e“speak.

The universal language of e-services

Alan H. Karp
Chief Scientist
Open Services Operation
Hewlett-Packard
http://www.hp.com/go/espeak

The Essential Difference

Hardware + Software

Tell the computer how to do the job

Services

Tell the computer what job you want done

What is E-speak?

E-speak is an open services platform for the

- creation,
- composition,
- mediation,
- virtualization,
- management, and
- accessing

of Internet-based services.

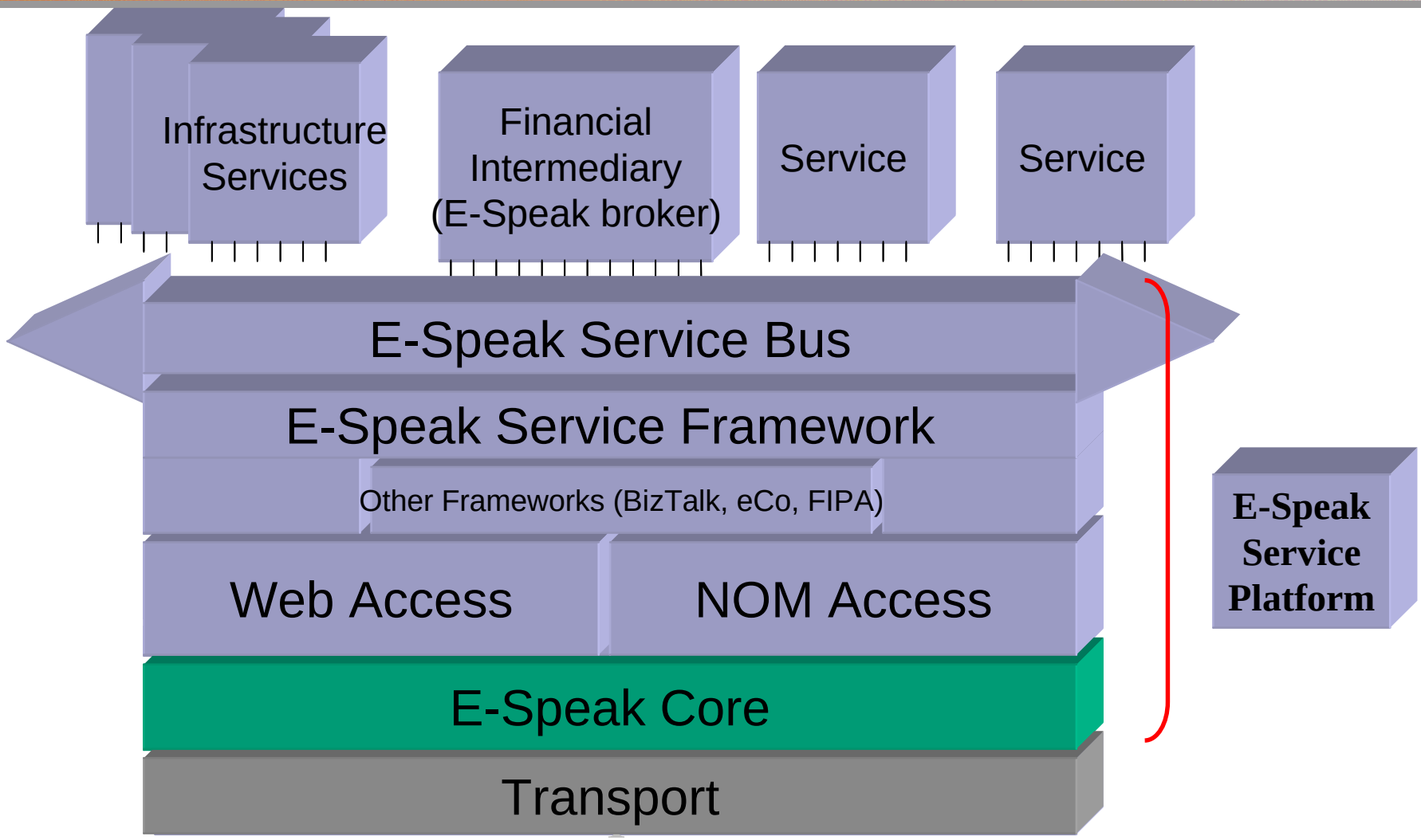
Technology Goal

Do for services what the Web has done for data

Make it as simple, in fact simpler and safer to create, compose, deploy, manage, personalize, and access services as it is to publish and access data on the Web.

Service: anything that can be transmitted digitally, including access to the communication channel itself.

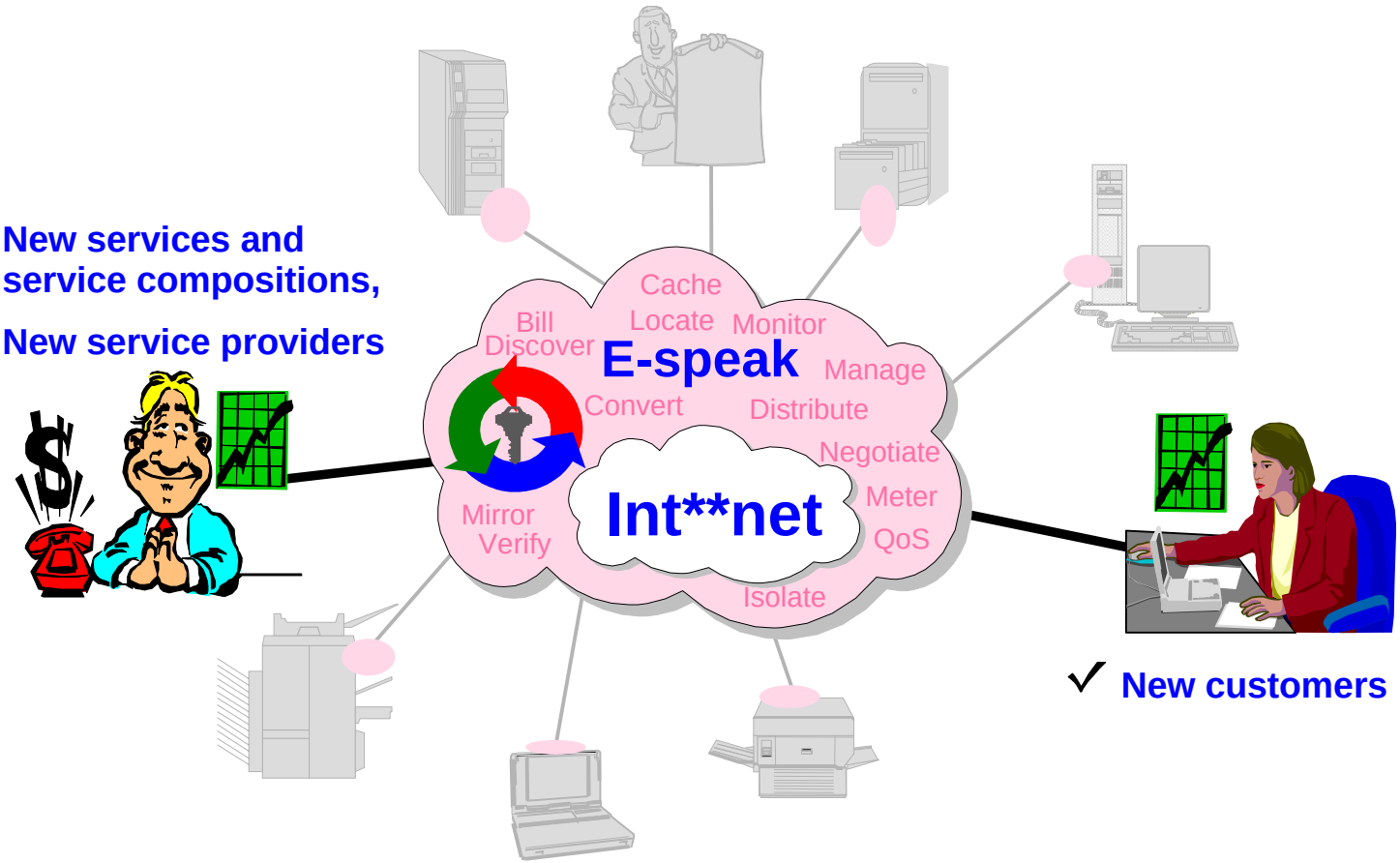
E-Speak Technology Stack



Why e-speak?

Open Services Marketplace

- ✓ New services and service compositions,
- ✓ New service providers

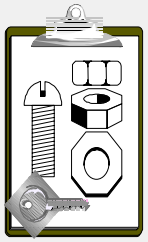


- ✓ New customers

Services Framework

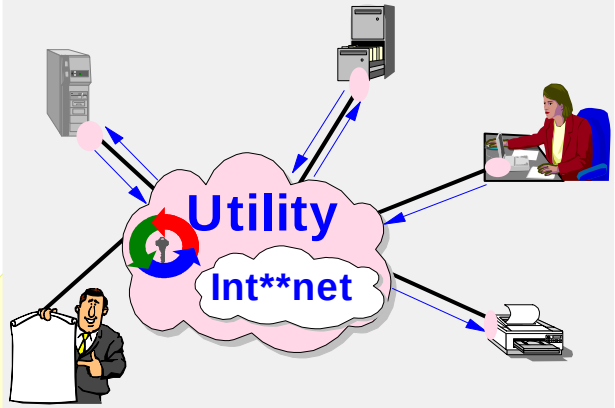
Reducing the barrier to new, competitive services

Service Specification

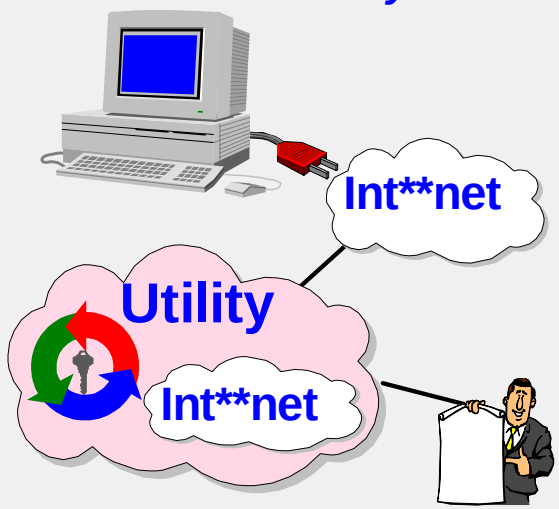


Workflow
- Requirements
- Access control
- (security, billing, ...)

