HOW SOFTWARE ARCHITECTURE CAN MAKE AN APPLICATION-FRIENDLY INTERNET

Pamela Zave

AT&T Laboratories—Research

Florham Park, New Jersey, USA

ABOUT AT&T RESEARCH (www.research.att.com)

- we are a direct descendant of Bell Labs
- about 200 researchers
- many of us live in New York City and travel to the lab by train
- we hire new Ph.D.s every year

HOW SOFTWARE ARCHITECTURE CAN MAKE AN APPLICATION-FRIENDLY INTERNET: OUTLINE

1 THE STATE OF INTERNET ARCHITECTURE

2 IDEAS THAT MAKE PROGRESS POSSIBLE

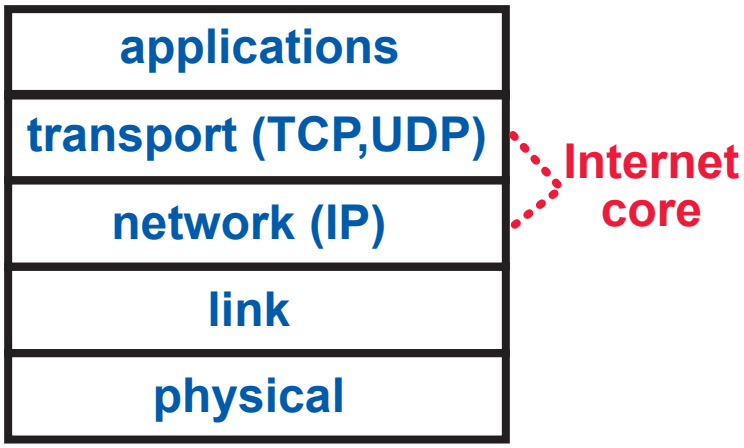
3 EXAMPLE: MOBILITY

4 A RESEARCH AGENDA

THE "CLASSIC" INTERNET ARCHITECTURE

THE LAYERS

Each layer is universal and has a different function.



THE PHILOSOPHY

[Clark 88]

- cooperation among trusted parties (no security)
- best-effort service (no reliability or performance guarantees)
- end-to-end transparency (no built-in servers)
- designed to empower users and encourage innovation

despite its limitations

(or maybe because of them and the resulting simplicity),

this architecture has succeeded beyond anyone's wildest dreams

THE AGE OF THE WEB (1993 - NOW)

Firewalls and Network Address Translation (NAT) boxes provide security and address-space expansion.

default firewall allows no initiation of communication from this side, even if desirable

machines here have no persistent public addresses; NAT boxes play with addresses and ports

NAT boxes and firewalls . . .

. . . are so tightly intertwined with TCP and UDP,

. . . and vary so much across the Internet,

that unless they *know about an application*, it is unlikely to work.

lack of separation of concerns

in other words, only applications built on HTTP work reliably

people speak of HTTP as the new Internet core

[Popa et al. 10]

THE STATE OF INTERNET EVOLUTION

[Handley 06]

INTERNET "OSSIFICATION"

- there has been no important change in the transport layer (TCP/ UDP) since 1988
- there has been no important change in the network layer (IP) since 1993

" . . . technologies get deployed in the core of the Internet

when they solve an immediate problem

or when money can be made"

- it is very difficult for an Internet service provider to make money with improvements, because most have no effect until everyone else adopts them
- a crisis is the only way to get the global consensus required for real change

THE AGE OF DIVERSITY (NOW - FUTURE)

TO MEET THE NEEDS OF SOCIETY, THE FUTURE INTERNET MUST SUPPORT A MUCH WIDER RANGE OF . . .

. . . APPLICATIONS

- will replace current data, telecommunications, and broadcast networks
- real-time, peer-to-peer, and end-to-end transparent applications, not just Web applications

. . . STAKEHOLDERS

- all segments of society are stakeholders, their interests must be balanced

[Clark et al. 05]

. . . RESOURCES

- many kinds of wireless networks
- delay-tolerant networks ("sneaker nets") incorporate very slow, very diverse link technologies

. . . COMMUNICATION FUNCTIONS

- many aspects of privacy and security
- support for mobility, multihoming, anycast, customized routing, *etc.*

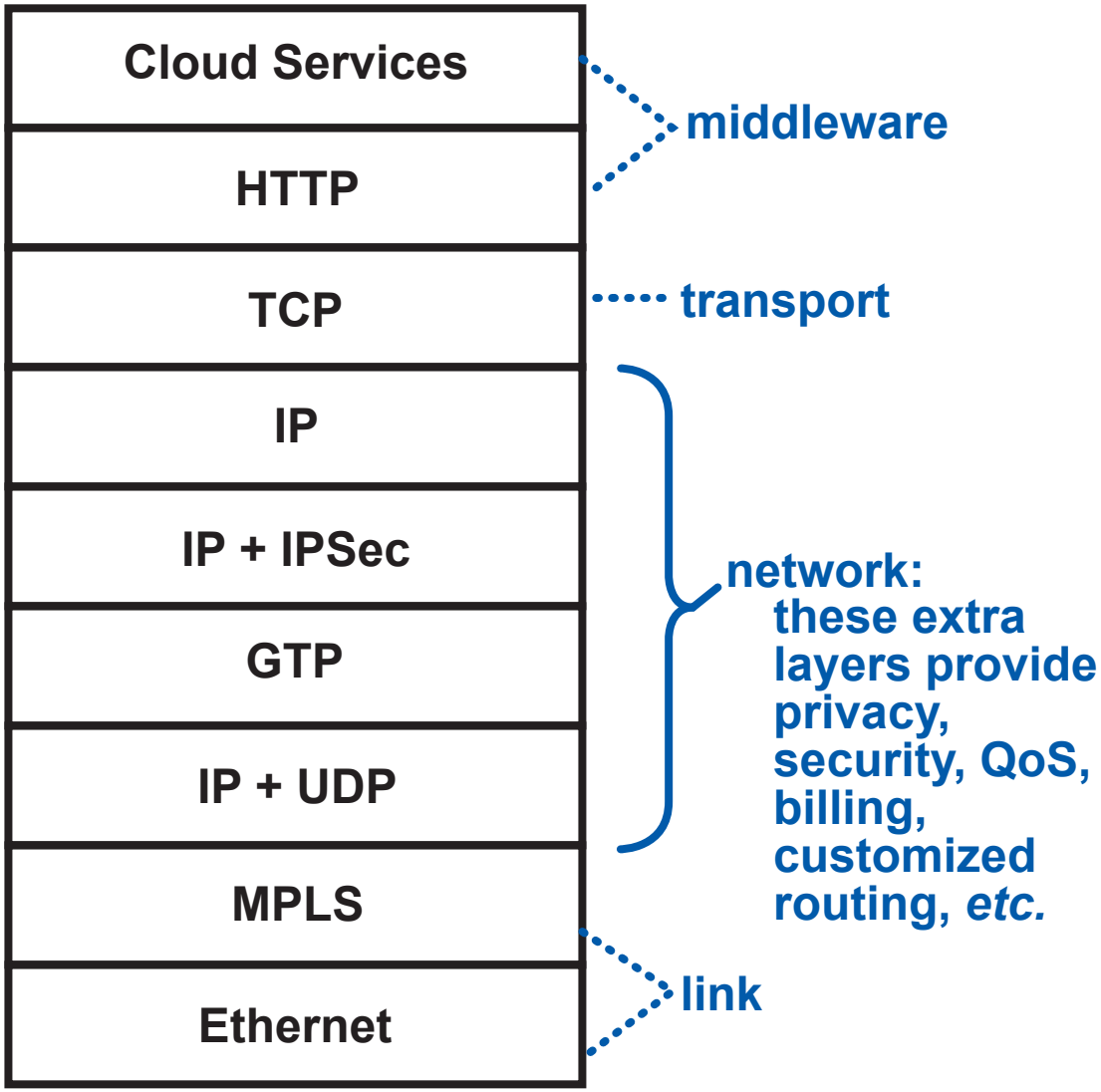
. . . POLICIES

- selective reliability guarantees
- selective quality-of-service (QoS) guarantees
- selective privacy and security guarantees

the classic architecture and the Age of the Web have created a one-size-fits-all Internet that cannot provide the needed diversity

WHEN THE NEED FOR DIVERSITY MEETS OSSIFICATION

a typical real packet shows (through headers) the use of these layers (*simplified picture*):



THIS COMPLEXITY MAKES IT DIFFICULT TO MANAGE RESOURCES WELL:

- 15 or more load-balancing and routing algorithms apply to this packet
- each algorithm has different goals, and each has been analyzed mostly in isolation

[Spatscheck 10]

- unanticipated effects of buffering on TCP are causing widespread performance problems (high latency and jitter)

[Gettys 11]

THE COMPLEXITY CHALLENGE

the future Internet must support
much greater diversity, at much greater scale

HOW ARE WE DOING WITH RESOURCE MANAGEMENT?

Not so well.

We have composition of too many layers, and too little ability to predict how they behave when composed.

HOW ARE WE DOING WITH APPLICATIONS?

Also not well.

Because of the pervasive lack of separation of concerns, . . .

. . . a mechanism to solve a problem usually works only in a specialized environment, and

. . . mechanisms to solve problems tend to break each other.

As a result, it is much too difficult to build, deploy, and maintain applications.

