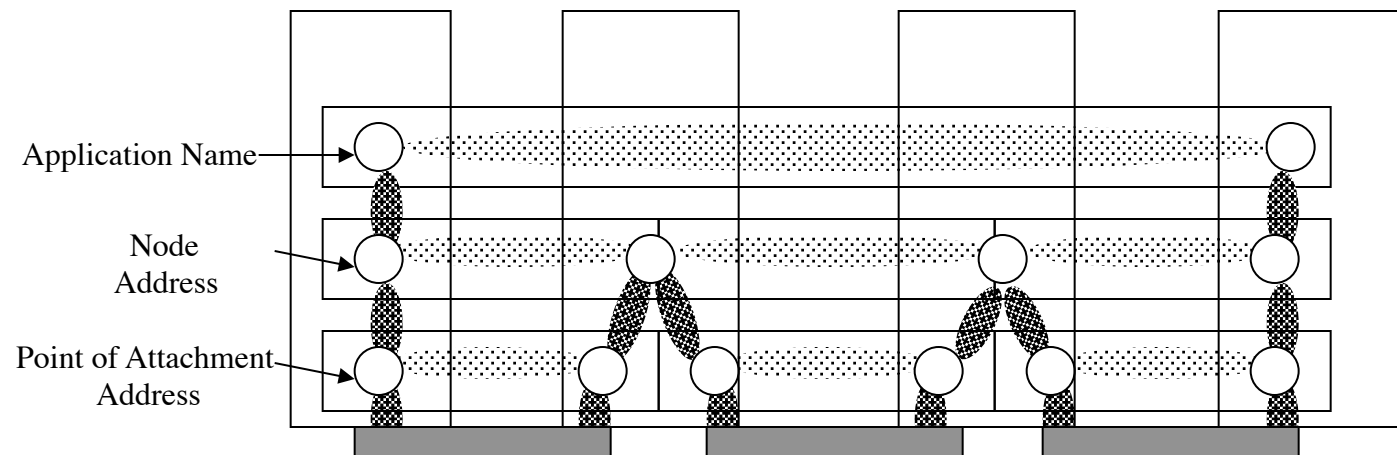
An address space is a location-dependent name space.

In Operating Systems, we have found a need for 3 such *independent* spaces.

Virtually all uses of names in computing are for *locating*.

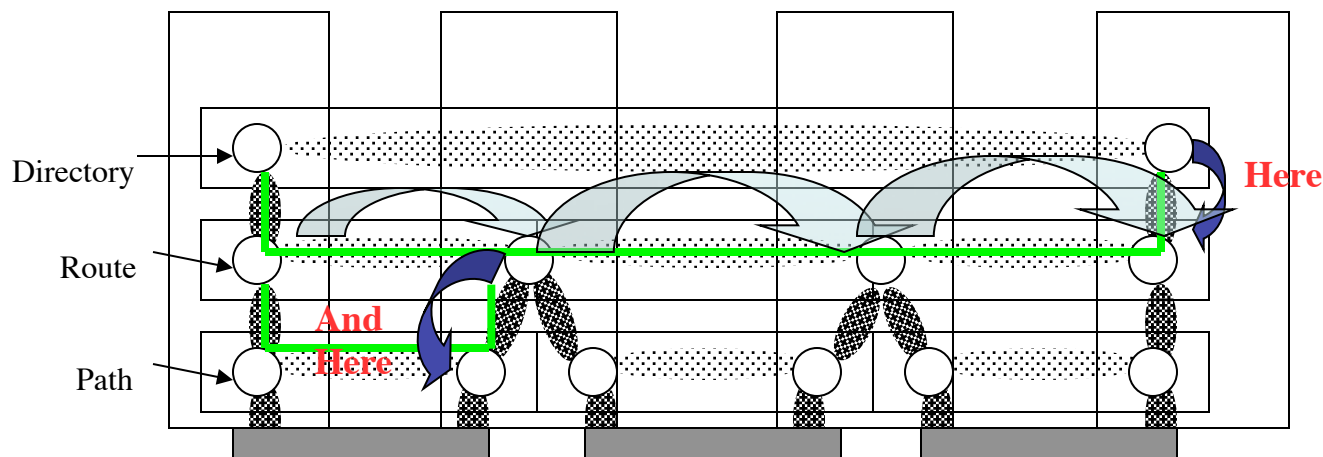
# Saltzer's View of Networks

- Application names map to node addresses.
- Node addresses map to points of attachment addresses.
- Routes are sequences of points of attachments.
  - Just as in an operating system.



# But Networks Are More General

- Directory maintains the mapping between Application-Names and the node addresses of all Applications reachable without an application relay.
- Routes are sequences of node addresses used to compute the next hop.
- Node to point of attachment mapping for all nearest neighbors to choose path to next hop. (Even though Saltzer notices this case, he misses its importance.)
- This last mapping and the Directory are the same:
  - Mapping of a name in the layer above to a name in the layer below of all nearest neighbors.