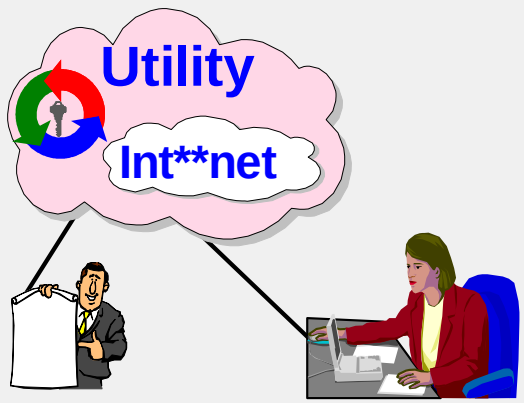
Service Execution



Service Discovery



Service Access



Internet Challenges

- Today's e-business web sites are proprietary, massive and costly to develop.
- Companies are forced to build out their entire offerings from the ground up.
- Even though they are connected to the Net, getting e-businesses and e-commerce sites to talk to one another in a meaningful way is difficult, special-case work.

The volume of business is limited by the bandwidth of eyeballs.

The Big Shifts Coming

- Ubiquitous e-services
- Modular building blocks
- Easy access from a wide array of devices and platforms:
  - Info appliances
  - PCs
  - Servers
  - Supercomputers
- E-services talk to each other in order to:
  - advertise capabilities
  - discover and ally with services offering new capabilities
  - negotiate to broker, bill, manage and monitor each other
- E-services interact with each other in a way that ensures security

E-speak Origins

- 1982 - Joel Birnbaum, Information Utility
- 1985 - Bill Rozas, I just want to be me
- 1989 - Alan Karp, Global Computer
- 1990 - Rajiv Gupta, Use obsolete machines
- 1994 - Arindam Banerji, Extensible OSes
- 1996 - Rajiv Gupta, World of services

Systems Evolution

Monolithic, proprietary systems

Open systems

2-tier client-server systems

Open data (Web)

3-tier, 4-tier, … systems
Proprietary, one-of services (Amazon.com, Expedia, eBay, …)

Open services (E-speak)

Dynamic n-tier systems
Brokered service composition (active personalization)

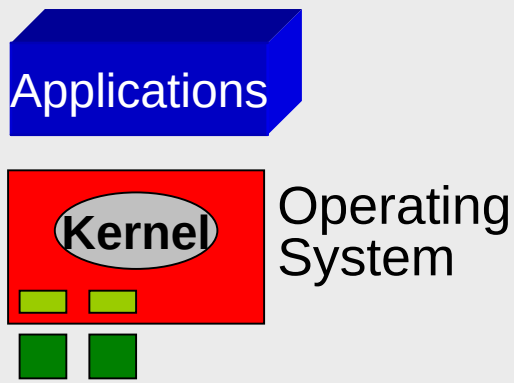
Assumptions and Implications

- Large number of machines
  - No centralized anything, forget consistency
- Dynamic
  - Deal with failures, new services
- Heterogeneous
  - Different hardware, OS, capability
- Hostile environment
  - Security is critical
- Different fiefdoms
  - Never look inside another machine

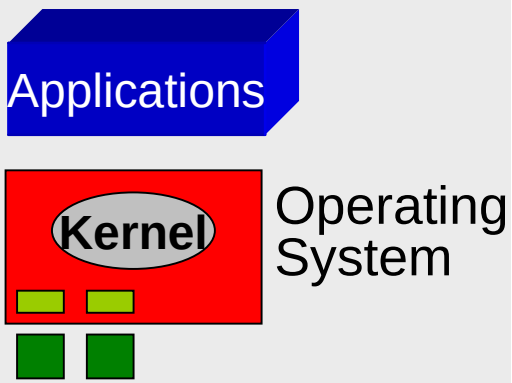
Architectural Principles

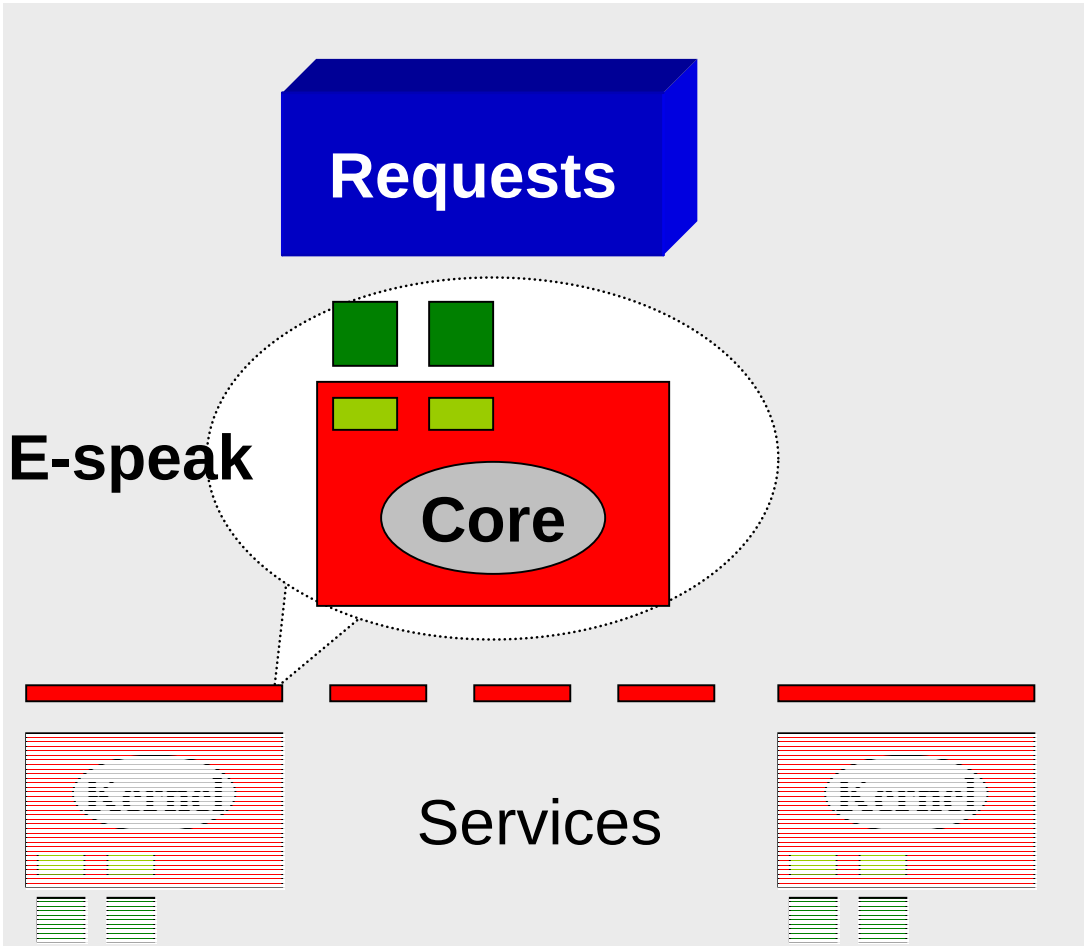
- Design for seamless, flexible, dynamic evolution
  - Current and future
  - Scalable, manageable, secure, extensible
- Simple and elegant abstractions and mechanisms
  - No "special-case" mechanisms, homogeneity requirements
  - Uniform abstractions for services and resources
  - Resource access virtualization and mediation
- Invent only where necessary
  - Leverage and complement industry standards