HOW SOFTWARE ARCHITECTURE CAN MAKE AN APPLICATION-FRIENDLY INTERNET: OUTLINE

1 THE STATE OF INTERNET ARCHITECTURE

2 IDEAS THAT MAKE PROGRESS POSSIBLE

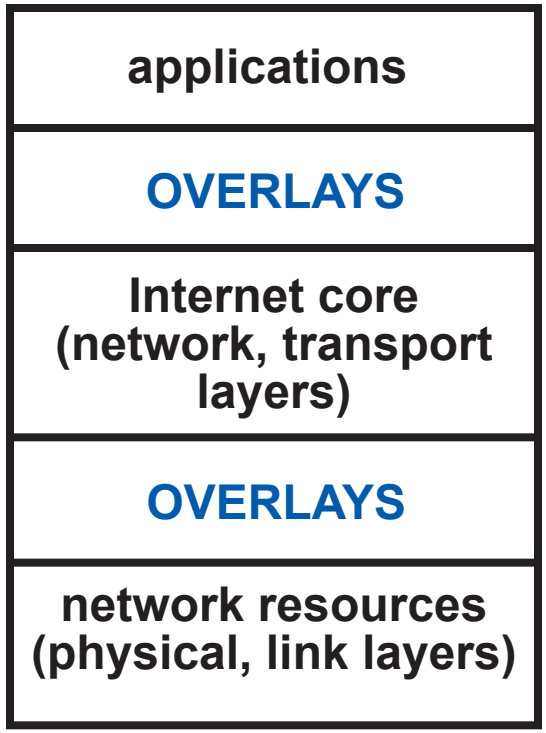
3 EXAMPLE: MOBILITY

4 A RESEARCH AGENDA

EVOLUTION THROUGH LAYERS AND VIRTUALIZATION

LAYERS ARE THE MODULES OF NETWORK ARCHITECTURE

we have seen that custom overlays are used frequently to modify an ossified Internet architecture

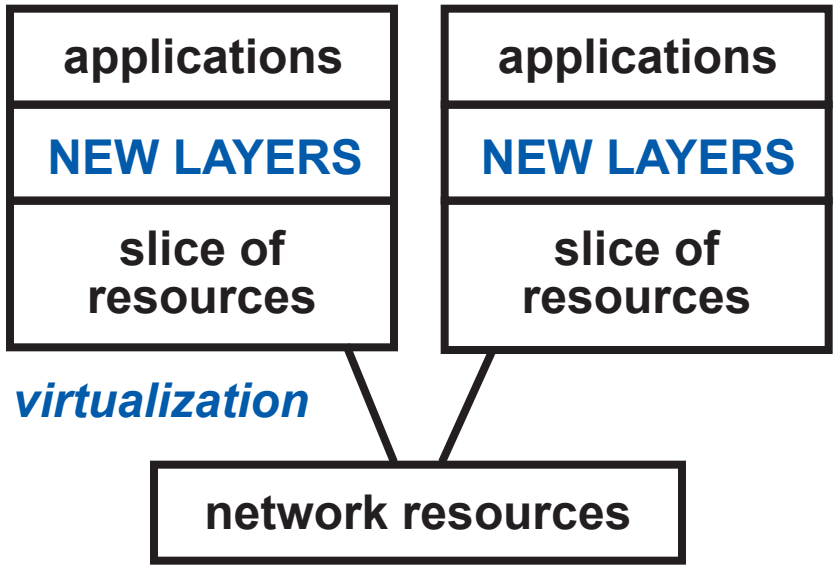


A NEW LAYER IS ALSO A "CLEAN SLATE" FOR DESIGN, . . .

. . . AND ITS VALUE IS ENHANCED BY VIRTUALIZATION

can experiment safely with new architectural ideas

can get closer to the resources than Internet overlays can



in the end, there may be no universal Internet layers [Roscoe 2006]

WHAT IS A LAYER?

Using Day's definition of a layer *[Day 08]*, each layer contains all the basic mechanisms of networking.

The definition is a template that can be instantiated differently for different *purposes, scopes,* and *levels.*

The definition makes clear how overlays compose in a hierarchy and what their relationships are.

THESE ARE NOT THE SAME AS CLASSIC INTERNET LAYERS:

All layers have the same functions (in the most abstract sense) rather than having different functions in different layers.

THEY ARE WHAT WE NEED!

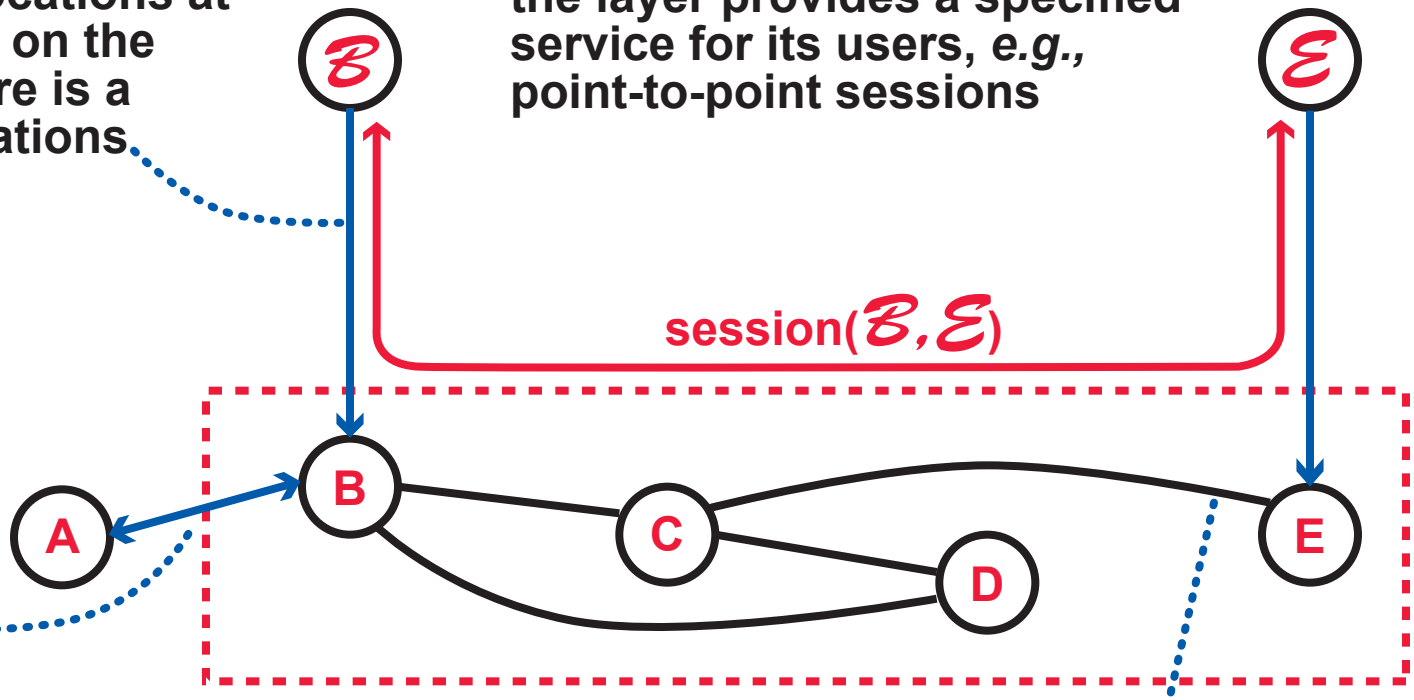
- deeply rooted in the history and practice of networking
- draws the module boundary in exactly the right place
- provides for diversity within a common structure

DAY'S DEFINITION OF A LAYER

Registration:
user processes in a higher layer can register their locations at member processes on the same machine; there is a directory of registrations

Communication Service:
the layer provides a specified service for its users, *e.g.*, point-to-point sessions

Membership:
the members are processes; each has a unique and persistent name from the name space; enrollment protocol accepts and names new members

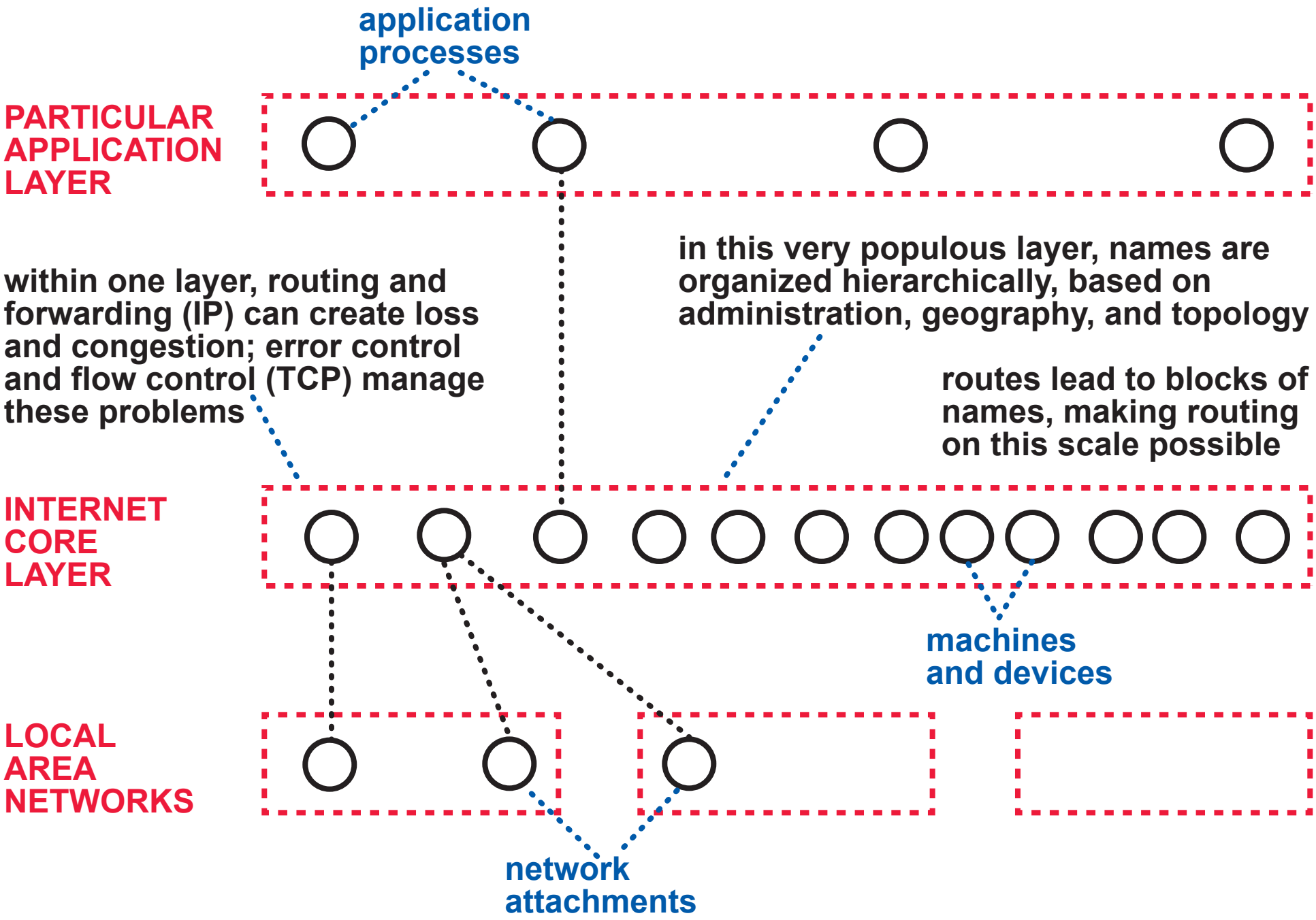


Routing:
any member can reach any other through a path in the layer; routing protocol spreads knowledge of links and paths; forwarding protocol uses path knowledge