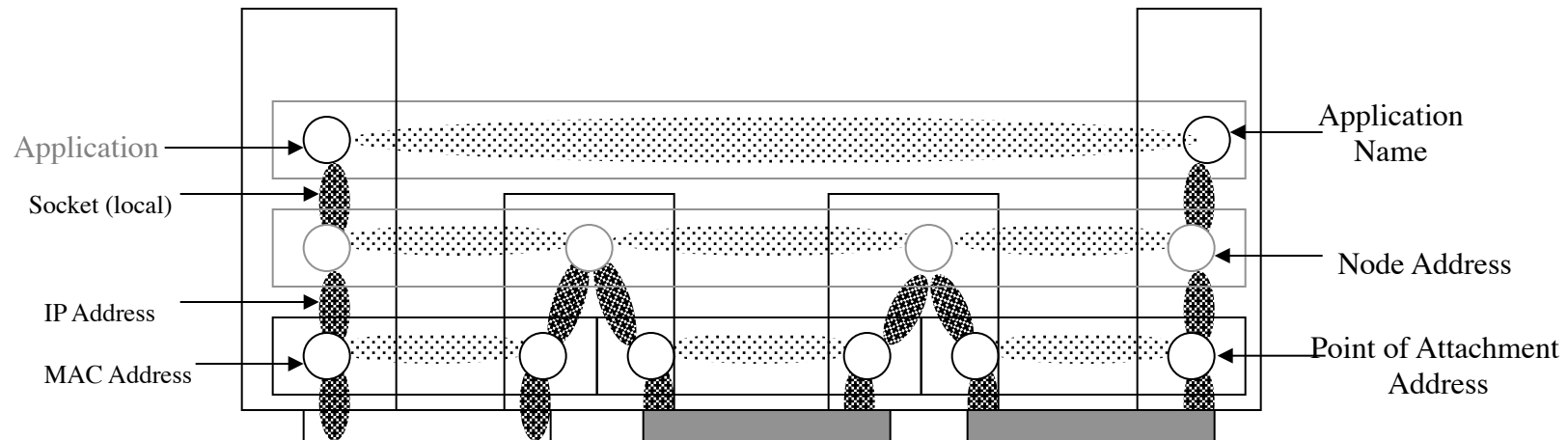




# Applying Results to Real Architectures: The Internet

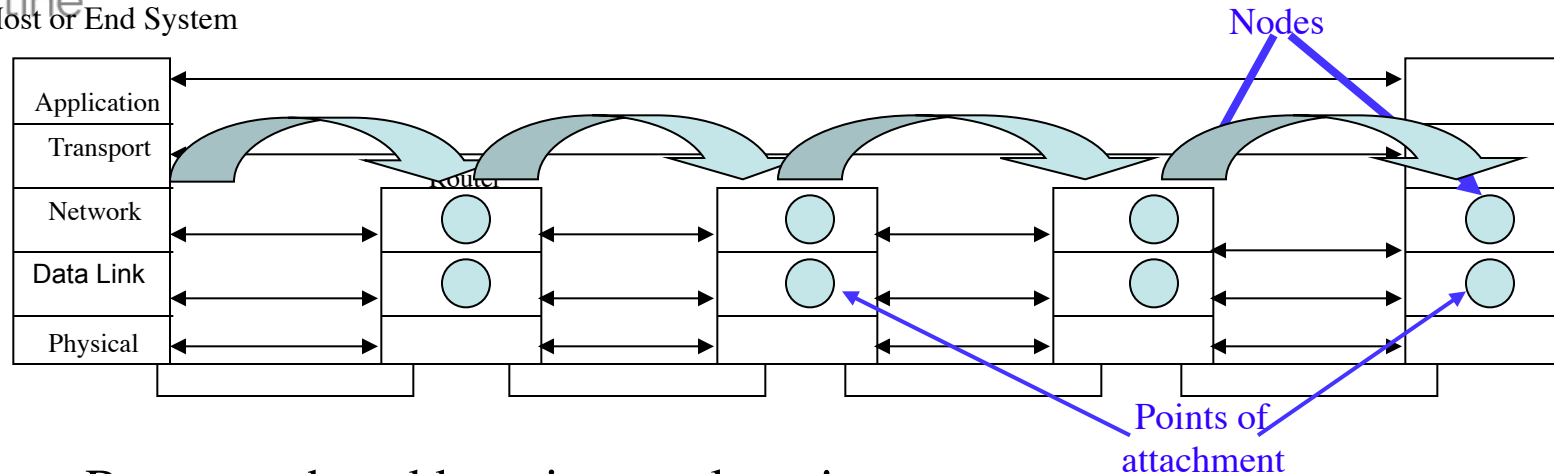
(This is a *Data Comm* Architecture)

- The most striking feature is that half of the addressing architecture is missing.
  - No wonder there are addressing problems.
  - The only identifier we have for anything is the IP address.
- There are no node addresses and no application names.
  - And the point of attachment is named twice!
  - If this is an *Internet* Protocol, where are the *Network* Addresses? (Lost Layer)
  - Domain Names are synonyms for IP addresses. URLs name a *path* to an arbitrary instance of an application.



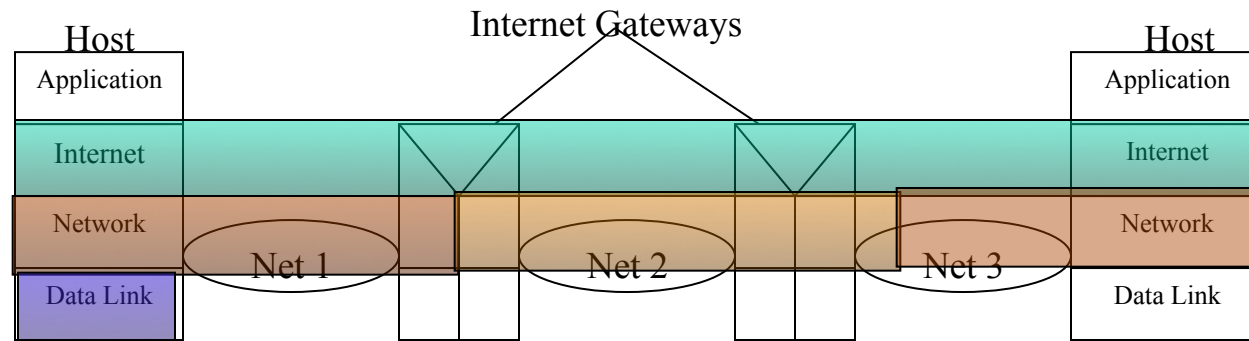
As if your computer worked only with absolute memory addresses.

# There is An Easier Way to See It.



- Route on the address in your layer!
  - Well, duh!
  - Routing on the Interface is routing on the address in the layer below.
- Learned a couple of other things along the way
  - Addresses only need to be unique within the scope of a layer
  - Addresses must generally be assigned by the network because it is the only one that knows *where* in the network the node is.
  - Don't concatenate an (N)-identifier with an (N-1) address to form an (N)-address.

# If This Were an Internet Architecture, Then



- There would be multiple Data Link Layers with limited scope.
- Network Layers with greater scope that did intra-domain routing
- Internet Layer with greater scope yet that did intra-domain routing.
- Network Layer addresses would be interface addresses for the Internet nodes.
- That brings us to the first Internet [sic] addressing crisis.

# The First Great Internet Addressing Crisis

- In 1992, we have the first addressing crisis.
  - IPv4 addresses are getting scarce
    - But the real problem is Router Table Size is increasing exponentially.
- The IAB convenes the ROAD process
  - Recommends CLNP as IPv7
    - Basically IP with variable length aggregateable addresses.
    - CLNP names the node. Hence, fixes the multihoming problem.
      - Oddly enough, OSI has an Internet architecture, but doesn't draw attention to it.
- The IETF goes berserk!
  - No OSI, no way, no how!
    - A model? We don't need no stinking model. *We've got*
    - Rough Consensus and Running Code!

# The IPng Process

- The Rules were:
  - Fixed Length Address
  - Continue to Name the Interface.
  - At least 20 octets of address
  - Open Standard
- Violá! IPv6!
  - or anything but CLNP.
- Still no solution to multihoming
  - Problem is now 20 years old

# All is Not Well



- Less than a Year later, light dawns (slightly)!
  - Name the Interface, but route aggregation is a must
  - Implies provider based assignment.
  - Means change providers, must renumber. WHAT!?
    - Kind of shot yourself in the foot, eh?
- Transition is dual stack with a NAT
  - Once you have a NAT, you don't need v6 . . . oops.
    - How many feet do you have?

# Techs Play Architect

- Finally around 2000, need to deal with multihoming, but
  - given negative reaction to naming the node, need a workaround
- Ahh! The problem is that we have been overloading the semantics of the IP address with location and identifier information.
  - We need to split them. Loc/id split is the Answer.
  - A locator we can route on and a flat endpoint id (EID)
    - Psssst! Can't identify something without locating it and vice versa
      - Saltzer [1977] defines “resolve” as in “resolving a name” as “to locate an object in a particular context, given its name.”
    - Got another foot?
- Don't bother me with pedantic terminology,
  - IPv6 is the future!

# Trouble is Not Much Interest in v6



- It offers no benefit to those who pay for adopting it
  - They just don't know they want it.
- For the next decade, there is the hype:
  - Better security, better QoS, better mobility
    - “A desert topping and a floor wax!” - Mike O'Dell
      - In fact, all of these are the same as IPv4 . . . no different
- “IPv6 has all the benefit of a minor technology change with all the disruption of a major fork-lift upgrade.” - Geoff Huston, 2008.
  - Just switch to v6 and all will be well.
- By 2000, dawning awareness the architecture is running out of steam
  - NEWARCH, Clean Slate, FIND and GENI are hot!
  - Starts to be a lot of talk about loc/id split.
  - This is clearly the answer . . . . (really?).




# Houston, We Have a Problem

## (The Second Great Internet Addressing Crisis)

- October 2006, major presentation at IEPG.
  - Router table size is on the increase, due to multihoming.
    - It is either quadratic or exponential, can't tell yet.
      - (does it matter!?)
  - Moore's Law won't bail us out this time.
    - This is a big time crisis. We are in big trouble.
- If not fixed, it is the end of the Internet as we know it.
  - Net will fragment. Costs in the core will skyrocket.
  - NetworkWorld sits on the story for a year.
  - Tons of papers written on loc/id split!