E-speak in Perspective

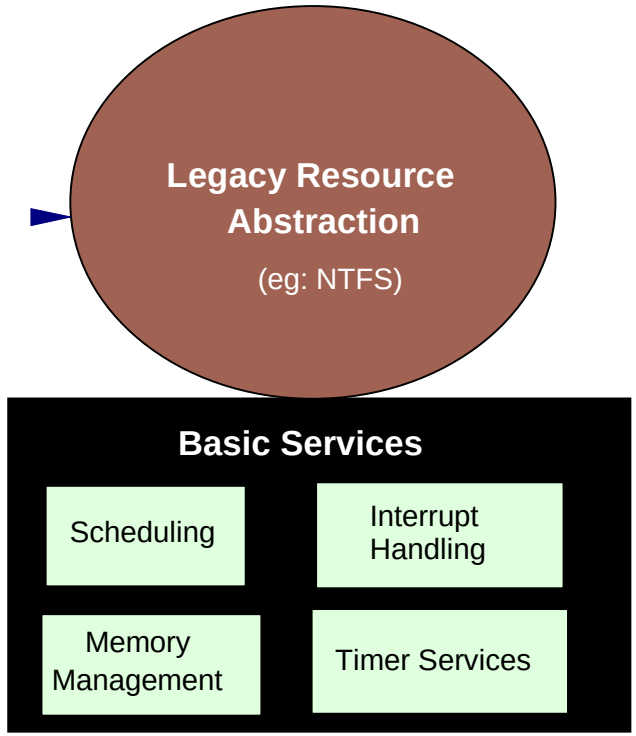


E-speak in Perspective



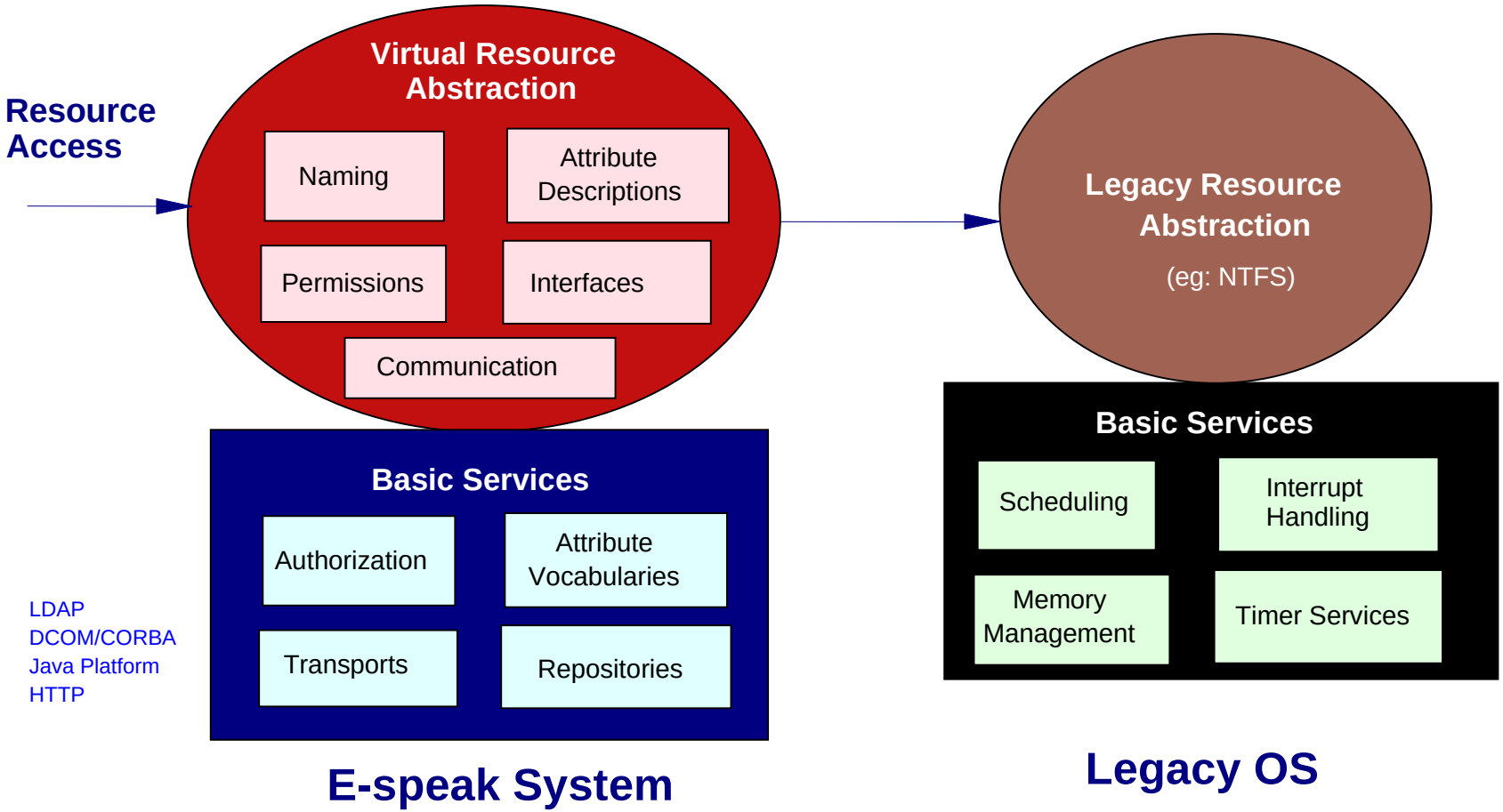


Client Utility Resource Model

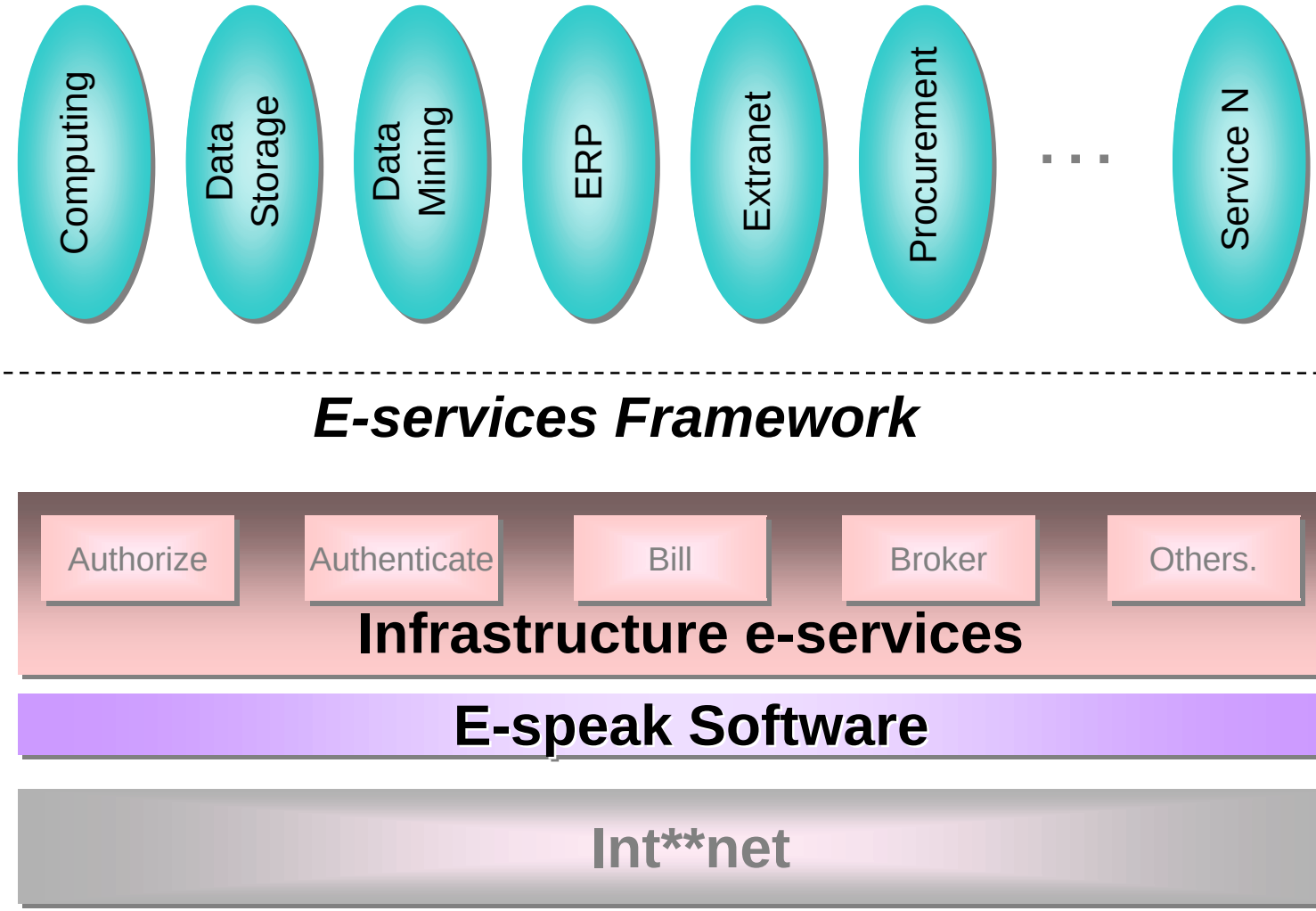


Legacy OS

E-speak Resource Model



E-speak Environment



Technology Innovations

- System Architecture
- Naming Model
- Security Model
- Meta-data Model
- Event Model

Key Abstractions

- Everything is a resource
- Naming
  - Only way to reference a resource
  - All names are private
- Security
  - Separate control of names and access rights
- Description
  - Customizable vocabularies
- Management
  - Every access mediated

Technology Innovations

System Architecture

- **Mediated access to services**
- **Uniform resource model**
- **Manipulation of resource representations, not resource specifics**

Creates Open Services Model for the Internet

- **Anything can be created as a new service using same model**
- **Heterogeneous management tools, security policies can be applied without compromising simplicity**
- **New service types and semantics can be dynamically introduced**
- **Services can be seamlessly interposed and distributed even across firewalls**
- **Provide functionality to enable commerce in services, e.g., monitoring, auditing, billing**

System Structure

- Federation of Logical Machines
- Logical Machine
  - Active entity - Core
  - Passive component - Repository
- Mailbox metaphor for requests to Core

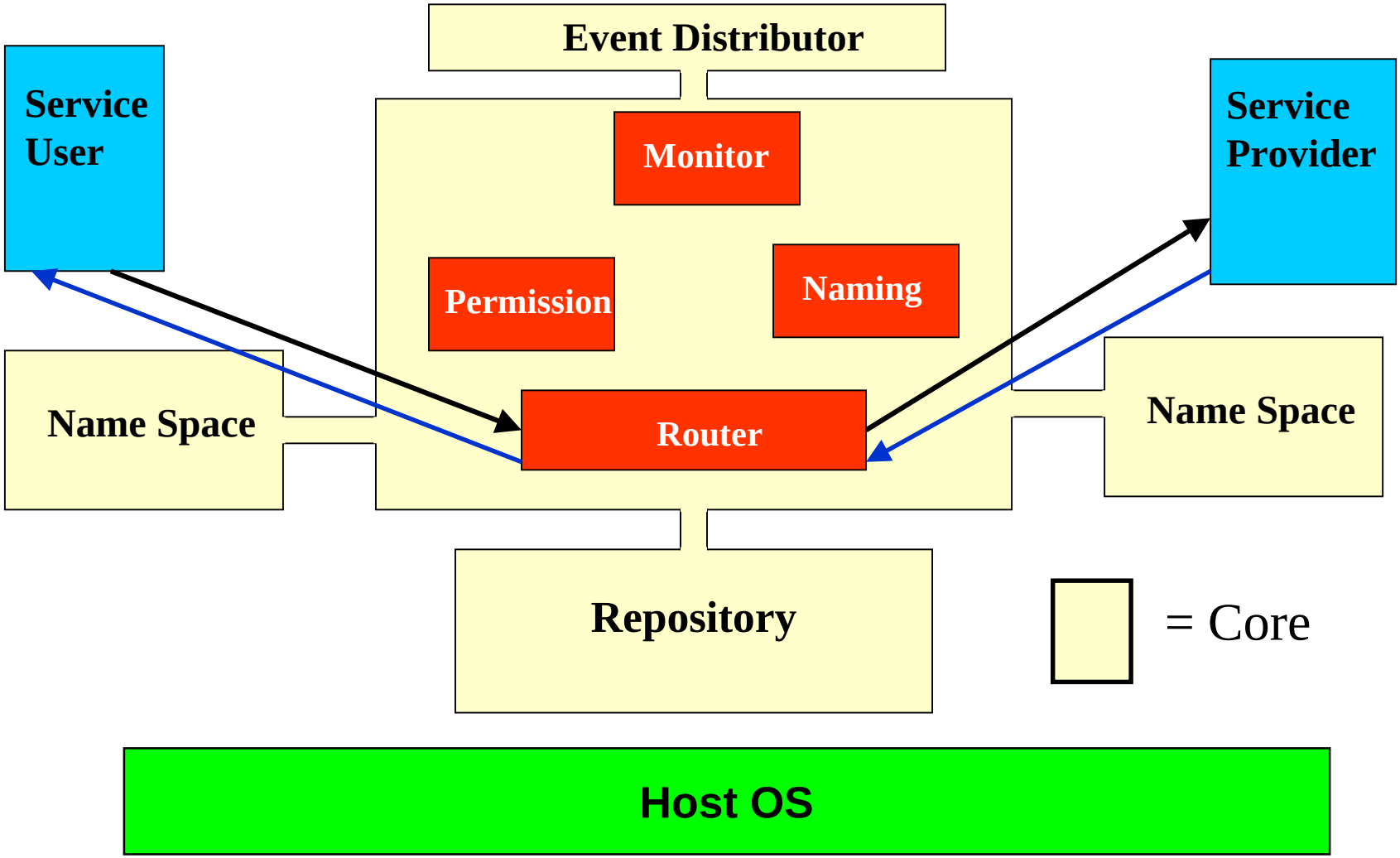
Fundamentals

- Every resource’s metadata registered with Core
- Tasks access resources by name
- Core associates name with resource metadata