Links:
there is a link between two member processes if both are registered in the same lower layer

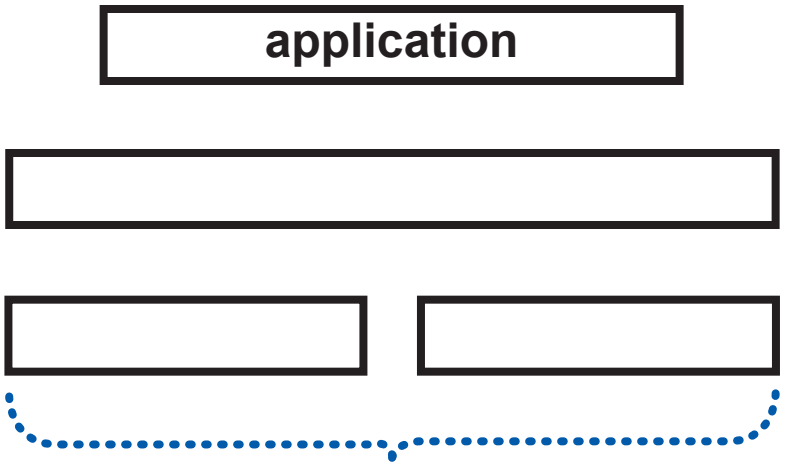
THERE IS SECURITY AND RESOURCE MANAGEMENT THROUGHOUT

THE CLASSIC ARCHITECTURE ACCORDING TO DAY



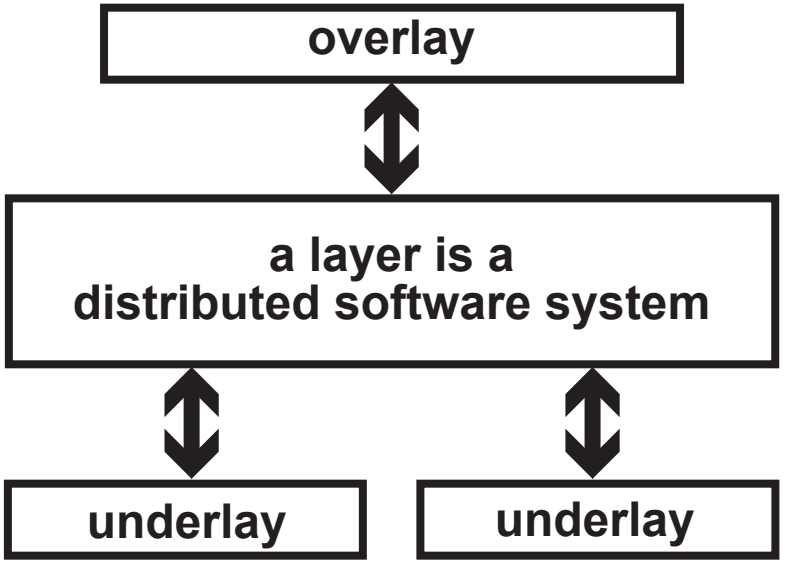
TO ACHIEVE DIVERSITY AND MANAGE COMPLEXITY

EACH APPLICATION SHOULD RUN ON EXACTLY THE RIGHT STACK OF LAYERS . . .



. . . BUT THIS WILL REQUIRE A LARGE NUMBER OF CUSTOMIZED LAYERS!

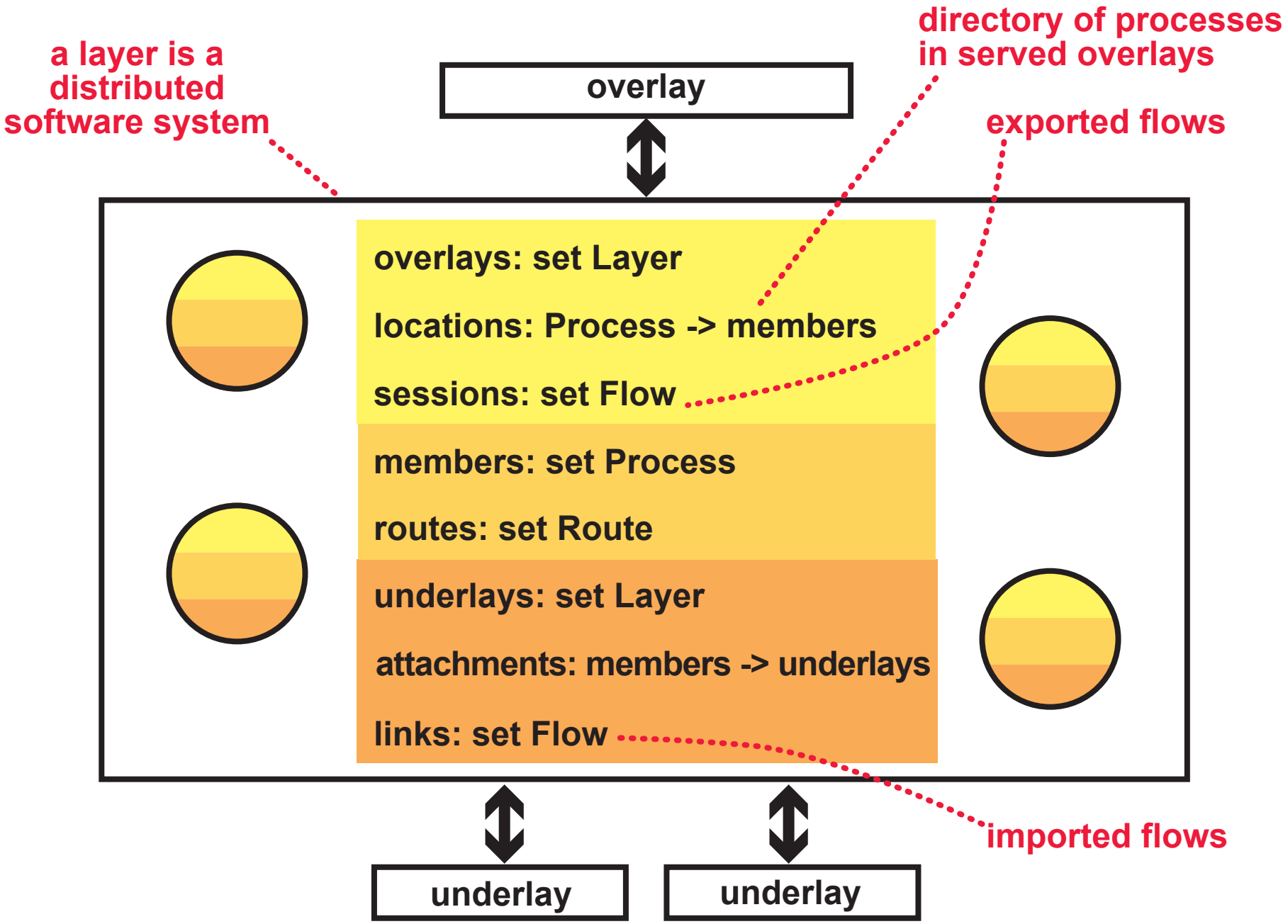
THE KEY IS UNDERSTANDING THE SOFTWARE ARCHITECTURE OF LAYERS . . .



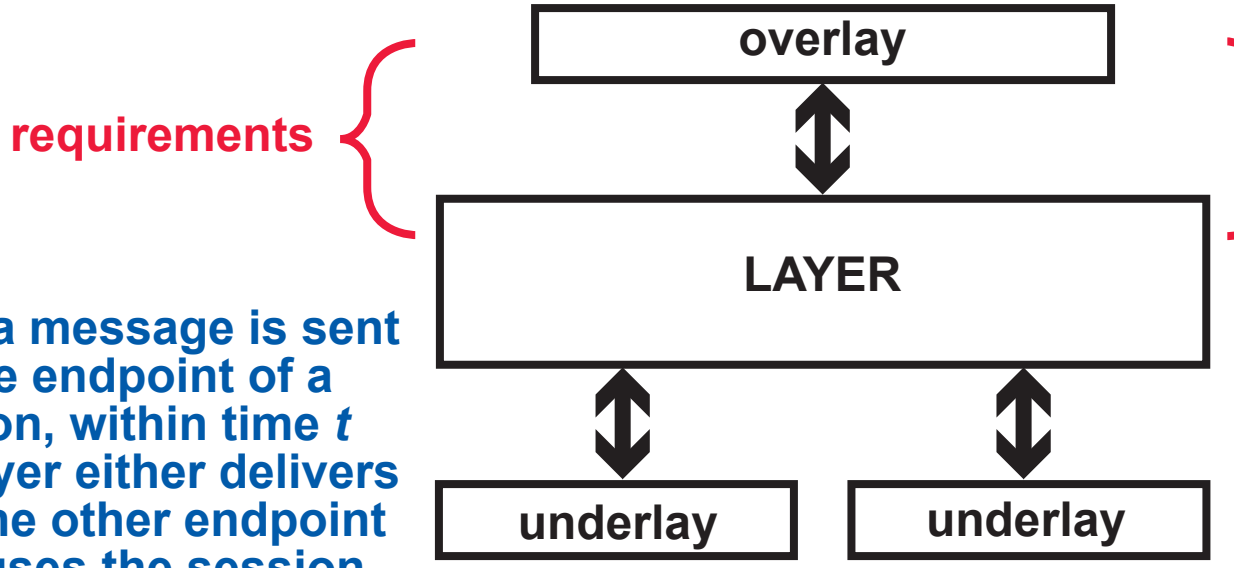
. . . WELL ENOUGH SO THAT WE CAN GENERATE THE NECESSARY (CORRECT, PREDICTABLE) SOFTWARE

[Loo et al. 05]

THE BEGINNING OF A LAYER ARCHITECTURE



TO UNDERSTAND THE ARCHITECTURE OF LAYERS



e.g.,
once a message is sent by one endpoint of a session, within time *t* the layer either delivers it to the other endpoint or causes the session to fail

e.g.,
while an overlay process is an endpoint of a session, it does not change locations in the layer

e.g.,
while a layer process is active in a session, the underlay does not terminate its attachment