# But We Do Multihoming!

(not really)

- We kludge it.
- Because we route on the interface, this forces route aggregation to be provider-based.
  - Addresses with a common prefix go to the same provider then we can store a single route (to the provider) rather than a route for every address on that provider.
- To do multihoming, must assign provider-independent addresses or new AS numbers (same thing).
  - Can't be aggregated  Router table size increases
  - Remember what our problem is? Router table size is increasing

# So Why Wasn't It Fixed?

- Odd that a DoD network touted to survive nuclear attack didn't support redundant links. Lots of “good reasons:”
  - Not that many hosts need to be multi-homed.
    - Not then, but the ones that did were the ones everyone wanted to get to.
  - Not everyone should have to bear the cost for a few.
    - Classic committee politics: Put a condition on the solution that guarantees any proposal will be rejected (asymmetry in this case)
    - Also assumes there is a cost.
  - Multihoming will be to different providers, so no point.
    - Assumption is wrong and even if right assumes a static network.
  - Remember, we tried to fix it but it was rejected by the IETF.

## But By This Time

- Cisco and others start proposing solutions to the multihoming problem.
  - Mostly requiring patches involving NATs and a bevy of new protocols.
  - In particular, LISP or Loc/ID Split Protocol
    - This makes everyone nervous, but what else to do?
    - Well, we could work out a theoretical framework to see what is going on.
- This is a crisis! We have to build something!
  - Best way known to stampede people to your view.

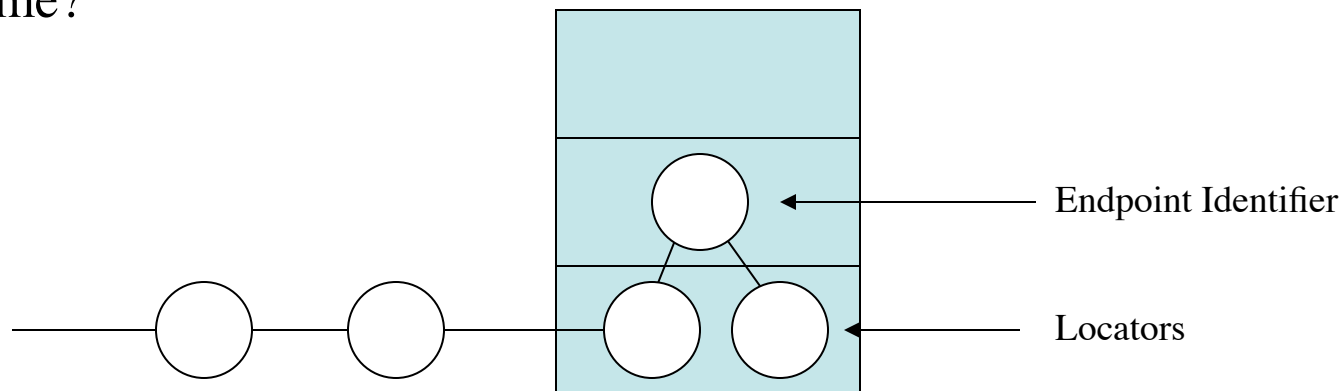
November 2008

# Houston, We have a Bigger Problem

- Dave Meyer calls, ‘I have an “architectural issue” to discuss.’
  - He has come across two problems in implementing LISP.
  - Both require doing path discovery.
  - Path discovery doesn’t scale.
  - LISP won’t scale. QED
  - He suspects that any loc/id approach will have the same problem.
    - draft-meyer-loc-id-implications-01.txt
    - In case you didn’t notice, we just went to DefCon1
- Dave: Why hasn’t anyone noticed this in the last 15 years?

# Dave is Right

- All proposals based on loc/id split have the same flaw.
- Once put in the context of the RINA theory, it is obvious.
- Let us look at it very carefully. What do the locator and the identifier name?



## The Locator Locates the Wrong Thing!

The locator is part of the path, not the final destination.

No wonder Dave ran into *path* discovery issues.

(Beads-on-a-string again: wires connecting boxes!)

Apply the e2e principle!  
(the answer to everything)



# Solve the Problem in the Hosts

- There are poor misguided souls claiming this.
  - Mostly done by changing the definition of multihoming.
  - Ignore that it might take seconds if not minutes to do the failover.
- Remember the fundamental problem is that the network doesn't know that two paths go to the same place.
  - This is a problem of *delivering*, not sending.
  - There is no host-based solution.
- There is no solution as long as one routes only on the interface address.  
Which means . . .

# If Loc/ID Doesn't Scale

- Then there is no solution involving routing on the interface that scales.
- In other words,

## IP is Fundamentally Flawed

(v4 or v6)

- But we saw that coming a long time ago.



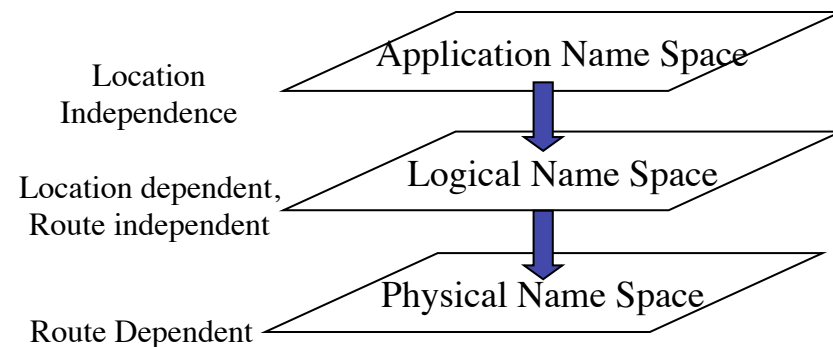
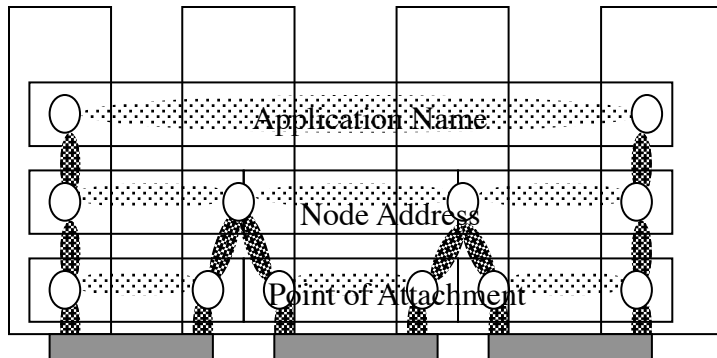
# Wanna Hear the Real P\*sser?

- We had the right answer in 1992.
  - The answer we had known since 1972.
    - It was rejected by the IETF.
- And to add insult to injury, it was **DEPLOYED** and Operational in the routers.
  - We could have spent the last 15 years working on transition
  - Rather than 100s of millions on a small incremental step that provides no benefit to your bottom line and is fundamentally flawed.
  - You just can't make this stuff up!