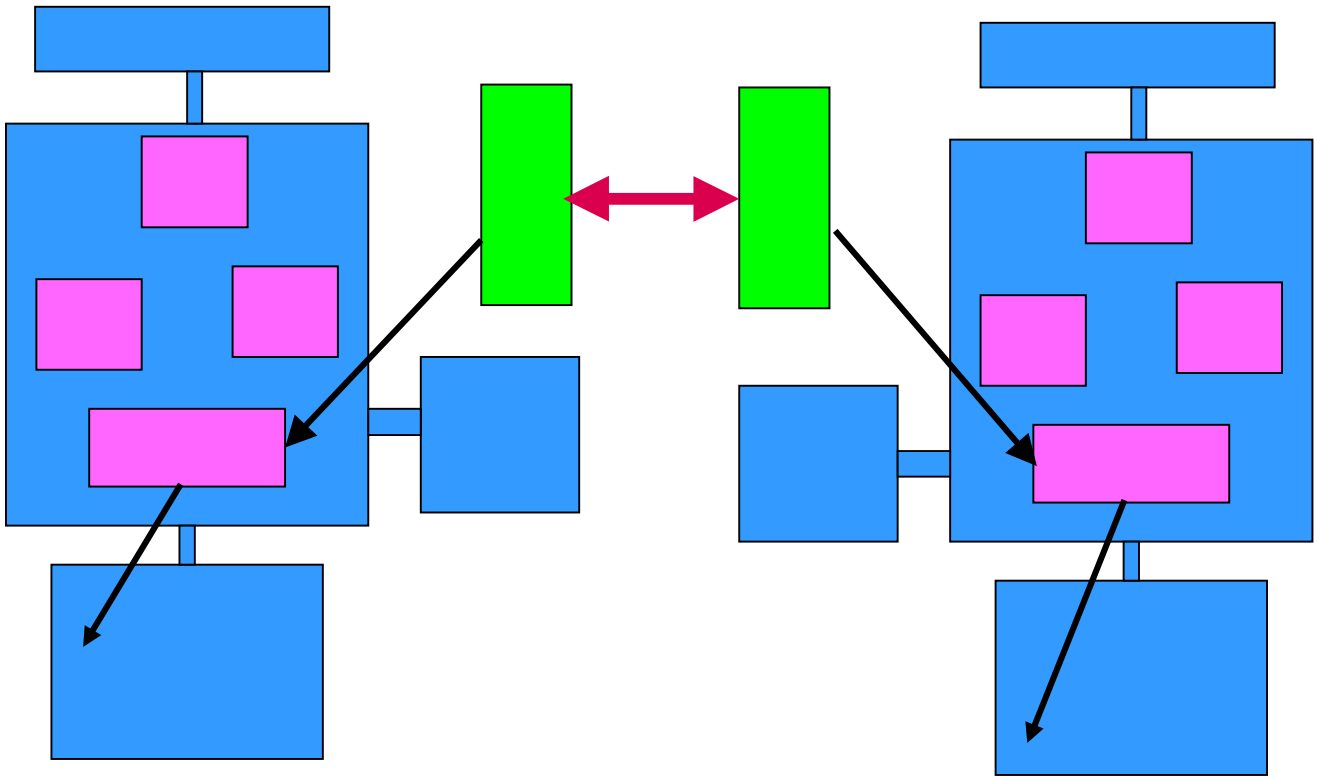
Use Model

- Each task has an outbox connected to the Core
  - Outgoing message has envelope and payload
- Each task has zero or more inboxes
  - Incoming message has envelope and payload
- Core-related data in envelope
- Application data in payload

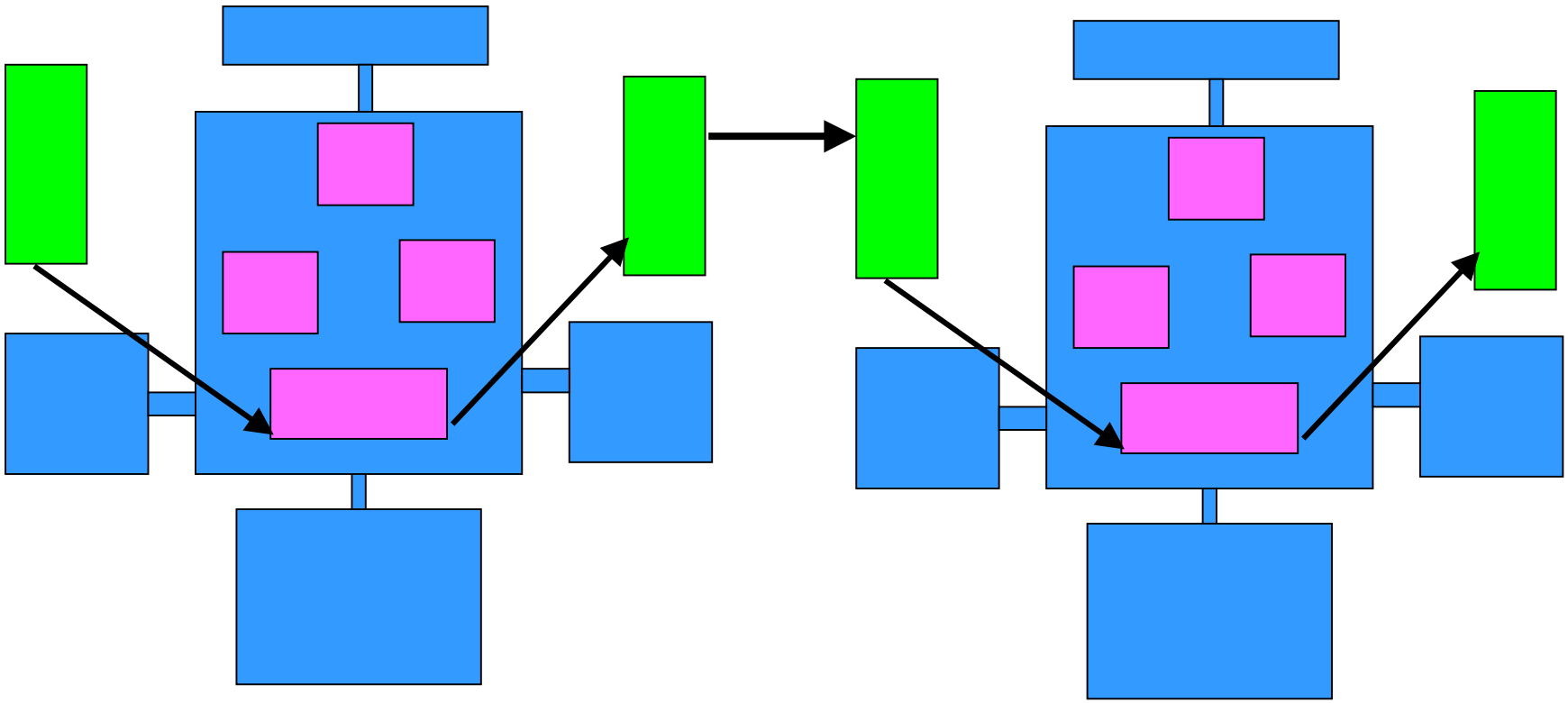
Single Machine View



Connecting Two Machines



Using a Remote Resource

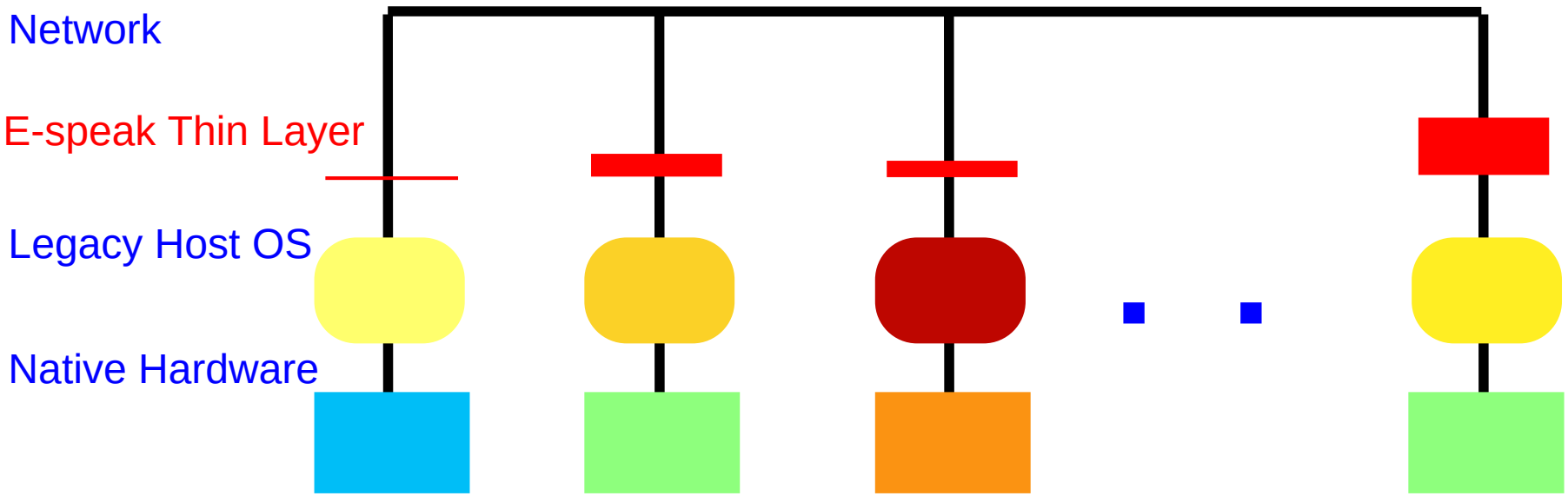


Distribution Model

- **Seamless Distribution**
  - **Uniformity in service interactions**
  - **Support for both remote evaluation and remote operation**
    - **Dynamic loading is subsumed**
  - **Proxies mimic resource handlers**
  - **ESIP-ABI *defines* the inter-machine architecture**