FOR EACH COMMUNICATION FUNCTION (set of related requirements):

1. understand the range of requirements and assumptions
2. understand the "design space" of implementations, including applicability and properties
3. understand how to map each design into the layer architecture, retaining separation of concerns so that other functions will map without interference
4. understand the principles for allocating instances of functions to layers

HOW SOFTWARE ARCHITECTURE CAN MAKE AN APPLICATION-FRIENDLY INTERNET: OUTLINE

1 THE STATE OF INTERNET ARCHITECTURE

2 IDEAS THAT MAKE PROGRESS POSSIBLE

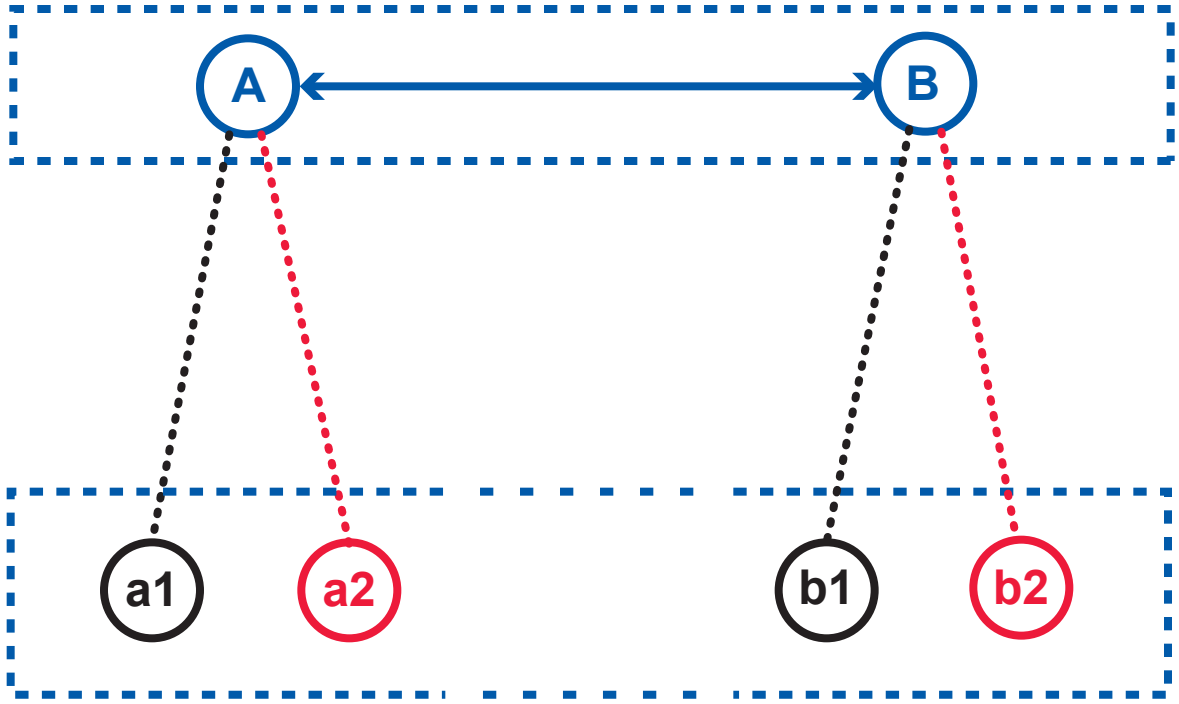
3 EXAMPLE: MOBILITY

4 A RESEARCH AGENDA

WHAT IS MOBILITY?

During the lifetime of a process, its registration in a lower level may change.

Mobility is the communication function that maintains the process's inter-level relationships despite the change.

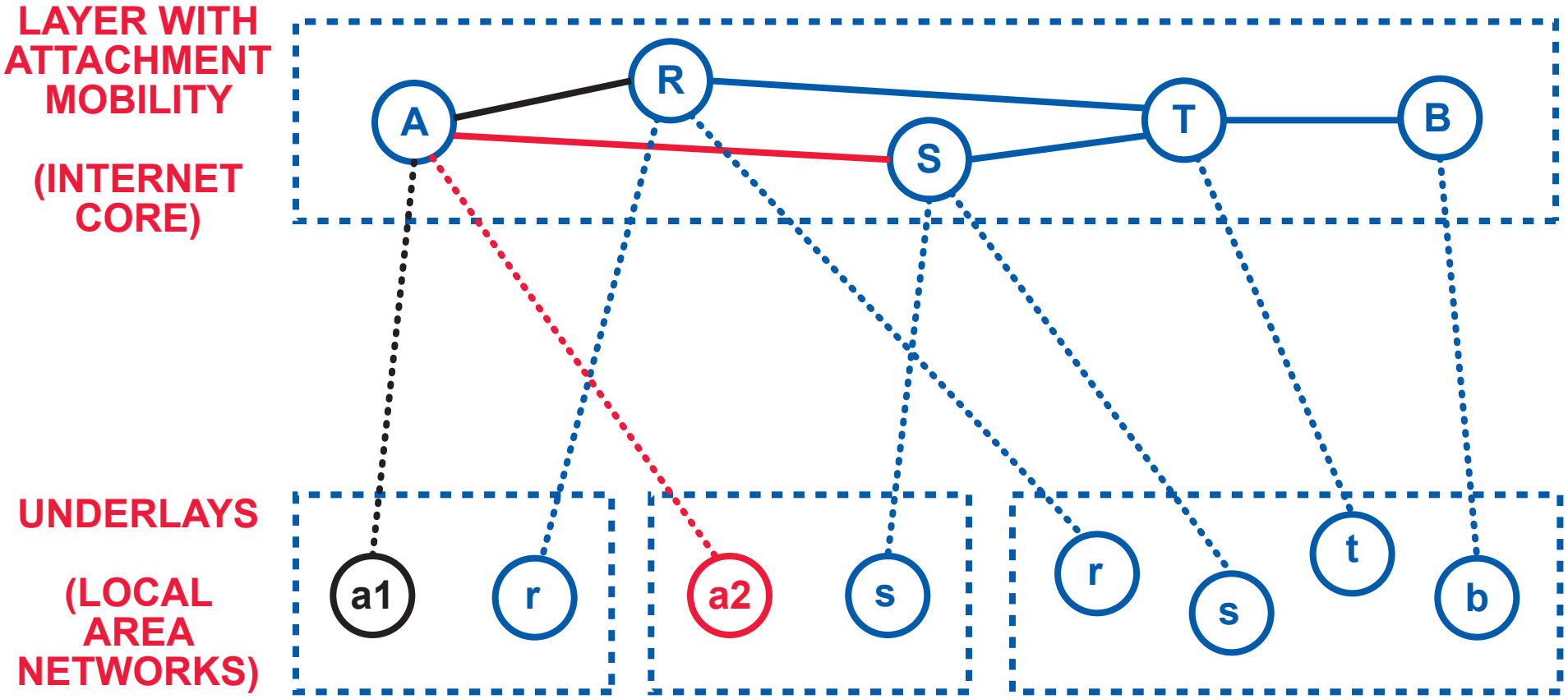


WHY MOBILITY FIRST?

Because mobility touches on many of the most central concepts of networking, such as naming and routing, and is very difficult with the classic Internet architecture.

ATTACHMENT MOBILITY

this layer must maintain what
it is getting from the underlays,
which is the reachability of A



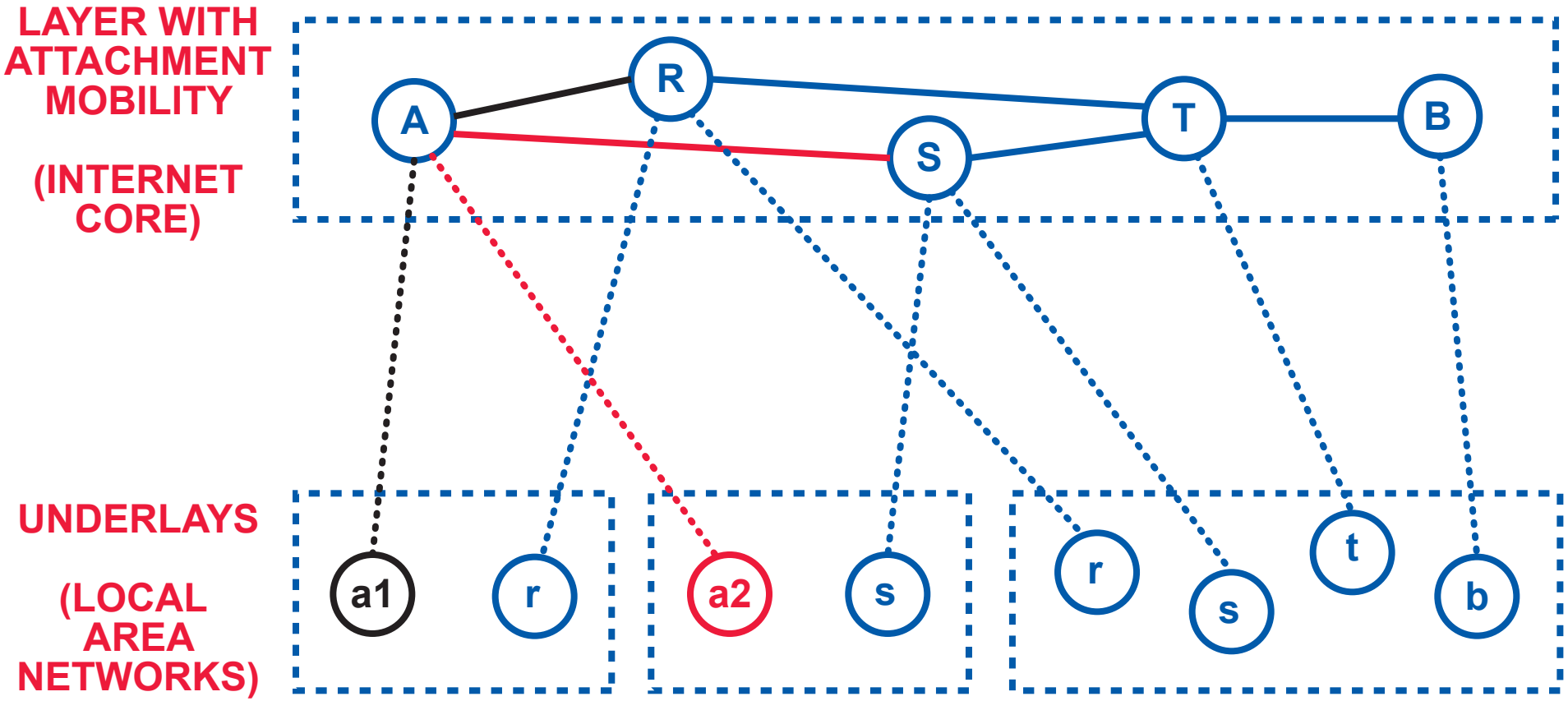
a mobile process (here A) is moving physically, moves beyond the scope of current LAN

in the old LAN, its process becomes disconnected and its attachment is terminated (violating an assumption)

