KQML

A Language and Protocol for
Knowledge and Information Exchange
or
A Language for Knowbots

Tim Finin
University of Maryland
Baltimore County
Baltimore MD USA

Richard Fritzson
Don McKay
Robin McEntire
Unisys Government Systems
Paoli PA USA

Overview

- The need for Intelligent Information Agents *(Knowbots)*
- The KQML language and protocol as the language for such agents
- KQML implementations, software architectures and applications
- Conclusions and future work

Sponsors

- ARPA Knowledge Sharing Effort
  - Advanced Research Projects Agency (G. Wiederhold)
  - Air Force Office of Scientific Research
  - National Science Foundation
  - Corporation for National Research Initiatives
- ARPA/Rome Laboratory Planning Initiative
  - ARPA (Steve Cross)
  - Rome Laboratory
- Other Supported Projects
  - BRL

Contributors

Jim Davis

Tim Finin

Richard Fritzson

Mike Genesereth

Michael Huhns

Dan Kuokka

Don McKay

Bob Neches

Ramesh Patil

Stu Shapiro

Marty Tenenbaum

Jay Weber

Gio Wiederhold

James Mcguire

Charles Simmons

and many more ...

Information Access in the Future

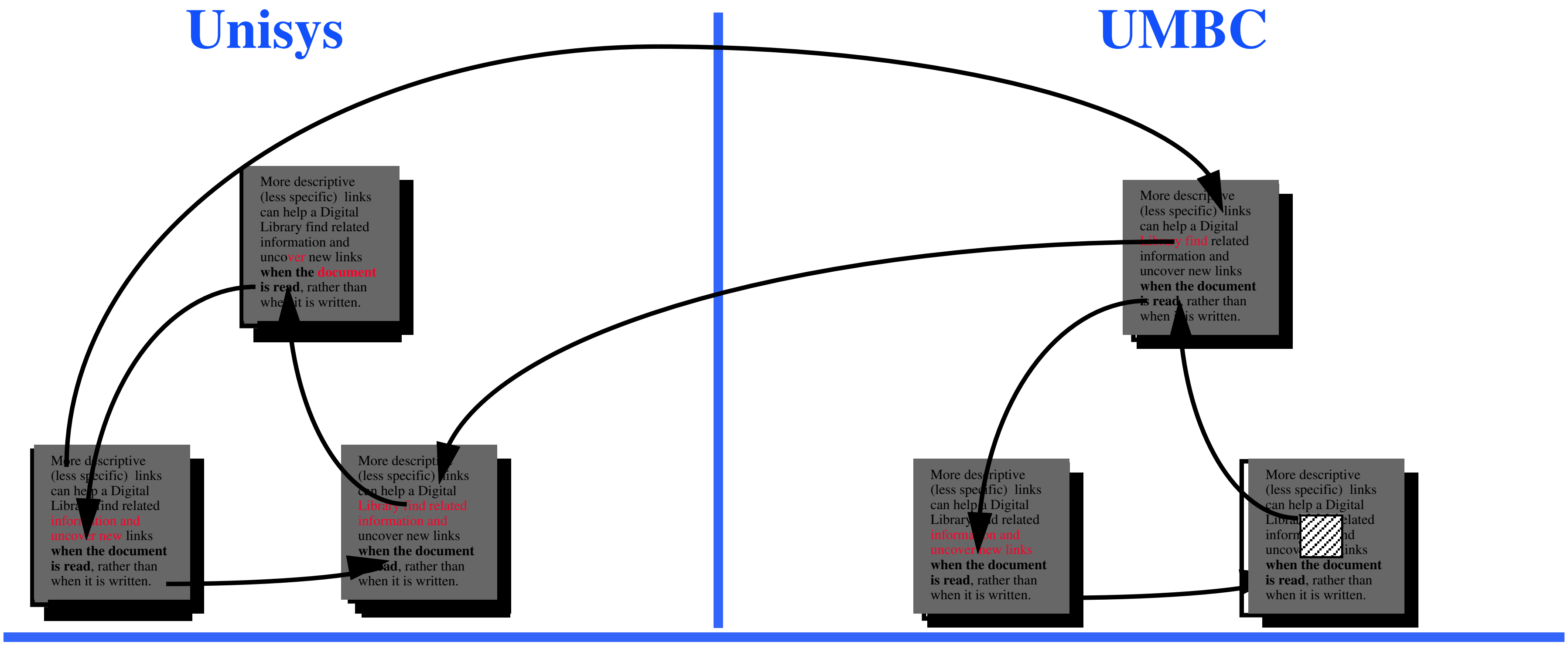
- Rapidly obtained information, useful to you, at the appropriate level of detail, from the most economical sources, anywhere in the world.
- Sophisticated interaction among many smart information agents.
- Digital libraries, mediated by digital librarians.
- An electronically levelled playing field.