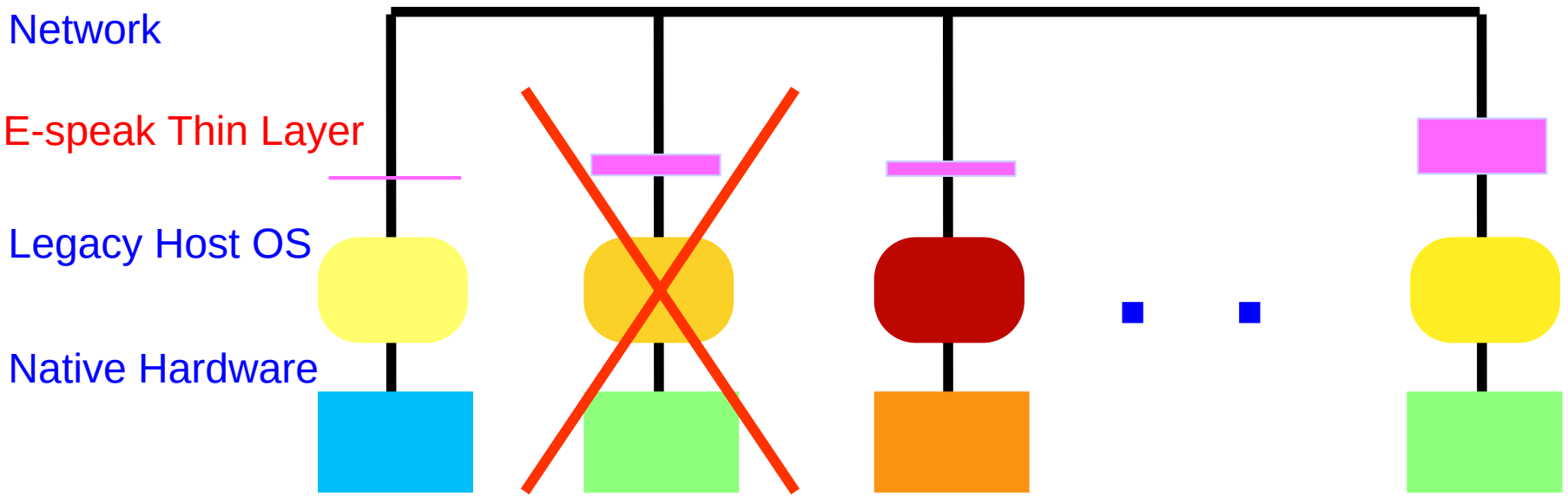
Architecture Overview

Dynamic Federation Model



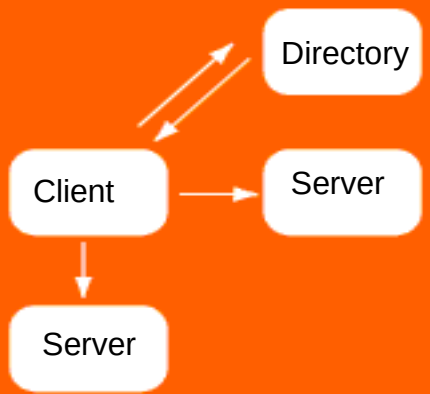
System Overview

Dynamic Federation Model



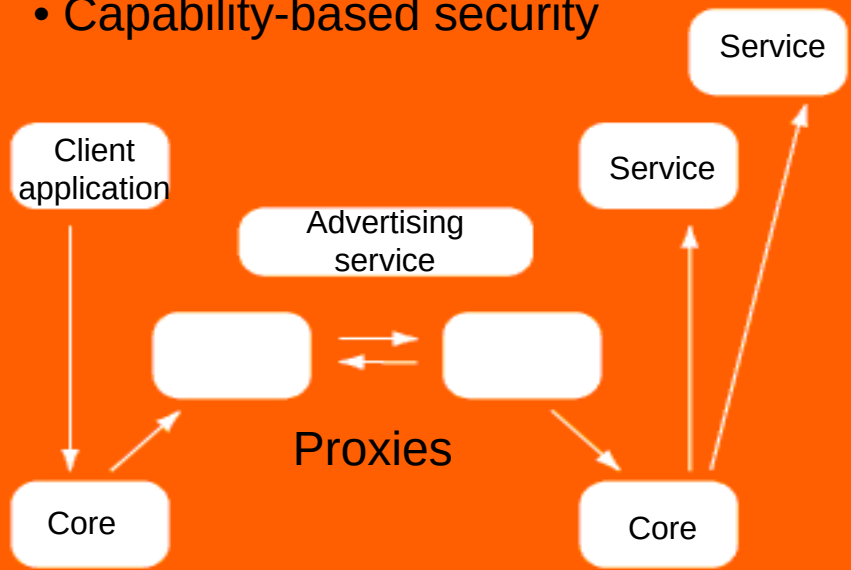
E-speak vs.Traditional Middleware

- Directory lookup
- Direct access to service
- No mediation
- No virtualization



Traditional Middleware

- Virtual service intermediation
- Uniform services model
- Attribute-based lookup
- Individual name spaces
- Capability-based security



e-speak

Technology Innovations

Naming Model

- Local, context-sensitive naming
- Name associations allow resource lookup algorithms
- Partial associations provide hooks for external decision services

Client-Service provider bond scalably, can be reasoned about at run-time

- **Services/agents do not require pre-negotiation**
- **Transformers can be seamlessly interposed**
- **Enables per-client, per-role, context-sensitive customization**
- **Enables hot-plug replacement, moving, mirroring of resources**

Flexible Name Bindings

- Bind a name to
  - A resource
  - A set of resources
  - A look-up request
  - All of the above
- Name is bound to an algorithm for finding resource
- Can pass bindings between tasks

Technology Innovations

Security Model

- Access rights, capabilities are resources
- Separation of name visibility from access rights
- Remote access based on trust established between machines

Fine-grained, dynamic protected access to services

- **Mapping between different security, authentication infrastructures**
- **Simple, selective yet dynamic delegation of privileges with revocation**

Access Control

- Name
  - Client can only reference a resource by name
  - Name is local to client with mapping in name space
- Rights
  - Client presents keys that open locks
  - Core delivers unlocked permissions
- Right to use name
  - Keys are resources referenced by name
  - Keys express name visibility rules

Technology Innovations

Meta-data Model

- **Vocabularies are resources**
- **Translations are integrated into design**

Flexible, scalable services discovery and location

- **Translation between XML-LDAP schema becomes a secure service**
- **Searches and service locations can be optimized through add-on services, e.g. find HP printer using Lexmark printer MIB grammar without requiring homogenization**
- **Advertising services can be used to scalably find remote resources**

Lookup Usage

- Comparison shopping

  find (“ServiceType == ‘AirLine’”,
  “Path == ‘LAX-SFO’ ”,
  “Cost < $90”);

- Locating services

  find (“ServiceName == ‘Citibank’,
  “Location == ‘Sunnyvale’ OR ‘Santa Clara’”);

- XML specification support

Advertising Service

- Lookup in local Repository results in name binding
- Look in advertising service if not found
  - Get back a machine to contact
  - Ask for resource once connection established
- No permanent connection needed
  - Advertise in many places
  - Lookup in many advertising services
- Used to form communities

Technology Innovations

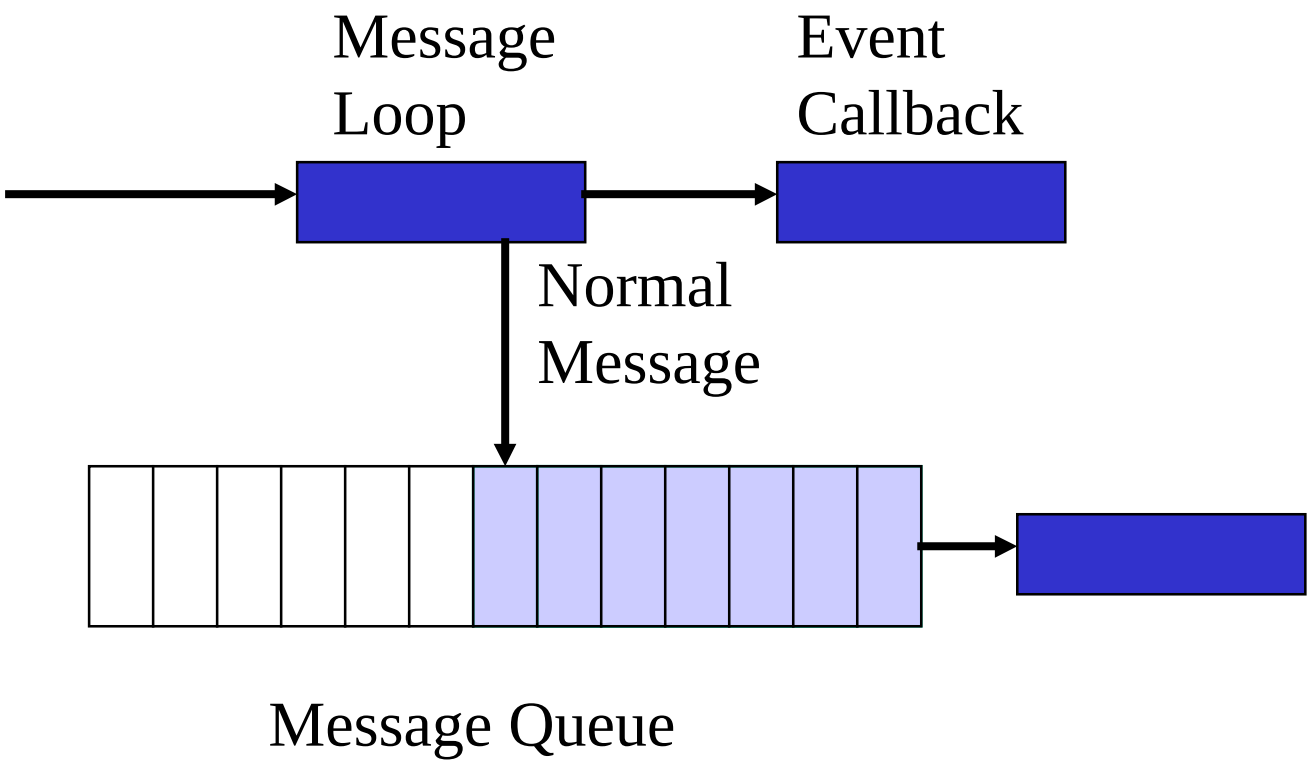
Events

- **Publish-Distribute-Subscribe**
- **Filters on subscription and publication**
- **Control of events with e-speak permissions**