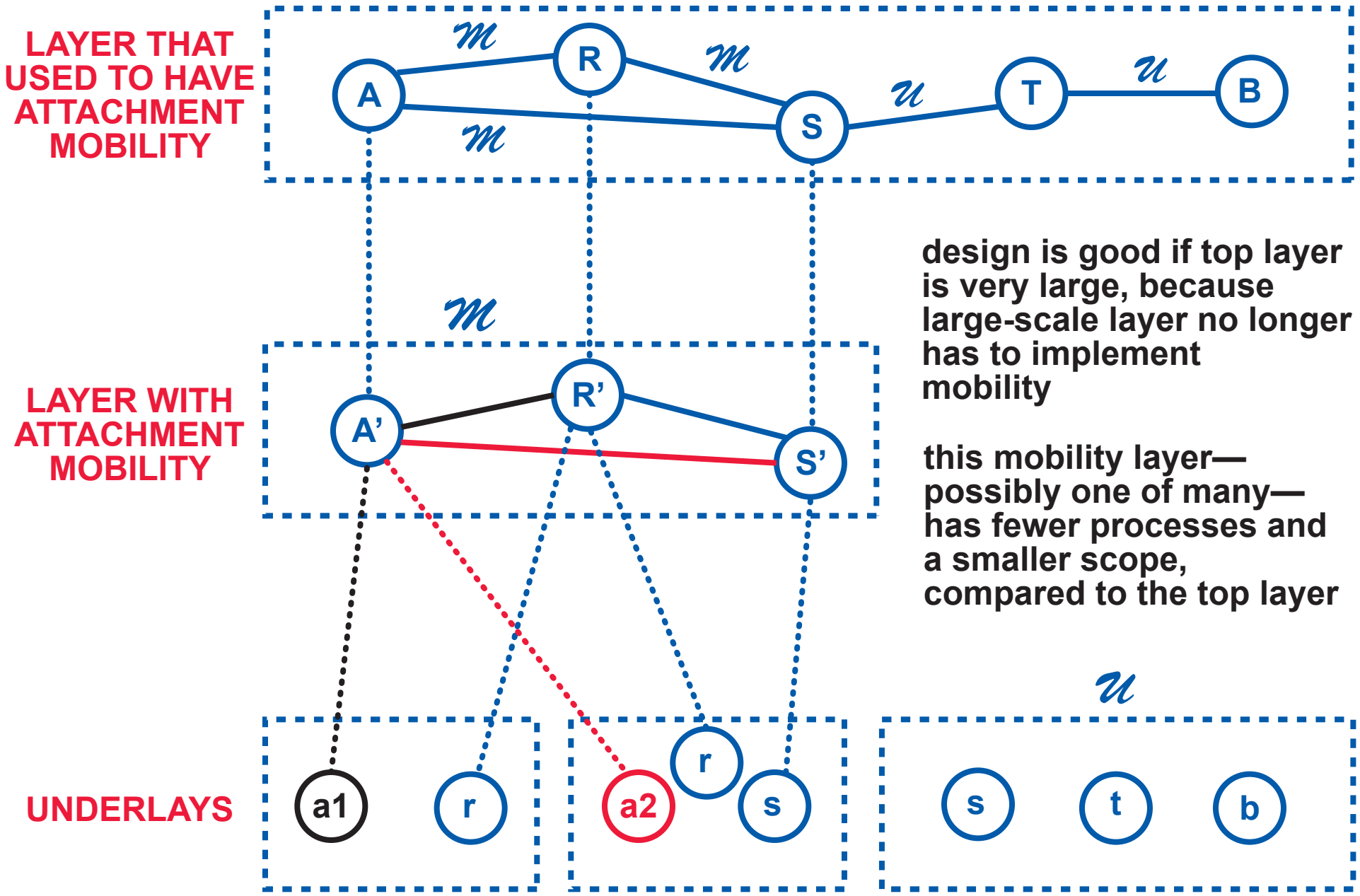
the mobile process registers with a new LAN

ATTACHMENT MOBILITY: THE PRIMARY DESIGN

as A's attachments change, its links change; layer must track these changes and update routes to maintain the reachability of A



ATTACHMENT MOBILITY: ANOTHER DESIGN



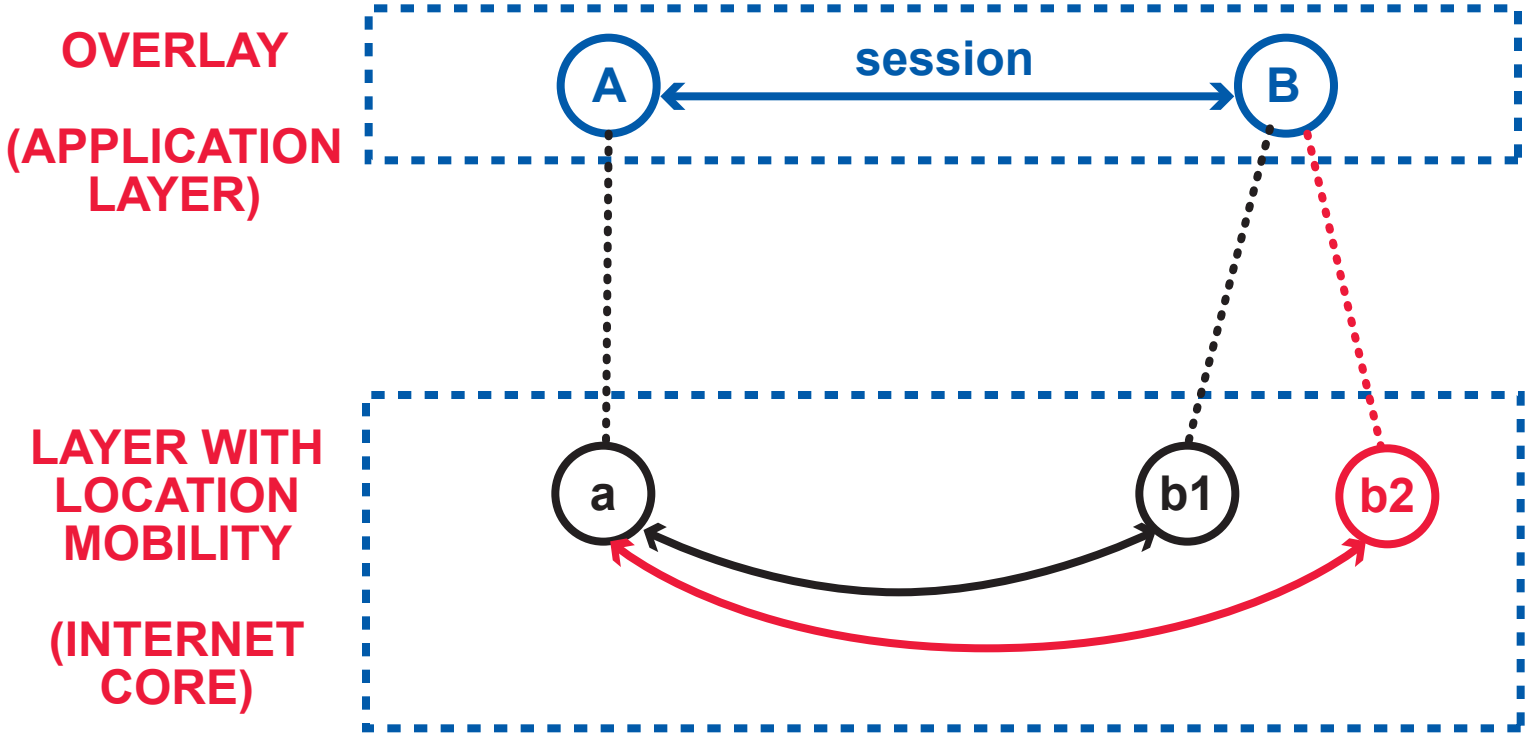
design is good if top layer is very large, because large-scale layer no longer has to implement mobility

this mobility layer—possibly one of many—has fewer processes and a smaller scope, compared to the top layer