⇨ Requires a standard language and protocol for information systems and agents.

Examples

- World Wide Web viewer, enhanced by a digital librarian.
- An Expert Report Generator
- Personalized and Interactive Newspaper and Television

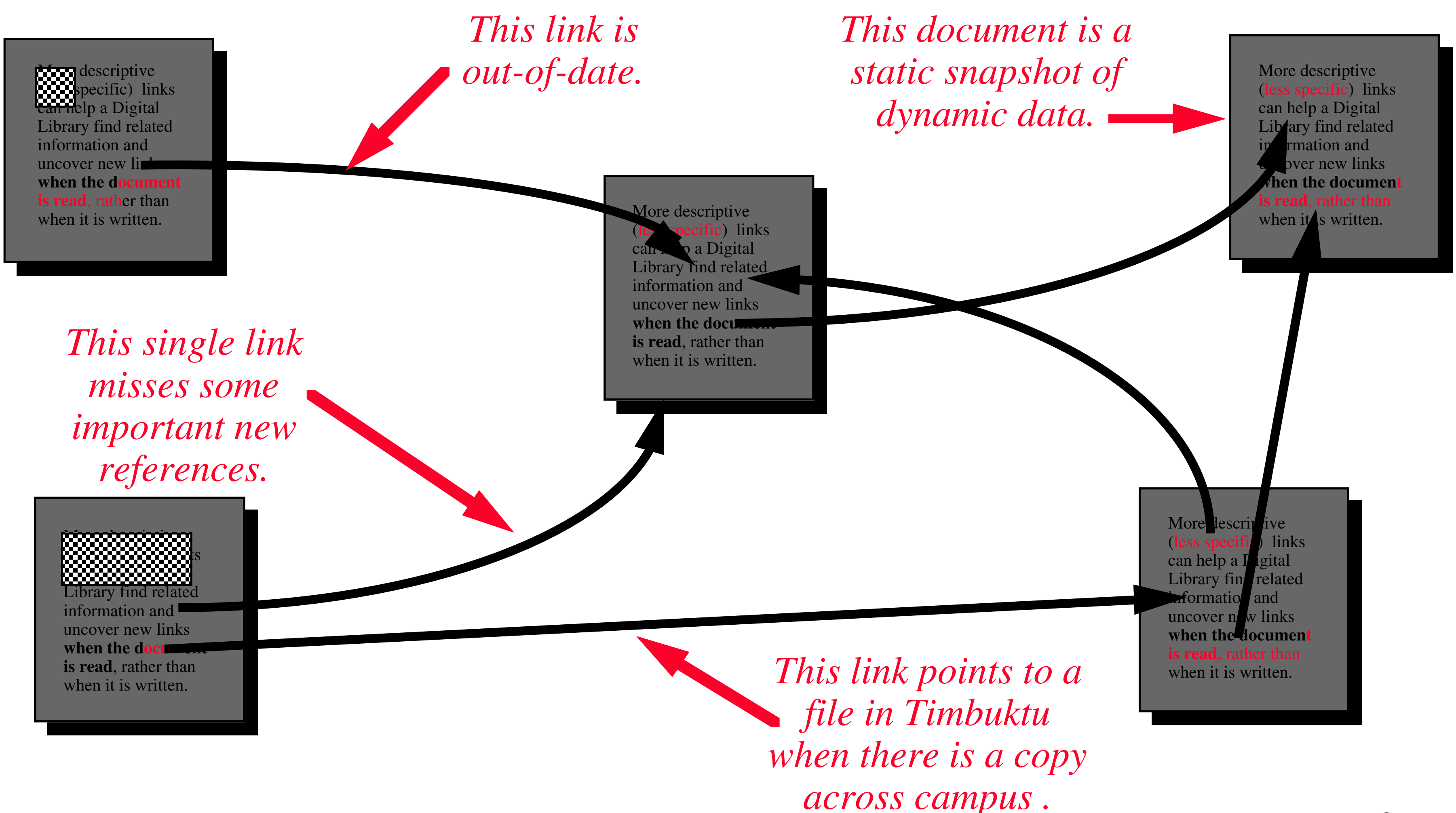
World Wide Web (WWW) as Distributed Hypertext



- Distributed Hypertext uses links which can point to text on another hypertext system.
- HTTP is the WWW protocol.