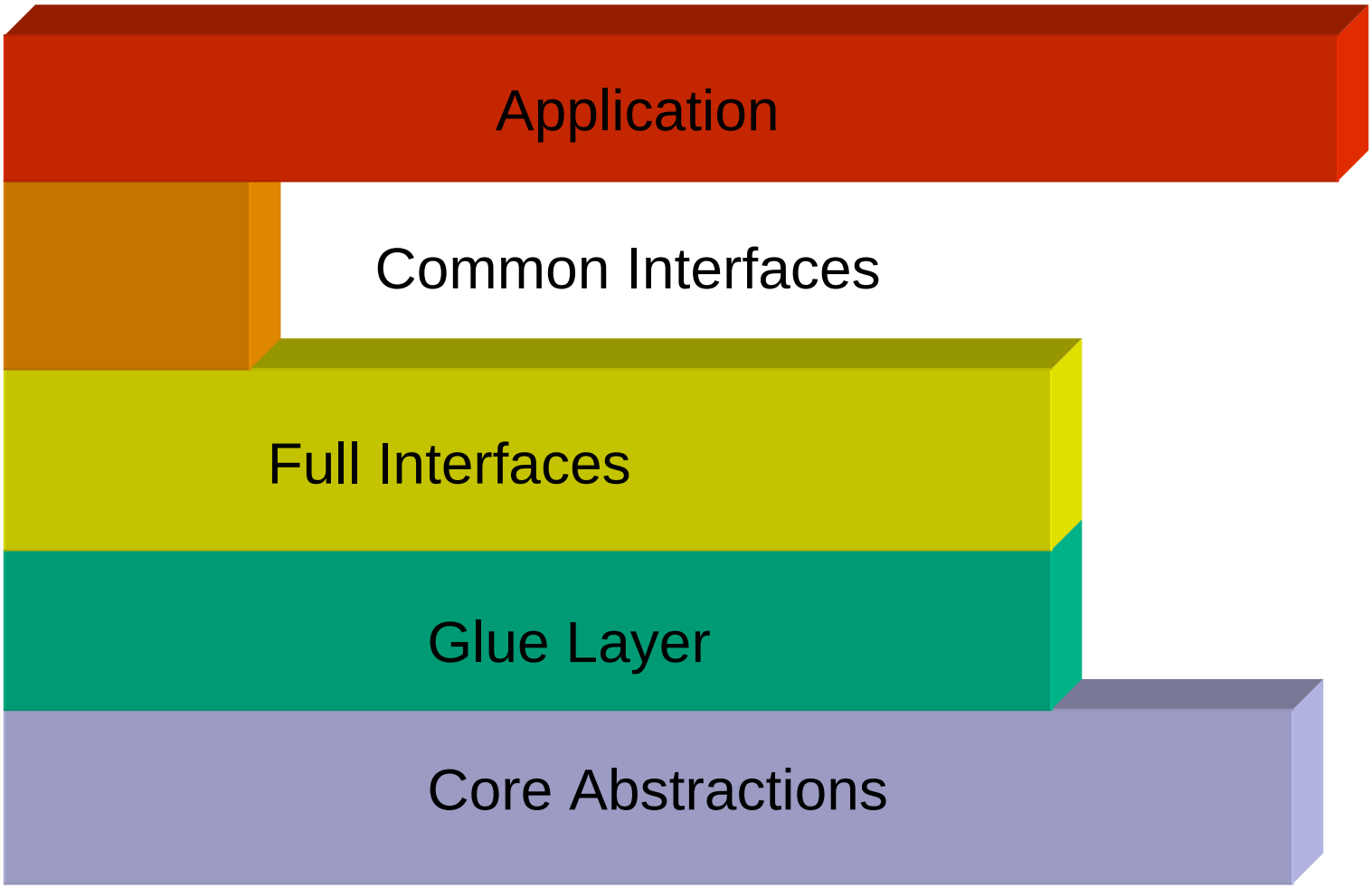
Flexible, controllable event infrastructure

- **Used to build management infrastructure**
- **Appropriate for data consistency**
- **Discoverable events**
- **Unified model for management and application events**
- **Event state and filters based on Vocabularies**

Event Handling



Programming Model



Programming Model

Three visible abstractions

- **Service - invoked by client**
- **Contract - defines interface**
- **Vocabulary - describes service for discovery**

Network Object Model

- **Stubs provided by IDL**
- **Download Java stubs**
- **Reference stub by name**

Direct messaging and Document Exchange also supported

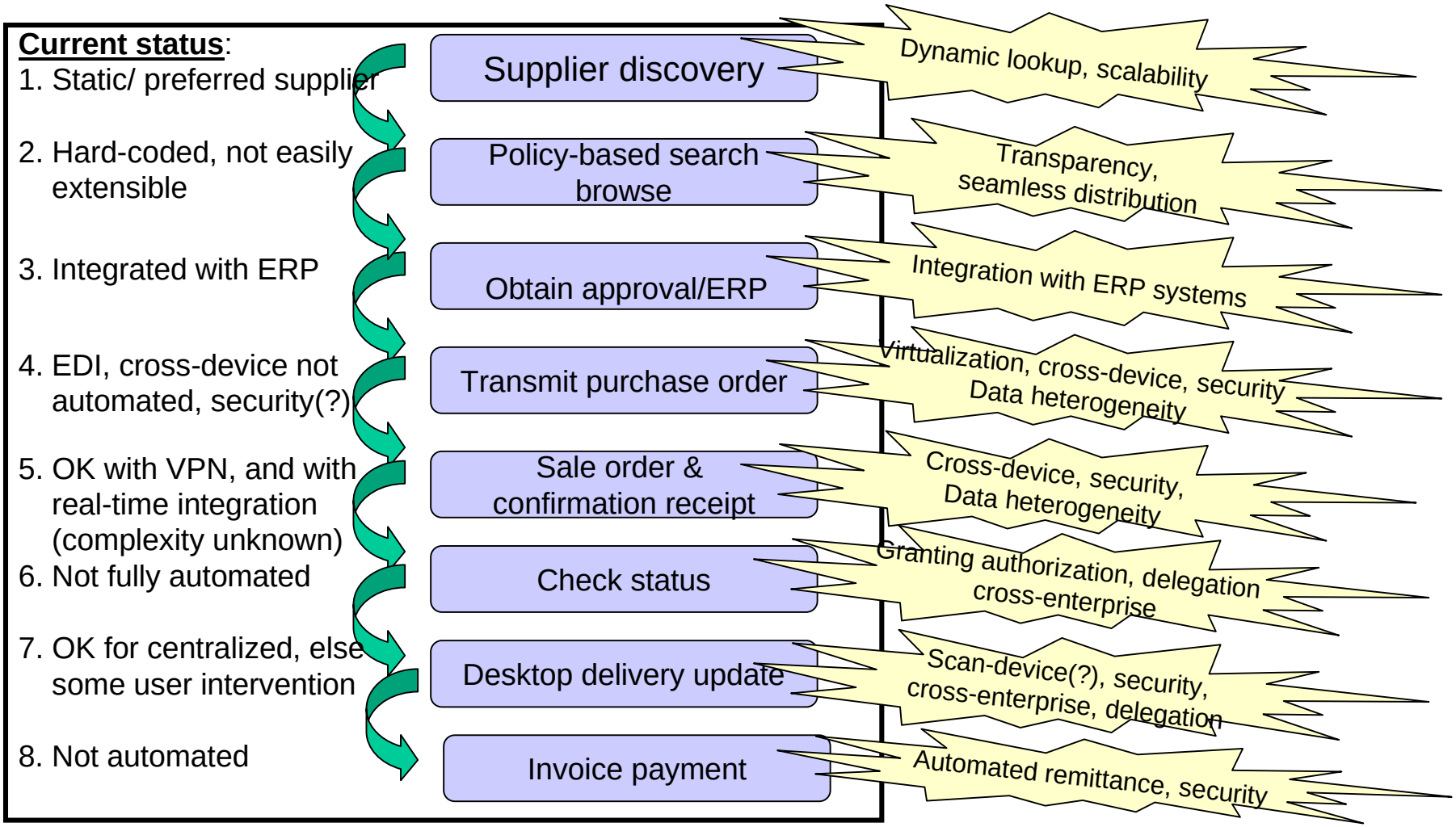
Using a Service

```
main ( )
{
  ESConnection c = new ESConnection(argv[0]);

  String intf = “echo.echoService”;
  ESServiceFinder sf = new ESServiceFinder(c,intf);
  EchoServiceIntf echoSvc = (EchoServiceIntf)
        sf.find(“Name=‘echoServer’”);

  String echo = echoSvc.echo(“Hellllooooo”);

  System.out.println(echo);
}
```

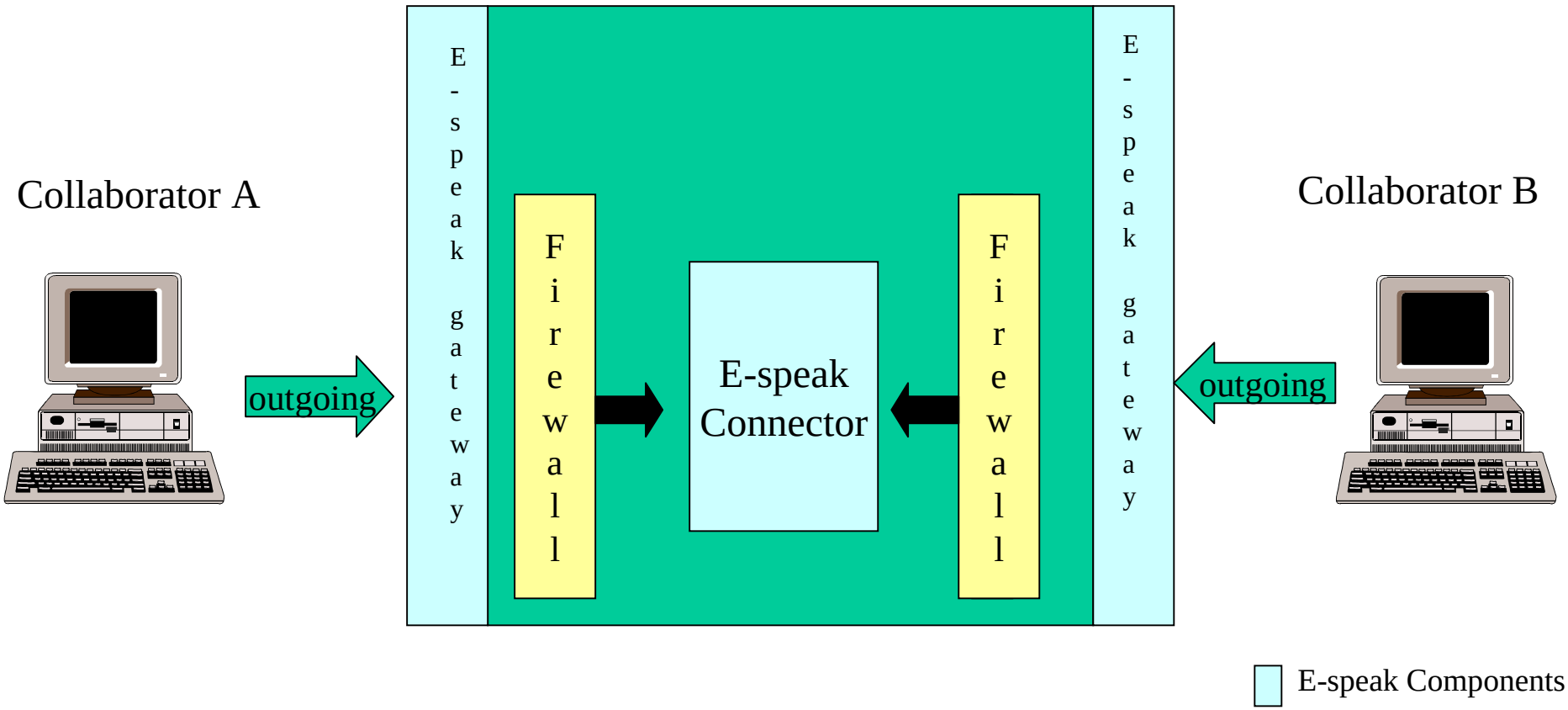
Create a Service

```
main ()
{
  ESConnection c = new ESConnection(argv[0]);

  ServiceComponent sc = new ServiceComponent(c,“echoServer”);
  sc.setImplementation(new EchoServiceImpl("EchoServer"));
  sc.registerService();

  sc.start ();
}

public interface EchoServiceIntf                    // E-speak IDL
{  public String echo(String in); }

Class EchoServiceImpl implements EchoServiceIntf
{  String echo(String in) { return in; } }
```