It was always about

# Separating Logical Location from Physical Location

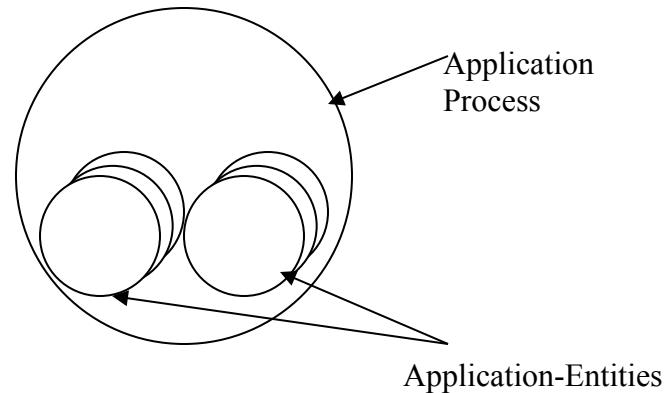
- It is impossible to locate something without also identifying it.
- This pseudo-problem arises from not having a complete address architecture.
  - And creates enough epicycles upon epicycles to make Clavius proud
- But we will give O'Dell the last word:
  - When all you have is a hammer, everything looks like your thumb
- It is still more like DOS than anything else.



# Addressing In RINA

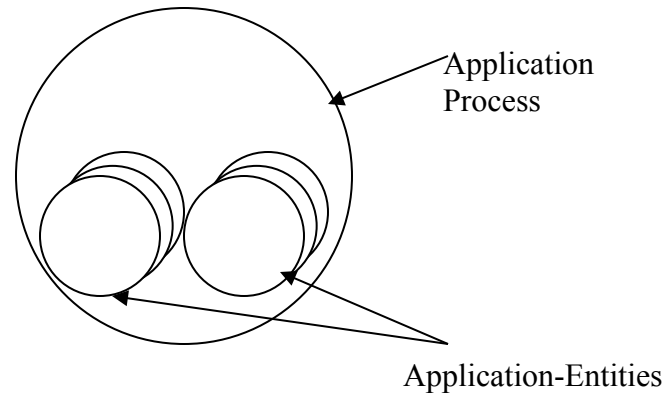
- All identifiers are based on the nature of the application process or in this case, the IPC Process.
- First, lets look in brief:

# Applications and Communication



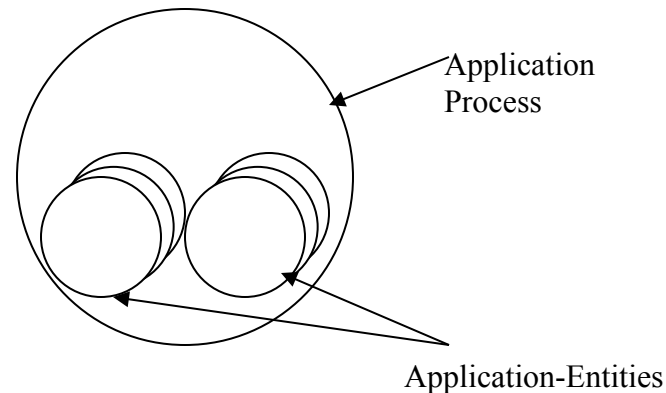
- The Application-Entity (AE) is that part of the application concerned with communication, i.e. shared state with its peer.
- The rest of the Application Process is concerned with the reason for the application in the first place.
- An Application Process may have multiple AEs, they assumed, for different application protocols.
  - An HTTP library linked into a web browser is an AE; FTP is another.

# Application Naming



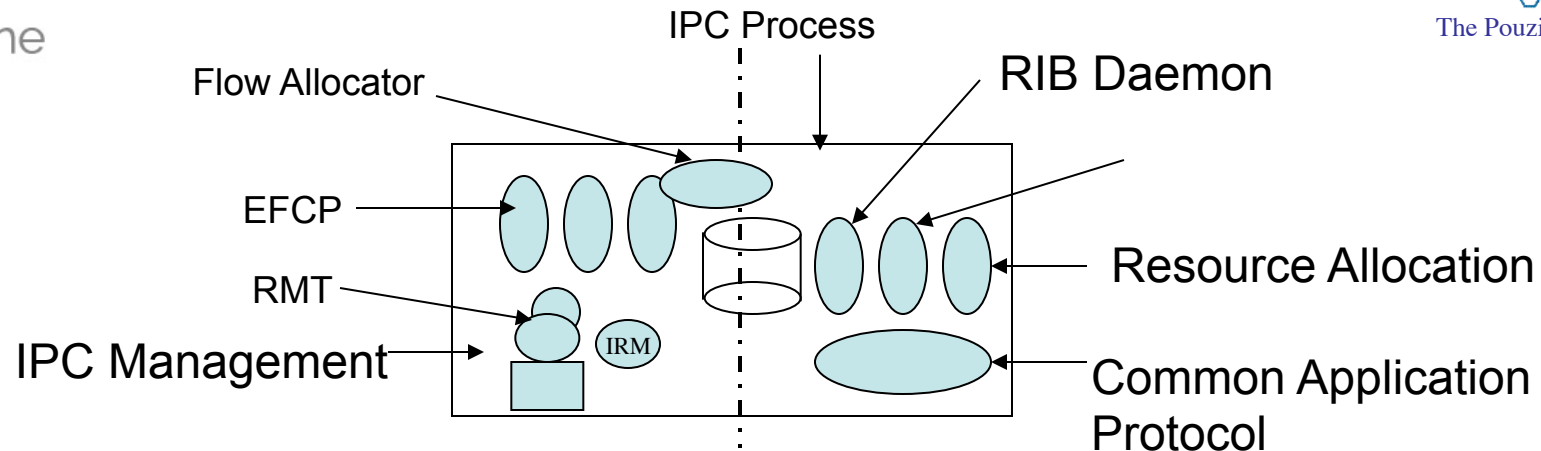
- Application-process names (APN) are globally unambiguous and location-independent, but system-dependent.
  - They may have synonyms of less scope from the same or different name space.
  - There may be multiple instances of the process in the same system.
    - APN-instance-identifiers are unambiguous within the scope of the Application Process.
- Application-entity-identifiers are unambiguous within the application process.
  - There may be more than one Application-entity (AE) in a process.
    - Unambiguous within the scope of the Application Process.
  - There may be more than one instance of each type of Application-Entity.
    - AE-instance-identifiers are unambiguous within the scope of the AE.
- Distributed Application Name is the name of a set of application processes and system-independent.
- Few applications need all of these but a complete theory requires all of them.

# What Good is All This?



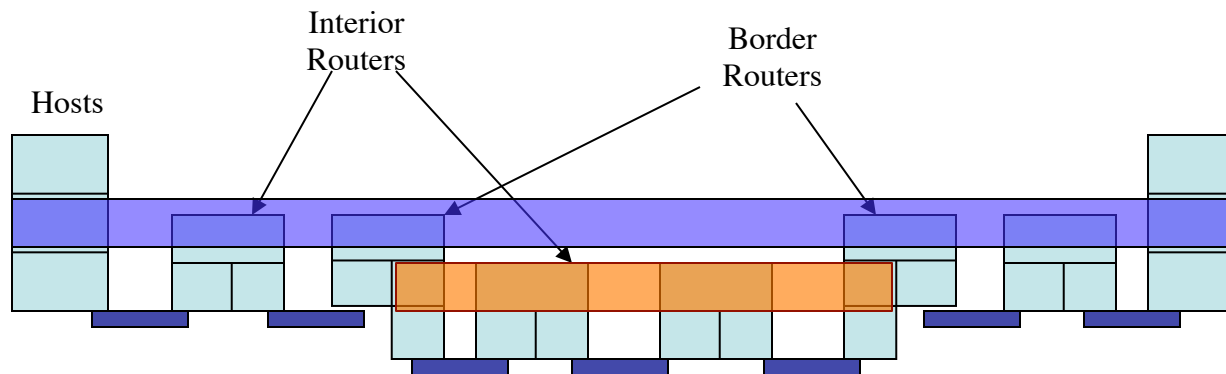
- Many capabilities not possible today or that require specific protocols are a consequence of naming and enabled by a complete architecture.
  - Handing off a connection from one system to another;
  - The need to pass IP addresses among applications is avoided;
  - Opening multiple connections with different “protocols” to the same instance of an application process.
  - Connecting to an existing “conference” call, etc.
  - And probably 1000s of things we haven’t thought of yet.