LOCATION MOBILITY

a mobile process (here B) changes its location in the layer while it is a session endpoint, violating an assumption

this might happen because B is a process migrating from server b1 to server b2

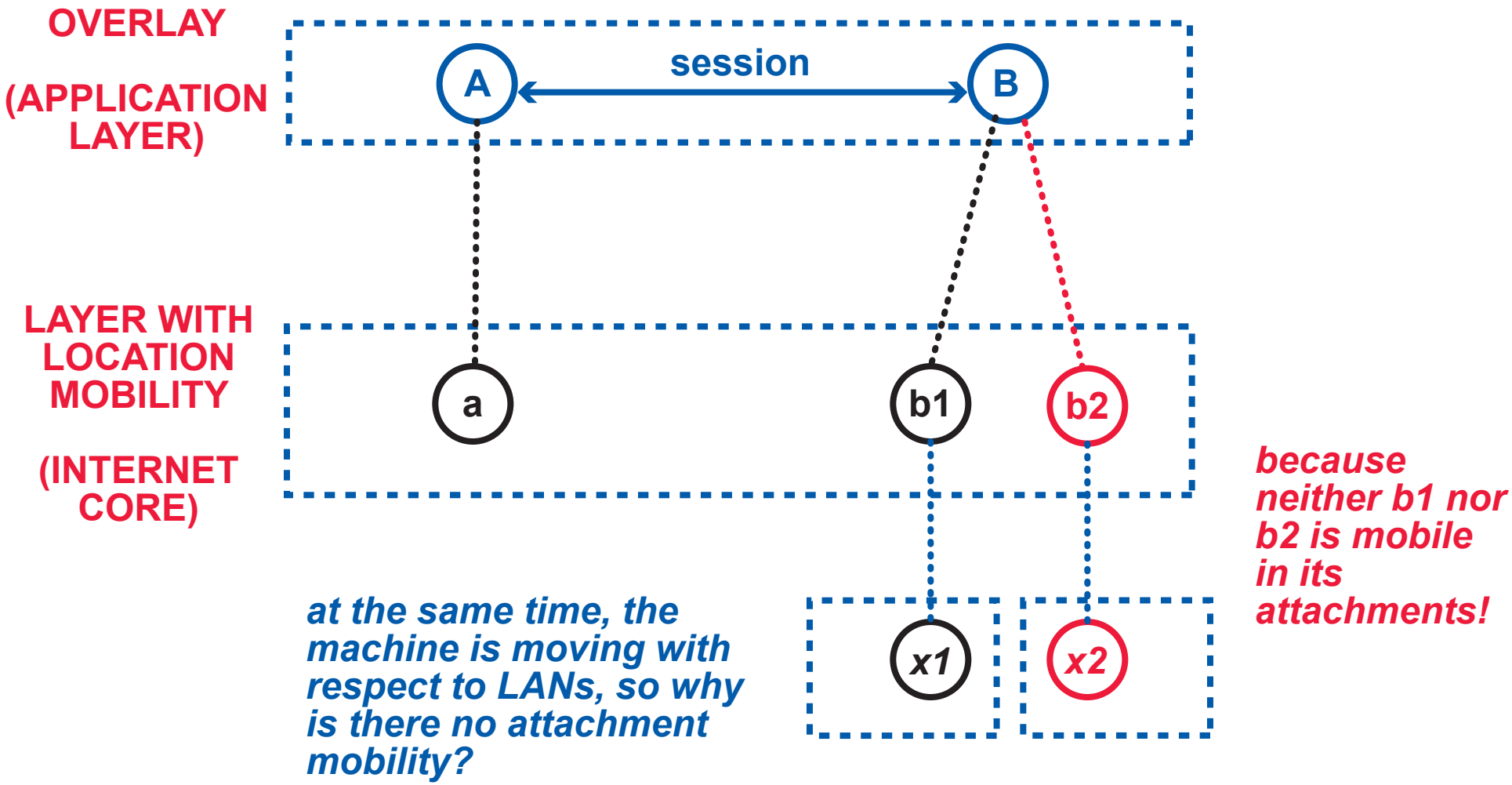


this layer must maintain what it is giving to the overlay, which is the session between A and B

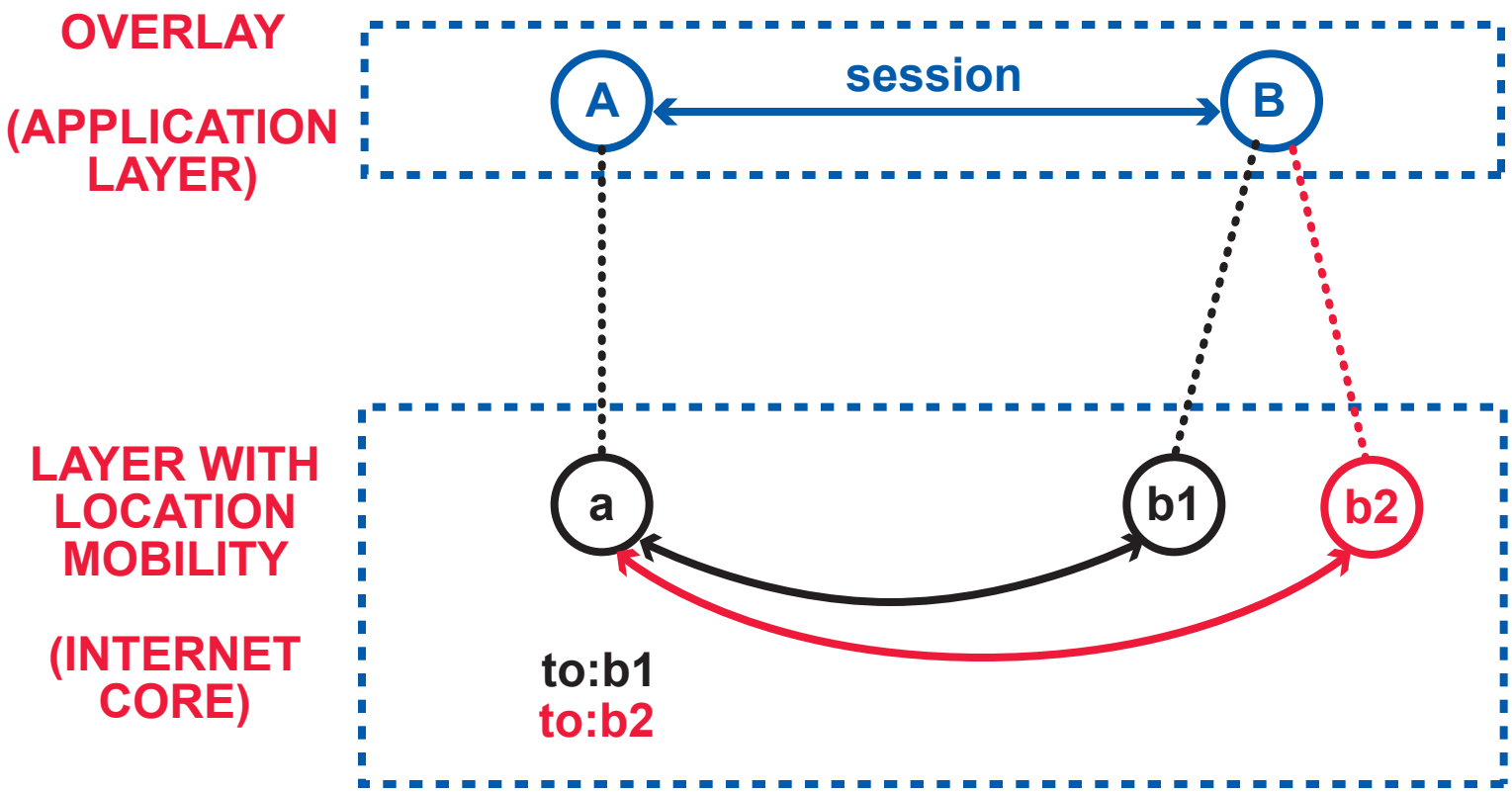
LOCATION MOBILITY, CONTINUED

a mobile process (here B) changes its location in the layer while it is a session endpoint, violating an assumption

this might also happen because B's machine has moved outside the area where it can have location-dependent name b1—the process that represents it must die and be reborn with name b2



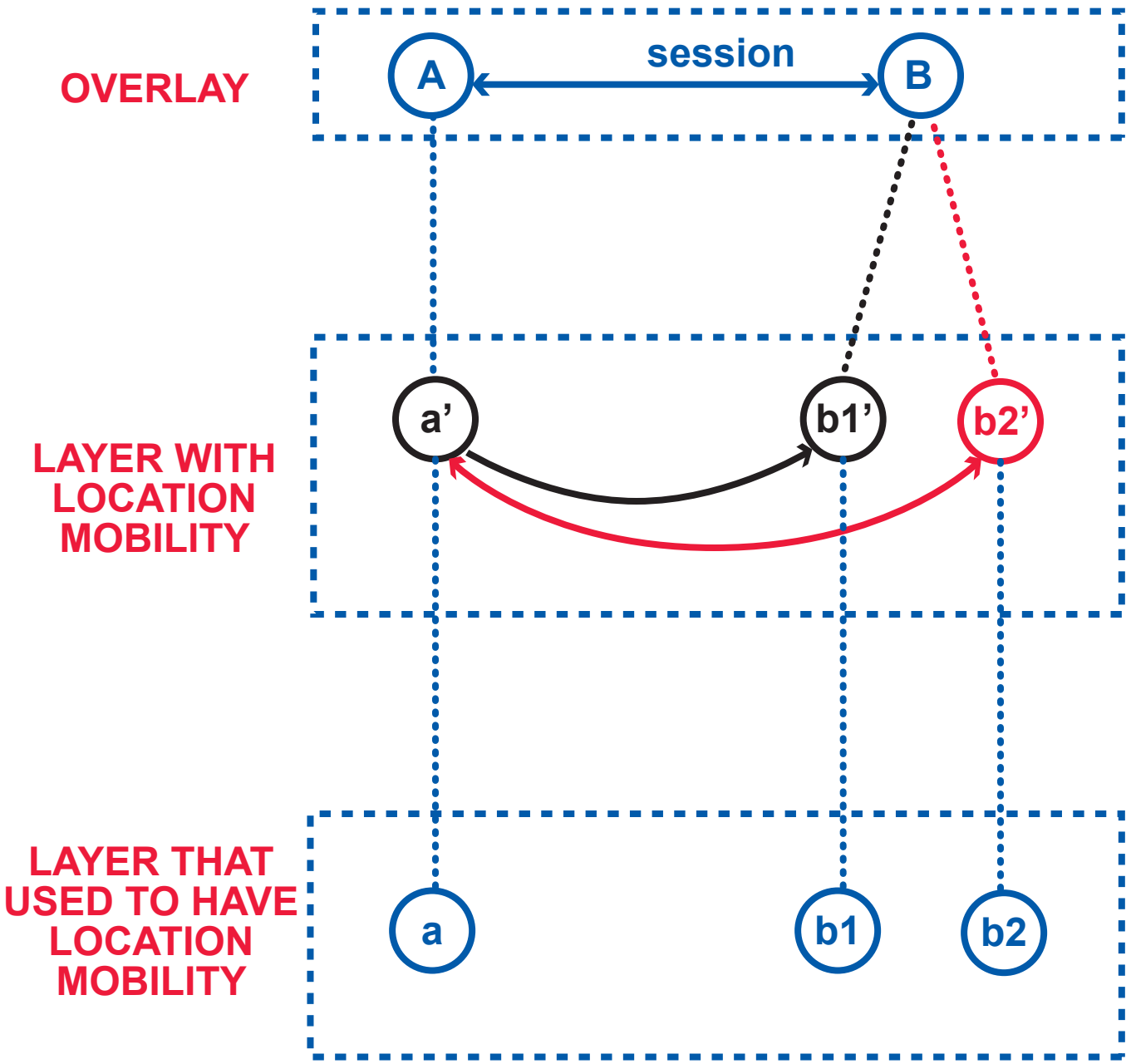
LOCATION MOBILITY: THE PRIMARY DESIGN



initially, a and b1 cache each other's names for this session

when B moves from b1 to b2, both the location directory and a's session state must be updated