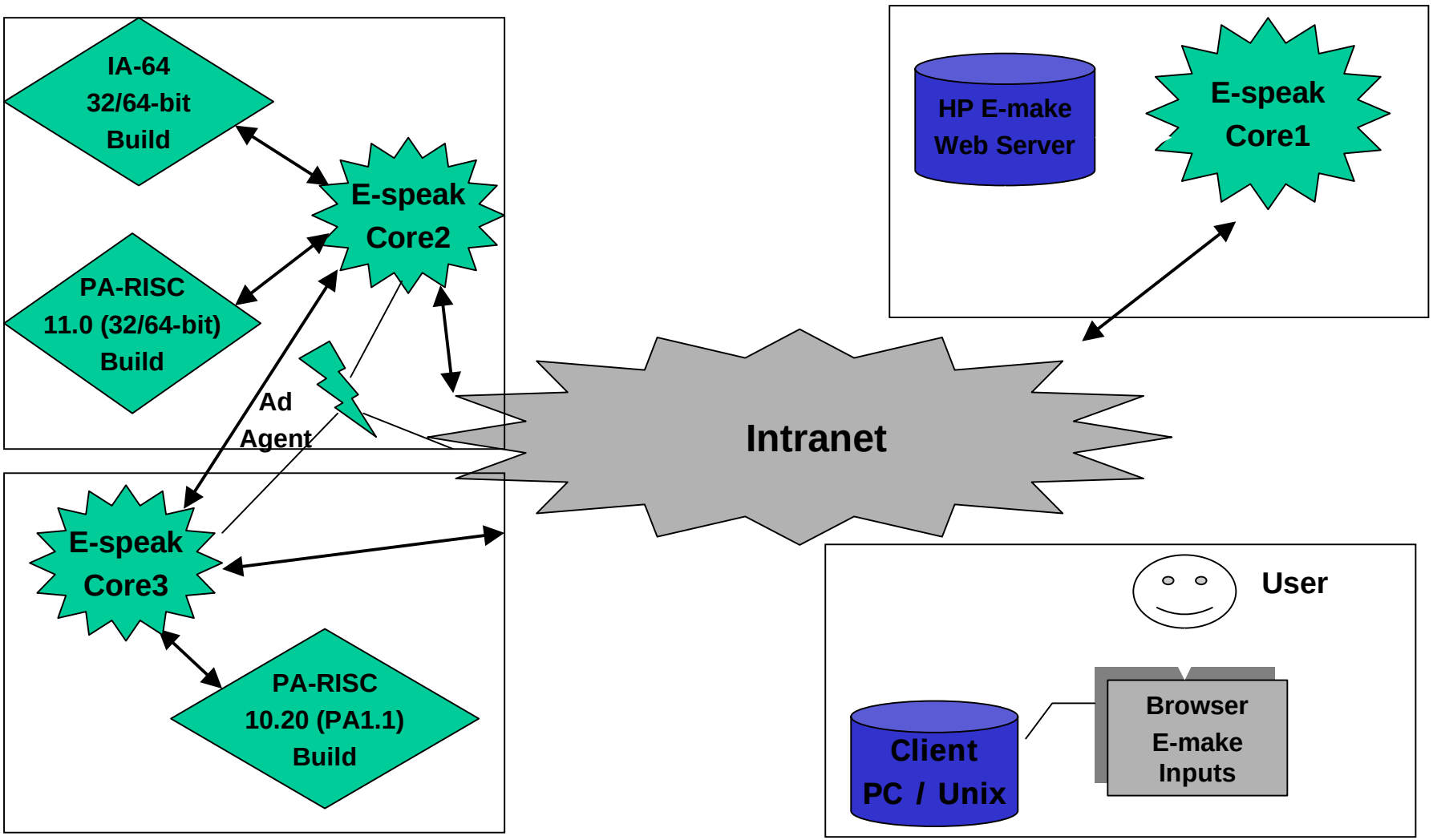
Business to Business Procurement/ SCM



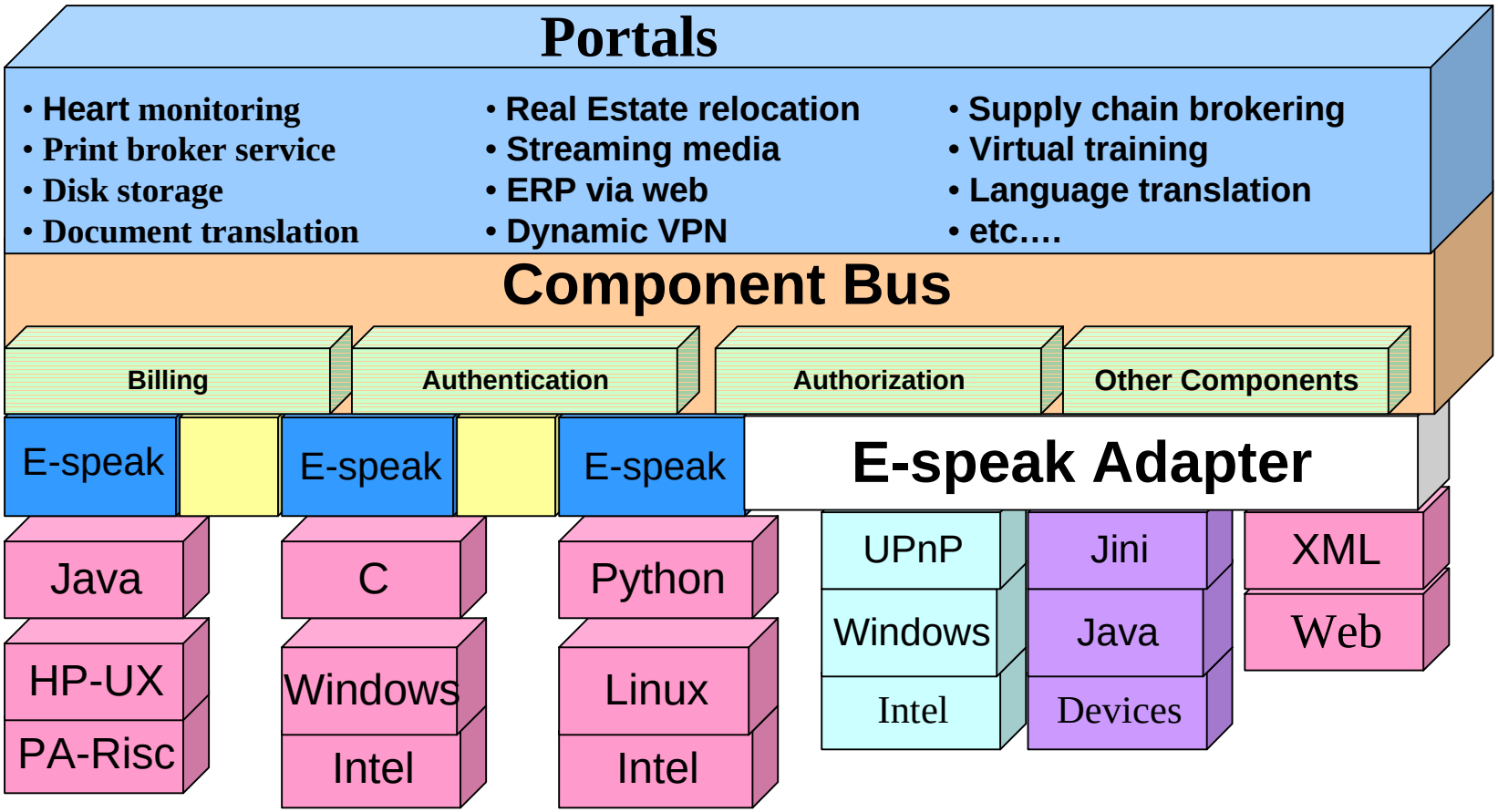
E-speak BPC Framework



Build Service



Truly Universal Environment



E-speak's Benefits

- Simplifies composition of services
- Dynamic creation of new value chains
- Spontaneous, ad-hoc, secure interactions across firewalls (without pre-negotiated names and standards)
- Slower rate of obsolescence
- Choice of standards