# Naming in RINA



- **IPC Process-name** is just an application-process-name
  - An **address** is a synonym for an IPC Process whosescope is restricted to the DIF and maybe structured to facilitate use within the DIF.
- A **port-id** is a Flow-Allocator-Instance-Id, an AE instance.
- A **connection-endpoint-id** (CEP-id) is an EFCP-instance-id, an AE instance.
  - Note that these are local to the IPC Process.
- A **connection-id** is created by concatenating source and destination CEP-ids.
- That's It!

# Routing at Different Layers

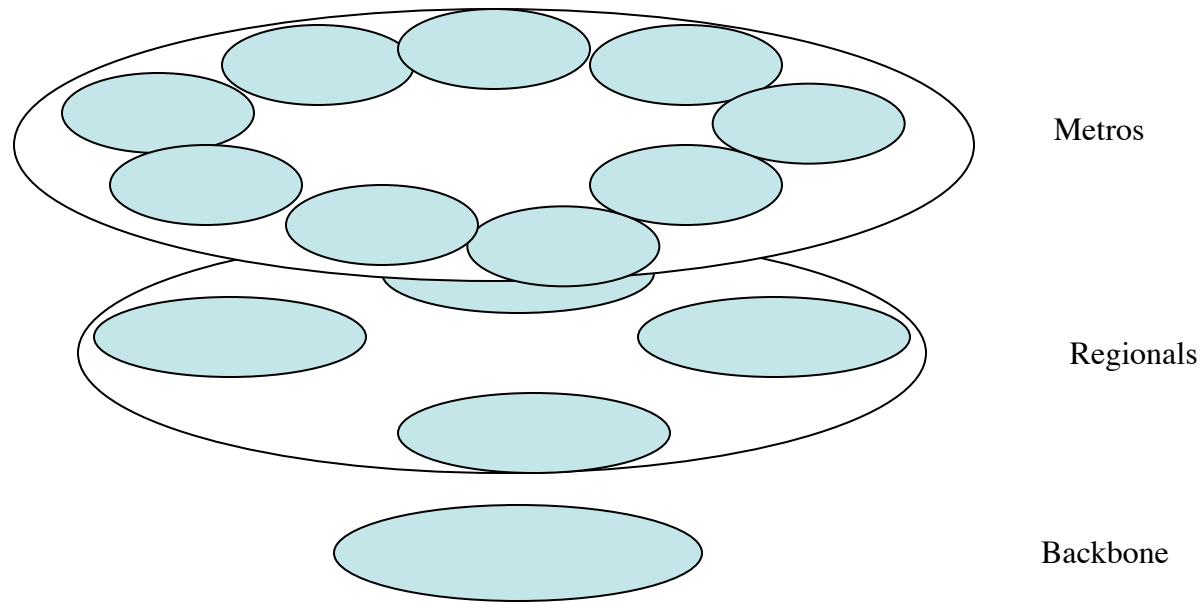


- With a Recursive Model, there are levels of routing with border routers acting as the step-down function creating interior flows.
- This tends toward a “necklace” configuration.



# Implications of the Model & Names (Routing Table Size)

- Recursion either reduces the number of routes or shortens them.



# Implications of the Model & Names

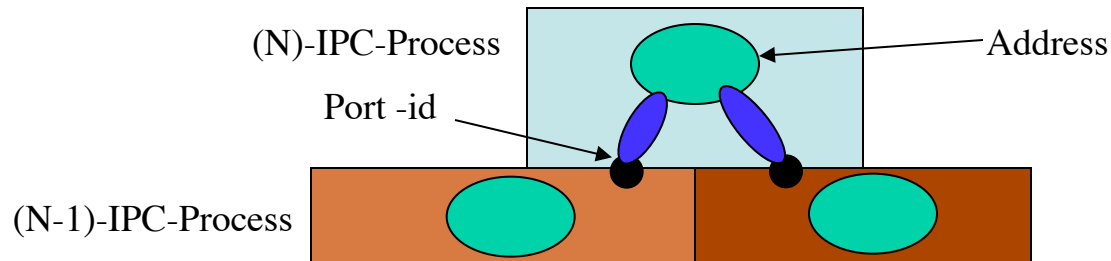
## (Routing Table Size)

	<u>Non-topological</u>	<u>Topological</u>
Metros-DIF 3	$O((2Dn)^2)$	$O((Dm)^2)$ where $n$ is the number of hosts and $m$ is the number of hosts and border routers on a single subnet.
Regionals-DIF 2	$O((2Dn_2)^2)$	$O((Dm_2)^2)$ where $n_2$ is the number of border routers around the outside and $m_2$ is number of border routers at this level on a single subnet.
Backbone-DIF 1	$O((2Dn_1)^2)$	$O((2Dn_1)^2)$ where $n_1$ is the number of border routers on the backbone. Note that $m \ll n$ .

- For the Internet  $O((6r)^2)$  where  $r$  is the number of routers in the network.

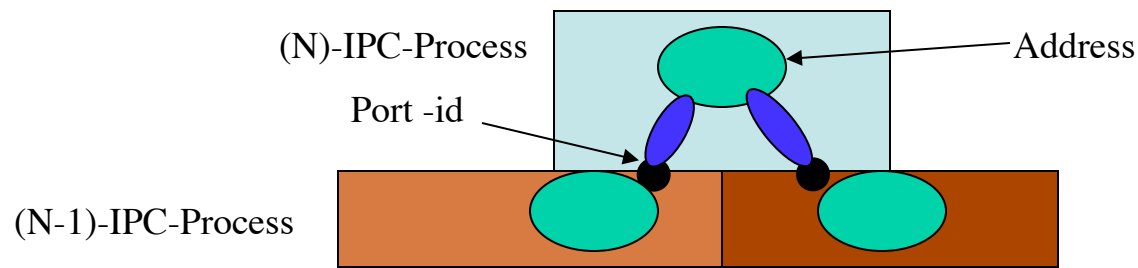
# Names & Implications of the Model

## (Basics)



- We have made a big deal of node and point of attachment, but they are relative, not absolutes.
  - An (N)-IPC-Process is a node in the (N)-DIF.
    - An (N-1)-IPC-Process is a node in the (N-1)-DIF
  - An (N-1)-IPC-Process is a point of attachment to an (N)-IPC-Process.
  - An (N)-address is a synonym for an (N)-IPC-Process.
- So as long as we keep that straight, there is no point to making the distinction.
- Note that it is the port-id that creates layer independence. With a port-id, No Protocol-Id Field is Necessary, or if there is such a field something is wrong.

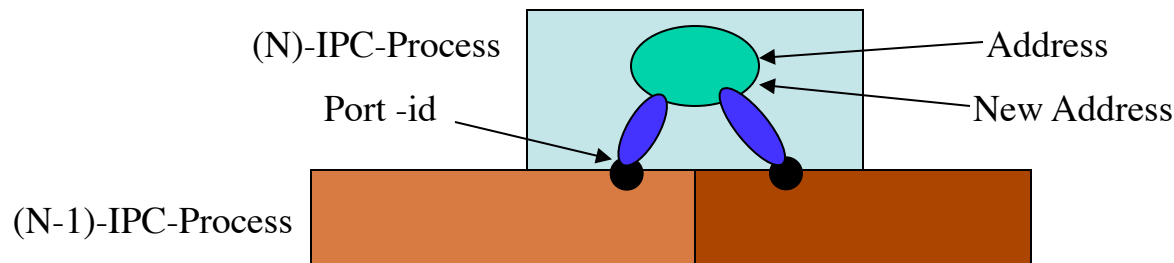
# Names & Implications of the Model (Multihoming)



- Yea, so? What is the big deal?
  - It just works
    - PDU arrives at the (N-1)-IPC Process. If the address designates this IPC Process, the CEP-id is bound to the port-id, so after stripping off (N-1)-PCI, it passes it up.
    - The process repeats. If the address in the (N)-PCI is this IPC-Process, it looks at the CEP-id and pass it up as appropriate.
    - Normal operation. Yes, the (N-1)-bindings may fail from time to time.
    - Not a big deal.