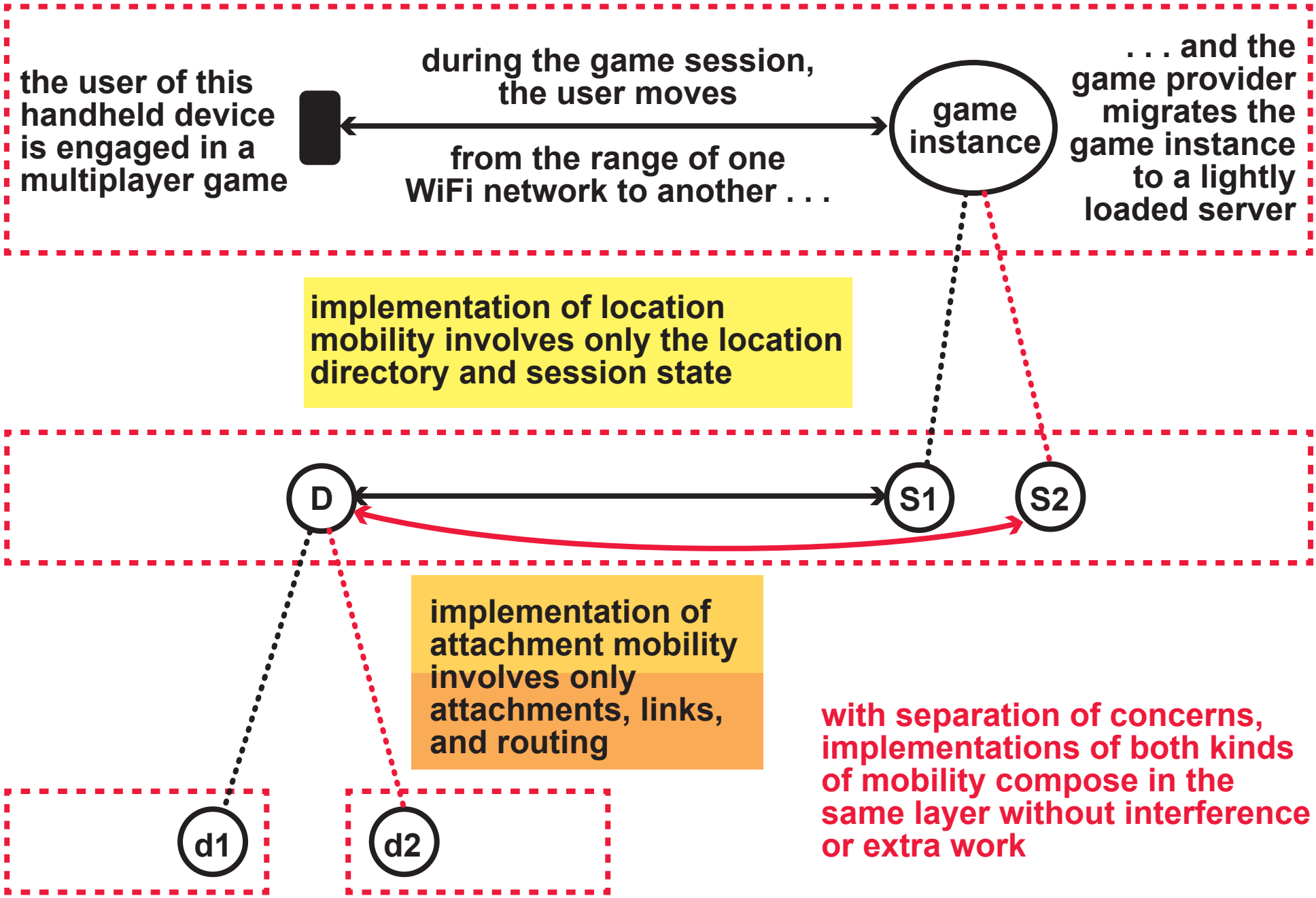
LOCATION MOBILITY: ANOTHER DESIGN



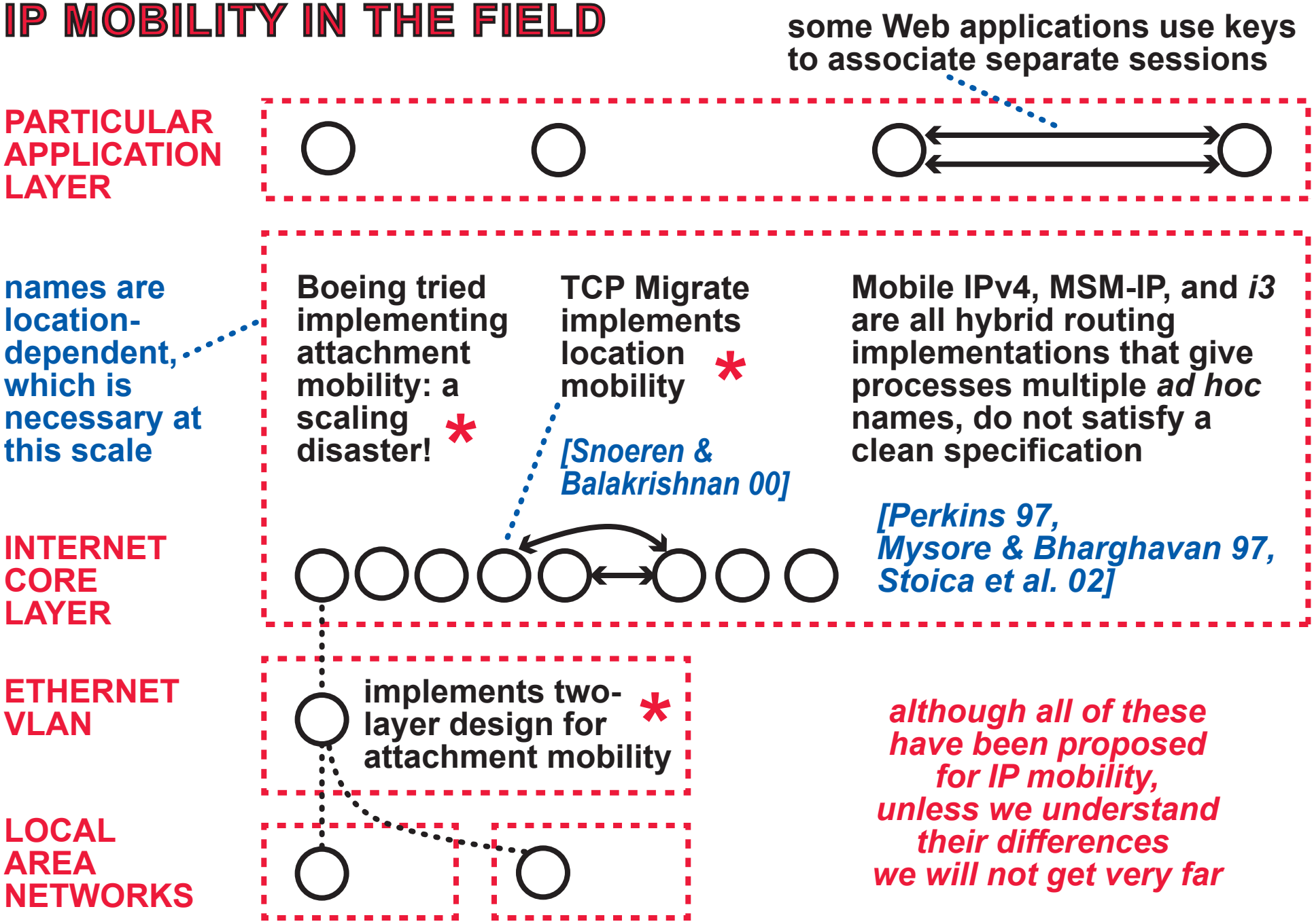
this mobility layer—possibly one of many—need serve only one overlay

design is good if bottom layer is very large, because large-scale layer no longer has to implement location mobility

COMPOSITION AND SEPARATION OF CONCERNS



IP MOBILITY IN THE FIELD



* in design space because they meet all criteria, including usability, composability