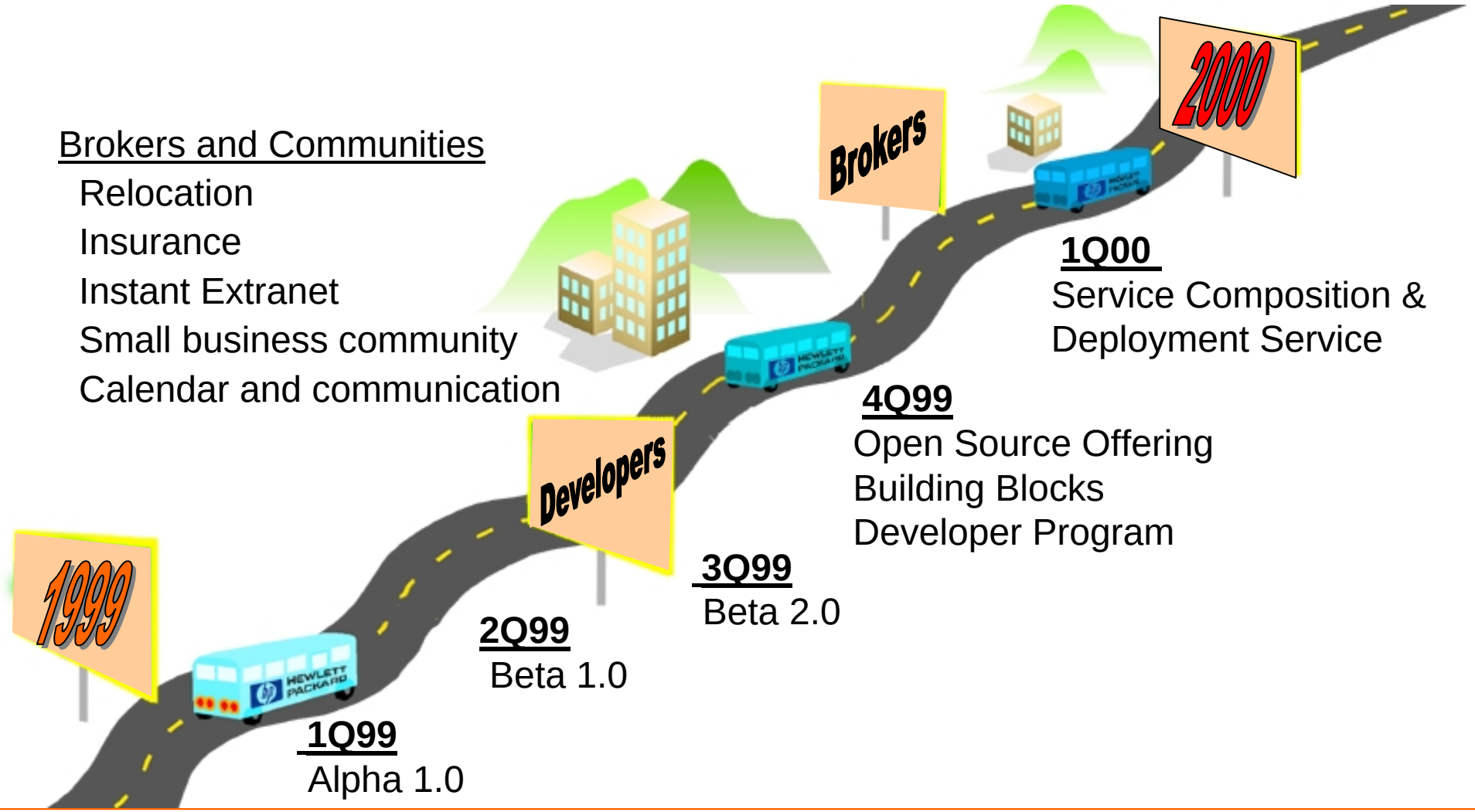
E-speak's Unique Features

- No global name space
- Novel capability model for access control
- Vocabularies as resources
- Separation of control data and resource semantics
- System structure
- Consistent service interface
- Dynamic service virtualization
- Brokering, delegation, revocation
- WAN-based fine grain security
- WAN-based scalability

Summary

- E-speak will do for services what the Web has done for data
- New business opportunities will be driven by the spontaneous composition of services
- E-speak will fuel the Internet's shift from the do-it-yourself model to the do-it-for-me model

E-speak Roll Out Plan



Brokers and Communities
- Relocation
- Insurance
- Instant Extranet
- Small business community
- Calendar and communication

1Q99
Alpha 1.0

2Q99
Beta 1.0

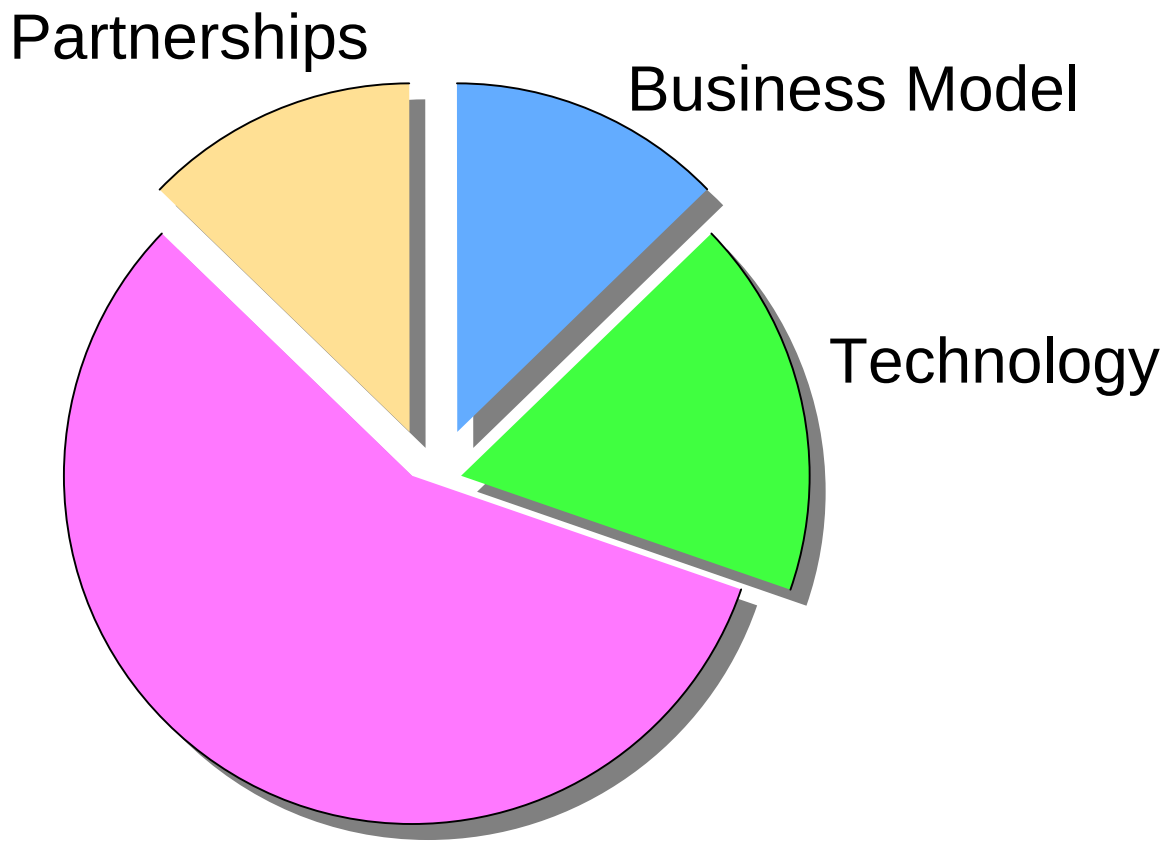
3Q99
Beta 2.0

4Q99
Open Source Offering
Building Blocks
Developer Program

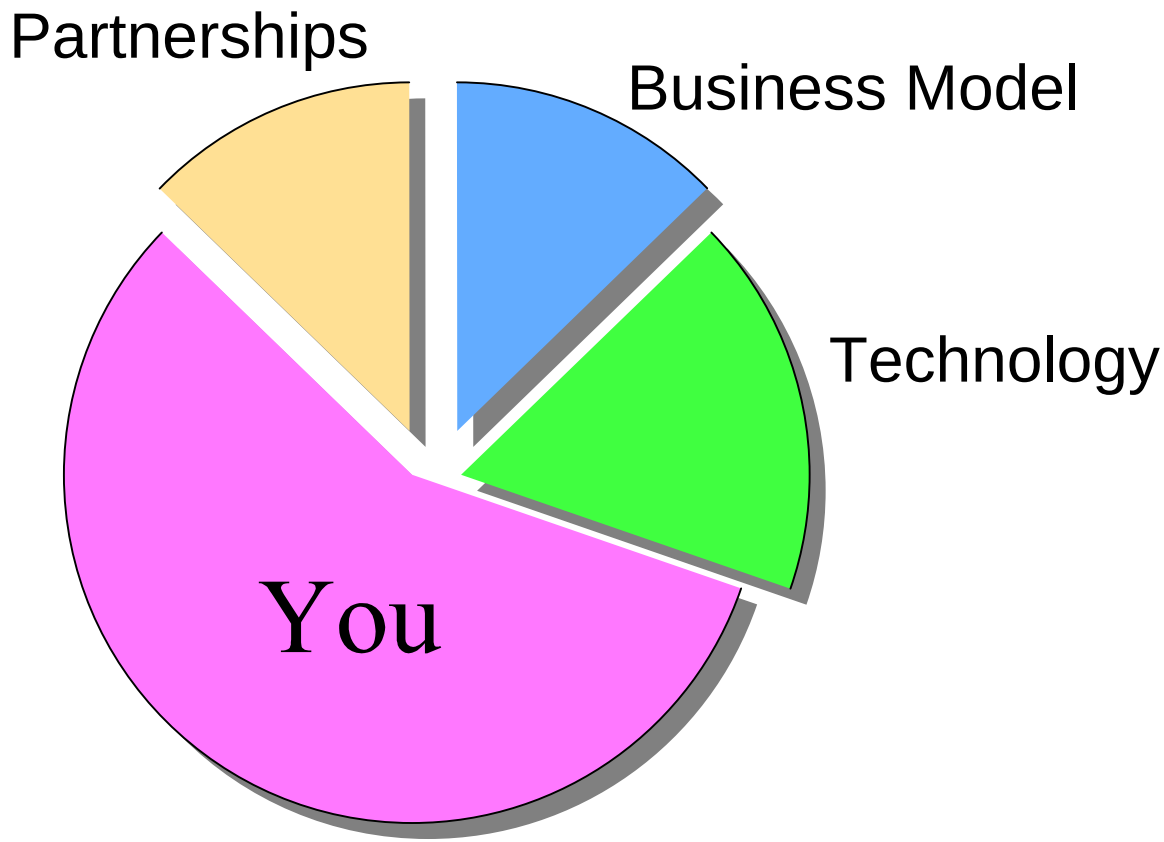
1Q00
Service Composition &
Deployment Service


Bell
NexGen SI
Integrating Technology Solutions for Business
GRIC
GRIC Communications, Inc.
frontier
GLOBALCENTER
USWEST
BT
ARIBA
NOKIA
CONNECTING PEOPLE
viaLink
Software.com
Oko Bank
HPY
Wherever you are.
Seagate
CompTel
ERNST & YOUNG
FROM THOUGHT TO FINISH.
LPG
LEXACOM
Systran
PEOPLE Soft
Yomi Media
BEA
US Internetworking
Sapient
Novell
Intellisys
NETWORK
STUDIO ARCHETYPE
Qwest
ORACLE
Covation
CAP GEMINI
Ideas People Technology
PSINet
SAP
INTERNET TRAVEL NETWORK
EDS
Andersen Consulting
sonera
Make things click.

Success Factors



Success Factors



What is E-speak?

E-speak is an operating environment for the Internet that reduces the barriers to creating e-services.

What You Can Do

- Try it out.
  - The price is right - free.
  - It’s easy to get started.
- Build e-services for new environments.
  - The whole world is not commercial services.
  - Find novel uses - device OS, educational software, etc.
- Build new programming models.
  - The whole world is not commercial services.
  - Define new abstractions for real time, scientific, etc.
- Work on the open source code.
  - Make your mark by fixing what we’ve done wrong.
- Join the open source board.
  - Seats are still available.

http://www.hp.com/go/espeak