# Names & Implications of the Model

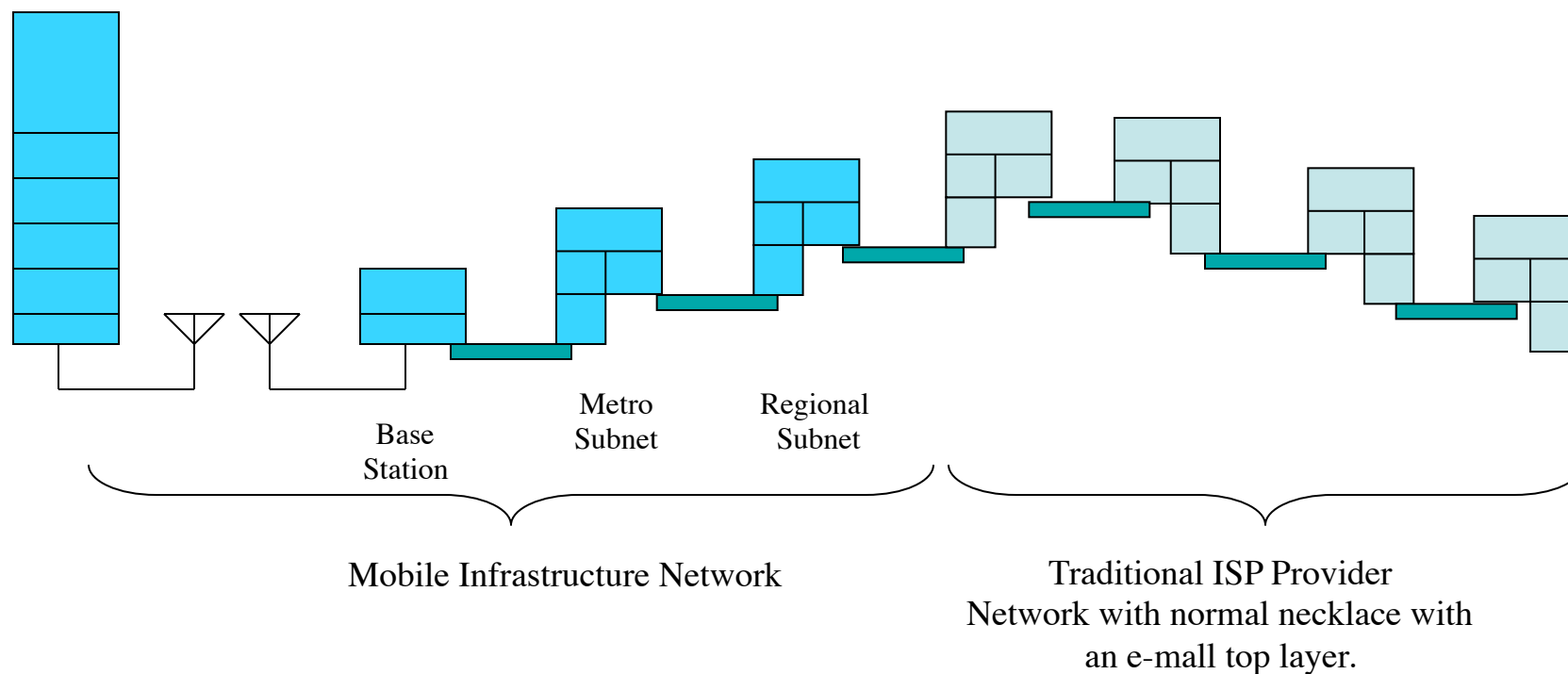
## (Mobility)



- Yea, so? What is the big deal?
  - It just works just like multihoming only the (N-1)-port-ids come and go a bit more frequently.
- O, worried about having to change address if it moves too far? Easy.
  - Assign a new synonym to it. Put it in the source address field on all outgoing traffic. Stop advertising the old address as a route to this IPC-Process.
  - Want to renumber the DIF for some reason? Same procedure.
- Again, no special cases. No special protocols. No concept of a home router. Okay, policies in the DIF may be a bit different to accommodate faster changing points of attachment, but that is it.

# The Skewed Necklace

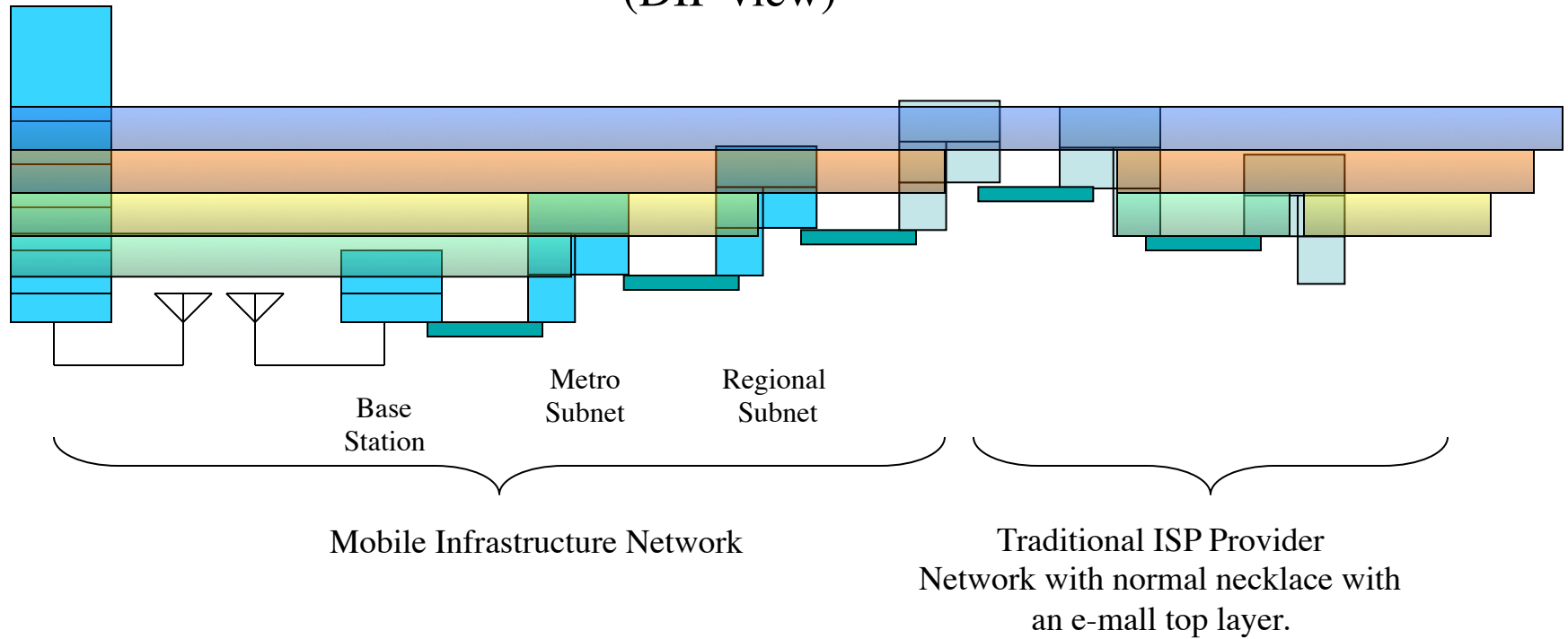
(A Typical Mobile Network)



- Space does not permit drawing networks at each layer. There would be internal routers between the border routers in a real network.
- It appears that one could make the mobile host appear stationary to the top layer, i.e. the top layer addresses don't change because all the routing is handled in the lower layers.