HOW SOFTWARE ARCHITECTURE CAN MAKE AN APPLICATION-FRIENDLY INTERNET: OUTLINE

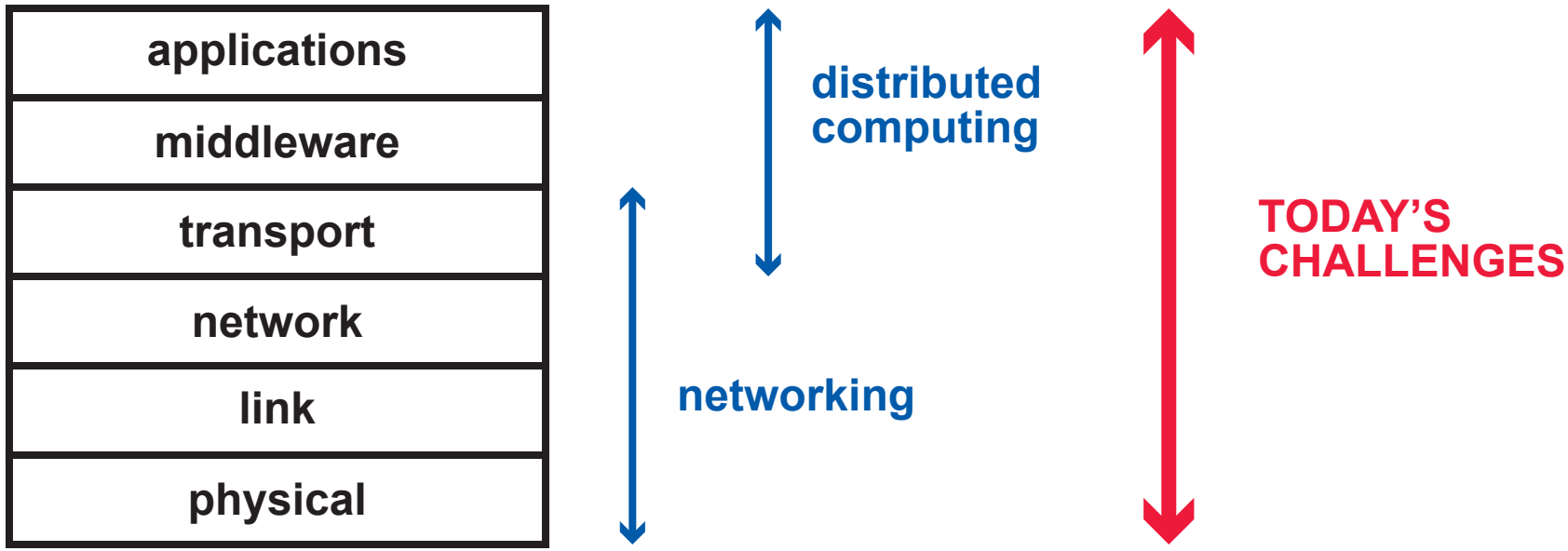
1 THE STATE OF INTERNET ARCHITECTURE

2 IDEAS THAT MAKE PROGRESS POSSIBLE

3 EXAMPLE: MOBILITY

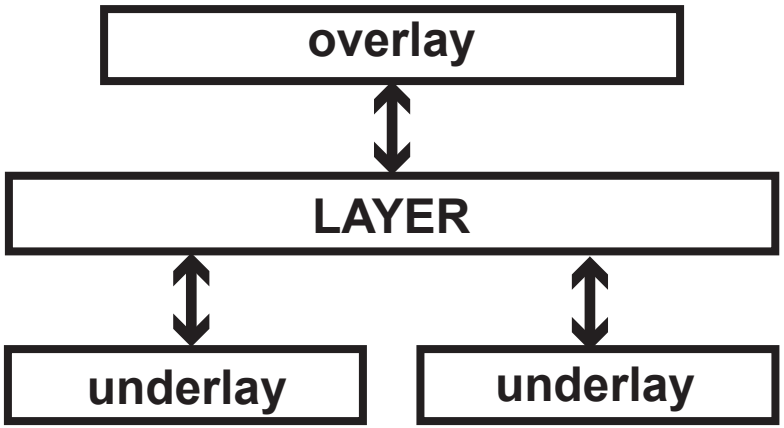
4 A RESEARCH AGENDA

A RESEARCH AGENDA IN WHICH FIELD?



what I like best
about Day's layer definition
is that it organizes networking concepts
so that software experts
can grasp their significance

UNDERSTANDING THE ARCHITECTURE OF LAYERS



FOR EACH COMMUNICATION FUNCTION (set of related requirements):

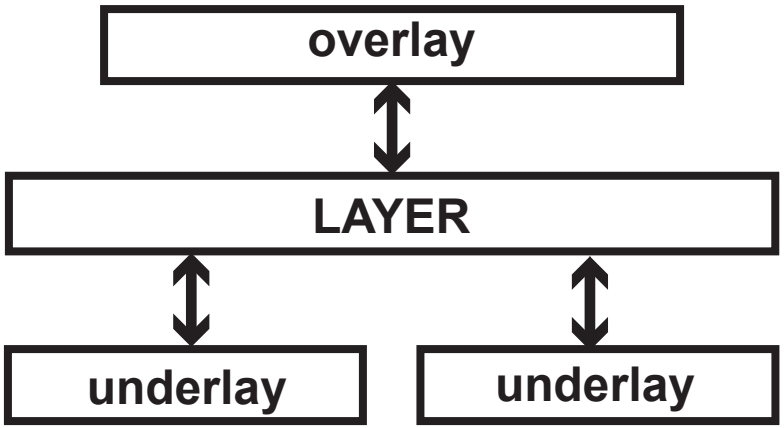
1. understand requirements, assumptions
2. understand the "design space" of implementations, including applicability and properties
3. understand how to map each design into the layer architecture, retaining separation of concerns so that other functions will map without interference
4. understand the principles for allocating instances of functions to layers

PROGRESS SO FAR

FOR MOBILITY (at least as understood in the networking community):

[Zave & Rexford 11]

UNDERSTANDING THE ARCHITECTURE OF LAYERS



FOR EACH COMMUNICATION FUNCTION (set of related requirements):

1. understand requirements, assumptions
2. understand the "design space" of implementations, including applicability and properties
3. understand how to map each design into the layer architecture, retaining separation of concerns so that other functions will map without interference
4. understand the principles for allocating instances of functions to layers

WORK TO BE DONE:

EVERYTHING ELSE!

FUNCTIONS RELATED TO MOBILITY

- multihoming
- anycast

FUNCTIONS THAT SEEM INDEPENDENT OF MOBILITY

- above all, security

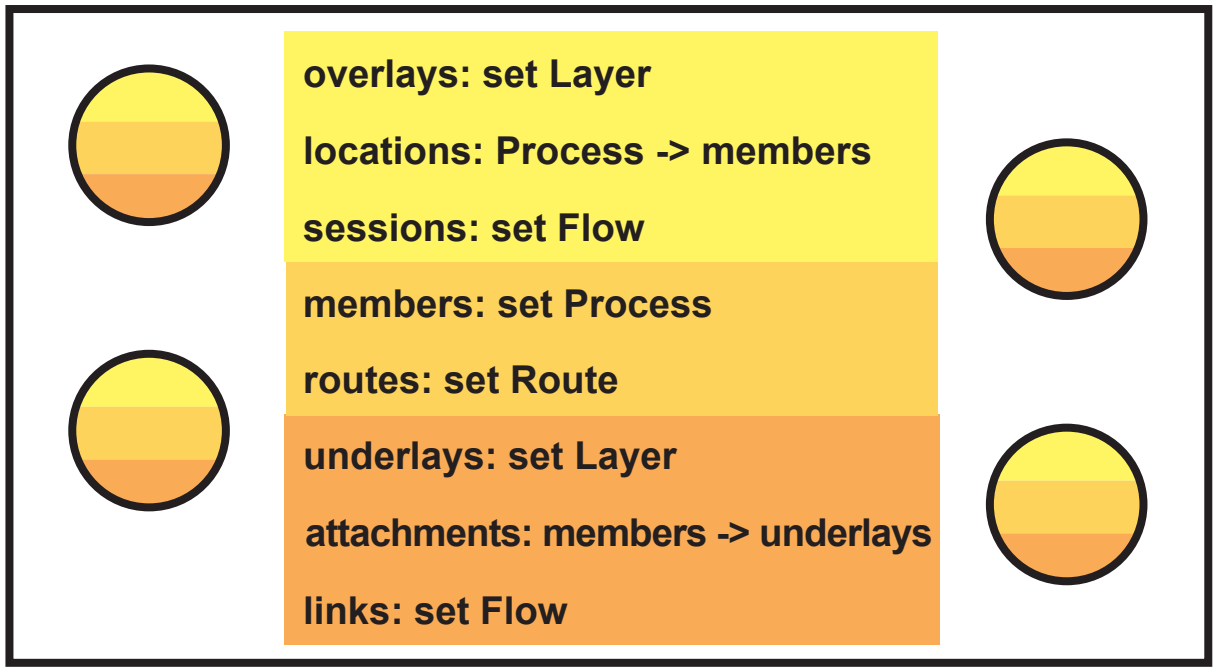
however, the distinction between "mobile computing" and "mobile computation" from security-oriented Ambients seems related to attachment mobility vs. location mobility!

[Cardelli 11]

WORK TO BE DONE ON SOFTWARE ARCHITECTURE

THIS PICTURE SHOWS SOME OF THE STATE THAT MUST BE KEPT IN A LAYER

omitting security, resource management, and no doubt many other necessities



A LAYER IS A DISTRIBUTED SOFTWARE SYSTEM—WHAT IS ITS ARCHITECTURE, OR RANGE THEREOF?

The answer should make it possible to . . .

. . . meet needs for performance, reliability, and efficiency

. . . implement various functions such as different kinds of mobility and security without interference, *i.e.,* with separation of concerns

. . . support automated generation of customized layer software

REFERENCES

[Cardelli 11] **Luca Cardelli, Mobile computational ambients,** `http://lucacardelli.name/Ambients.html`**, 2011.**

[Clark 88] **David D. Clark, The design philosophy of the DARPA Internet protocols, *Proc. SIGCOMM*, 1988.**

[Clark et al. 05] **David D. Clark *et al.,* Tussle in cyberspace: Defining tomorrow's Internet, *IEEE/ACM Trans. on Networking,* June 2005.**

[Day 08] **John Day, *Patterns in Network Architecture*, Prentice Hall, 2008.**

[Gettys 11] **Jim Gettys, Bufferbloat FAQ,** `http://gettys.wordpress.com/bufferbloat-faq`**, 2011.**

[Handley 06] **Mark Handley, Why the Internet only just works, *BT Technology Journal,* July 2006.**

[Loo et al. 05] **Boon Thau Loo *et al.,* Implementing declarative overlays, *Proc. SOSP,* 2005.**

[Mysore & Bharghavan 97] **J. Mysore and V. Bharghavan, A new multicasting-based architecture for Internet host mobility, *Proc. MOBICOM*, 1997.**

REFERENCES, CONTINUED

[Perkins 97] C. E. Perkins, Mobile IP, *IEEE Communications,* May 1997.

[Popa et al. 10] Lucian Popa *et al.,* HTTP as the narrow waist of the future Internet, *Proc. HotNets IX,* 2010.

[Roscoe 06] Timothy Roscoe, The end of Internet architecture, *Proc. Hotnets V,* 2006.

[Snoeren & Balakrishnan 00] Alex C. Snoeren and Hari Balakrishnan, An end-to-end approach to host mobility, *Proc. MOBICOM,* 2000.

[Spatscheck 10] Oliver Spatscheck, Cloud computing and my worries about the network that enables it, workshop presentation, 2010.

[Stoica et al. 02] I. Stoica *et al.,* Internet indirection infrastructure, *Proc. SIGCOMM,* 2002.

[Zave 10] Pamela Zave, Internet evolution and the role of software engineering, *Proc. Symp. on the Future of Software Engineering, 2010.*

[Zave & Rexford 11] Pamela Zave and Jennifer Rexford, The design space of network mobility, 2011.

slides will be posted by 28 June 2011 at `www2.research.att.com/~pamela`