# The Skewed Necklace

(DIF view)



- Clearly more layers could be used to ensure the scope allows sufficient time for updating relative to the time to cross the scope of the layer.
  - Space does not permit drawing networks.
- It appears that one could make the mobile host appear stationary to the top layer, i.e. the top layer addresses don't change because all the routing is handled in the lower layers.

# What New Is Needed?

- Nothing
  - Enrollment in a new DIF follows normal procedures
  - Allocation of a flow follows normal procedures
  - Changing the address of an IPC Process with in a DIF follows the normal procedure.
  - New points of attachments, i.e. new lower layer DIFs, are acquired in the normal way.
    - There are specific cases to work out here. In general, expect that a wireless device will be probing for new PoAs. But then a system with a down wireline interface should be doing the same thing.
  - Current points of attachment are discarded when they can no longer provide an acceptable QoS (criteria and measurement is policy as it is in the wireline case).

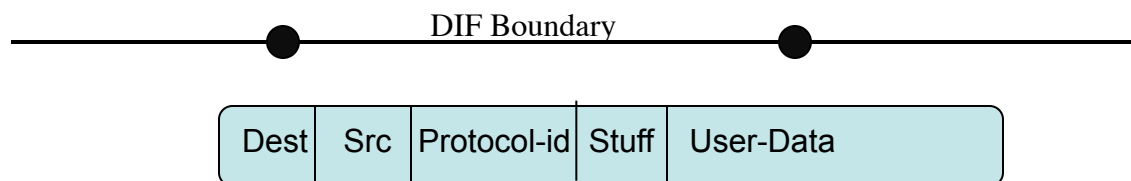


# The Other Mobile “Cases”



- 1) Purely mobile with no connection to a fixed network and no external reference.
- 2) Purely mobile with no connection to a fixed network with external reference, e.g. GPS.
  - The DIFs for these networks look like stand-alone fixed networks. Again nothing new is required. For case 1) the rate of change in position may be too great to make the assignment of topological addresses or the use of traditional routing feasible. Case 2) has other possibilities that might mitigate these constraints to some degree.
  - Conjecture: Ad hoc networks with a high rate of mobility will be limited in the size that can be reasonably sustained.
    - Note: most ad hoc networks do routing on demand.
    - Purposely embedding some slower moving systems among the fast moving could vastly improve performance.
  - There are specific cases where the nature of the problem allows assumptions to be made so that these techniques can be applied.

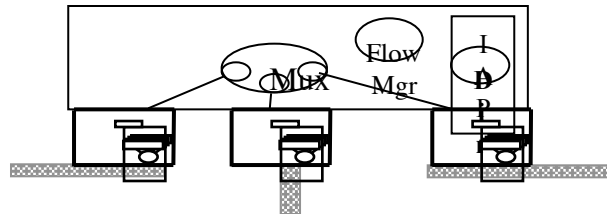
# Even the Shim DIF was Enlightening



- A Shim DIF creates the thinnest possible veneer to make a legacy layer service look like a DIF.
- Both IP and Ethernet (without LLC) have a ‘protocol-id’ field
  - Historically (1982) a problem: (N+1)-protocol identified in an (N)-protocol
- Without port-ids, there is no isolation and this implies that for each protocol type, there can only be one “user” of the “flow.”
  - There can be only one QoS-cube.
- Conclusion: Port-ids are necessary to a well-formed layer/DIF. These are ill-formed layers.
  - Ethernet with LLC is well-formed.
  - Port-ids provide isolation.

# Implications of the Model & Names (Choosing a Layer)

- In building the IPC Model, the first time there were multiple DIFs (data link layers in that case), to maintain the API a task was needed to determine which DIF to use.

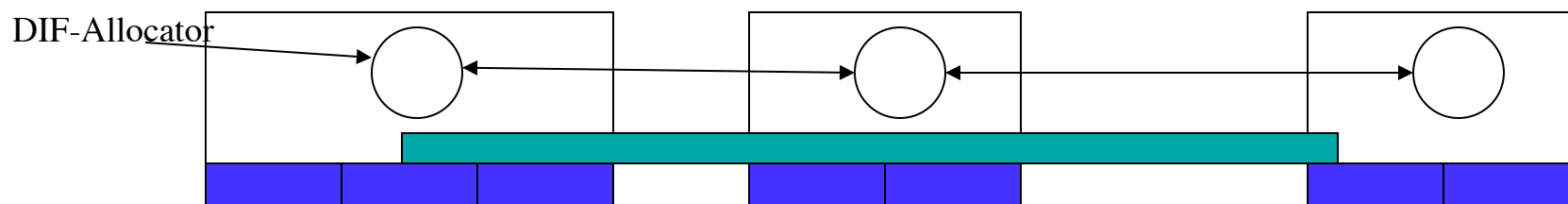


- User didn't have to see all of the wires
  - But the User shouldn't have to see all of the "Nets" either.
- This not only generalizes but has major implications.

# Implications of the Model & Names

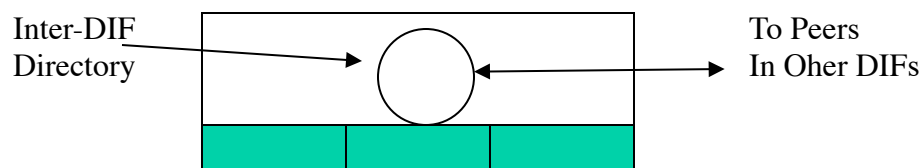
## (A DIF Allocator)

- A DIF-Allocator incorporating a Name Space Management function determines via what DIFs applications are available.
  - If this system is a not member, it either joins the DIF as before
  - or creates a new one.



- Which Implies that the largest address space has to be only large enough for the largest e-mail.
  - Given the structure, 32 or 48 bits is probably more than enough.
- You mean?
  - Right. IPv6 was a waste of time . . .
  - Twice.

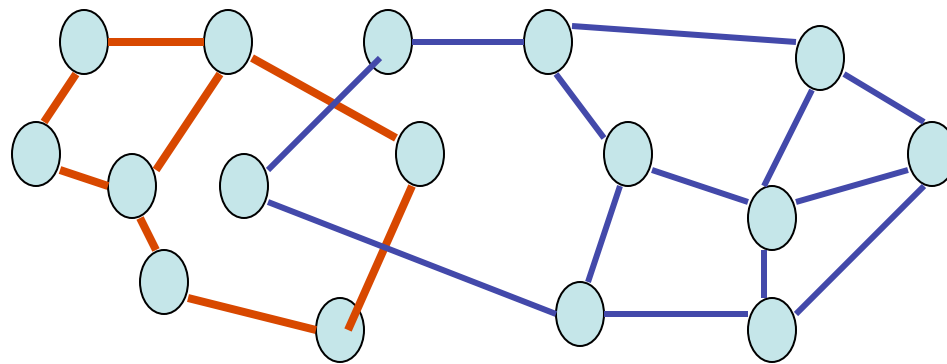
# So a Global Address Space is Not Required but Neither is a Global Application Name Space



Actually one could still have distinct names spaces within a DIFs (synonyms) with its own directory database.

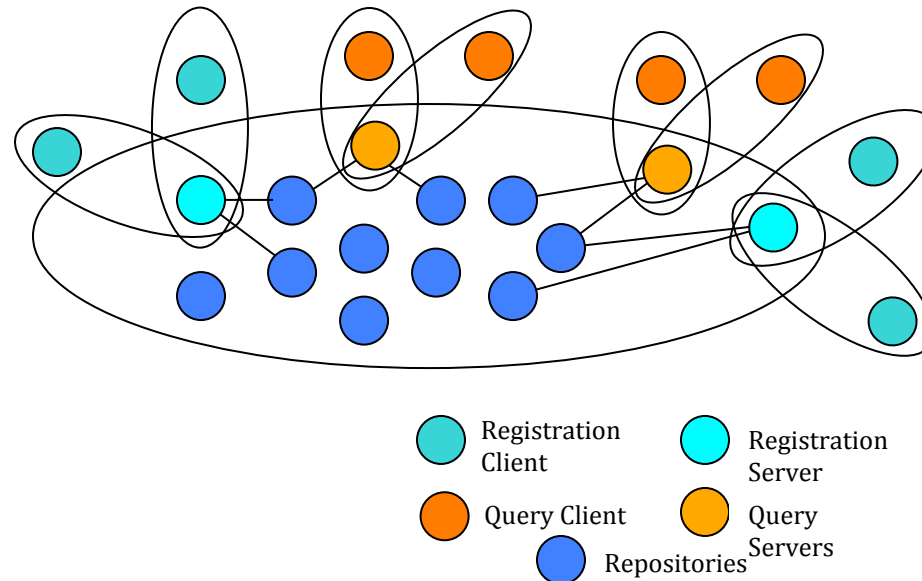
- Not all names need be in one Global Directory.
- Coexisting application name spaces and directory of distributed databases are not only possible, but useful.
- Needless to say, a global name space can be useful, but not a requirement imposed by the architecture.
- The scope of the name space is defined by the chain of databases that point to each other.

## Scope is Determined by the Chain of Places to Look



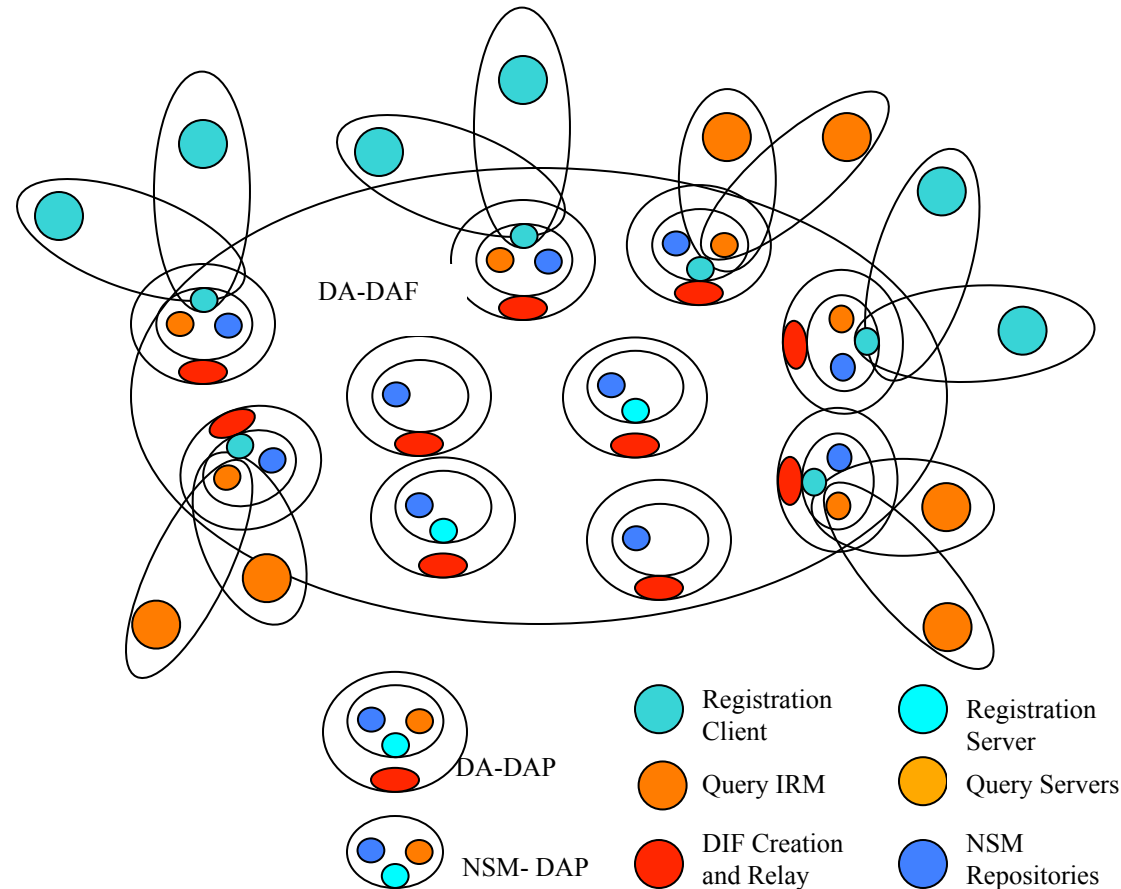
- The chain of databases to look for names determines the scope of the name space.
  - Here there are 2 non-intersecting chains of systems, that could be using the same wires, but would be entirely oblivious to the other.

# Name Space Management



- There is a common functional required to manage name spaces. It handles registration, query, and delegation of names and is incorporated found in both DIFs and DAFs.
- There are two major uses: the DIF-Allocator and the Flow Allocator, as well as in many distributed database applications, etc.

# DIF Allocator

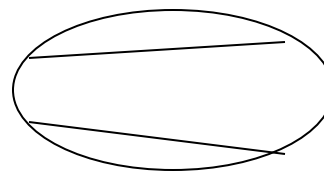
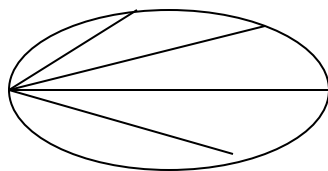


- A DIF Allocator is very similar to the Flow Allocator, both contain a NSM function for application names and addresses, respectively and operate at different granularities creating either DIFs or connections.



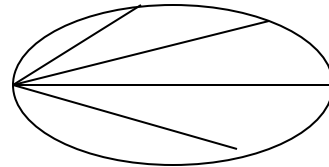
# Multicast and Anycast are Simpler Too

- Generalized to Whatevercast:
  - A set and a rule that returns as many members of the set that satisfy the rule.
- Unicast becomes a degenerate case of whatevercast.
  - Forwarding table entry maps Destination Address to a list of next hops. For unicast, the list has one element.
- Primarily handled by hosts or border routers, where all whatevercast traffic is either:
  - On this subnet (only do spanning tree within subnet if there is a lot) or
  - Transit to another subnet, (both cases degenerate to point-to-point).



- So we see Whatevercast devolves into Unicast.

# Multicast and Anycast are Simpler Too



- Information in topological addresses imply an approximate spanning tree, which can be used to identify the border routers. Thus, in most cases obviating the need for a separate spanning tree (multicast) routing algorithm.
- And also making it straightforward to multiplex whatevercasts with common sub-trees which will allow even greater efficiency.
  - Note that the common sub-trees do not have to be strict sub-trees but simply have a reasonable degree of commonality to be worthwhile.

# Like I Said

What's All the Fuss? ;-)

# Questions?