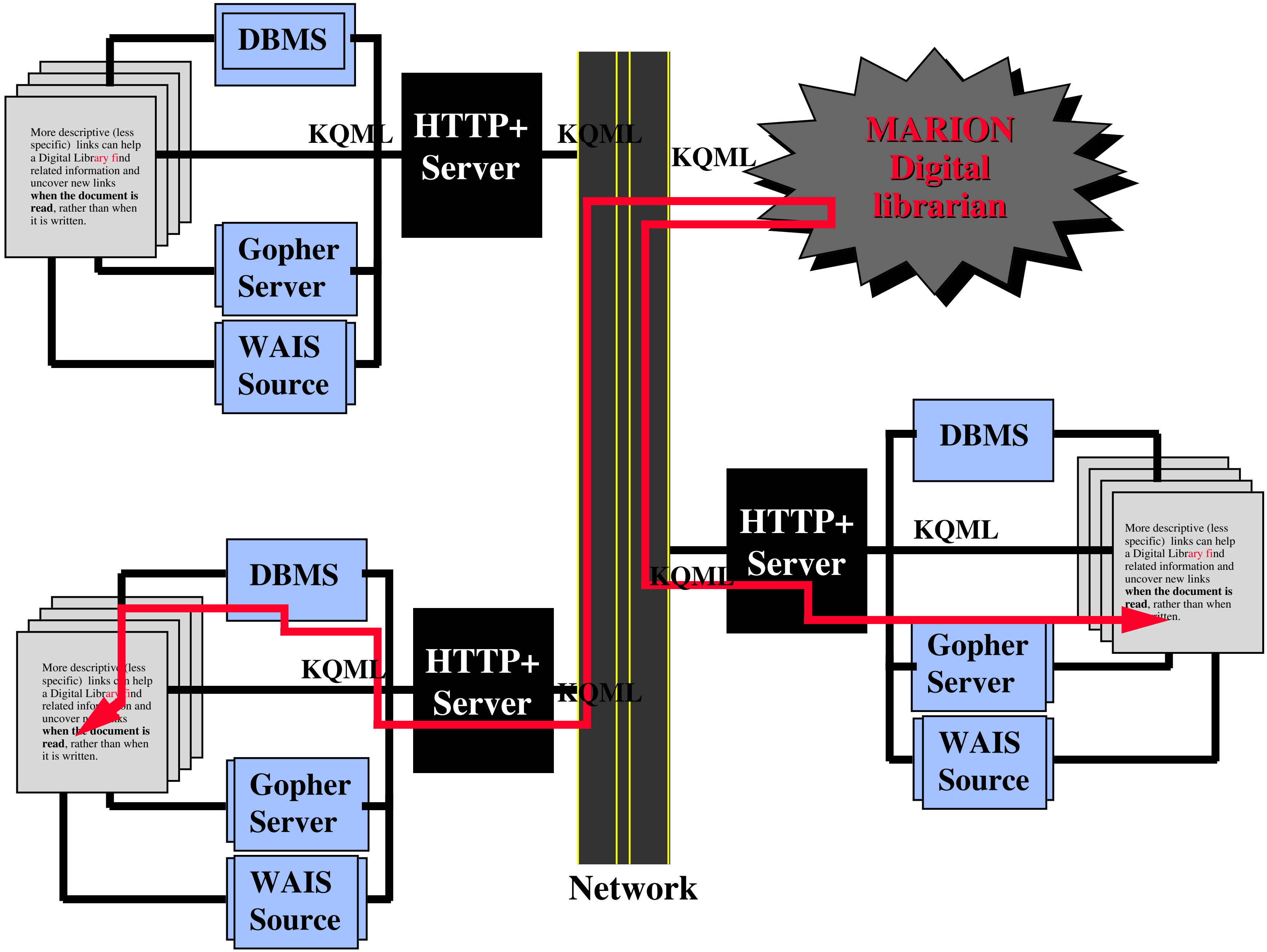
Improving the Web

Rigid, prewired links to static documents is not the right model



World Wide Web Enhancements

- New protocol: HTTP+KQML supports *links as queries*
- *Embedded queries* in the document placed by author or reader.
- Enhanced by the presence of a *digital librarian* who:
  - finds related information *when the document is read*, rather than when written.
  - can compose new, *custom documents* from databases and other agents.
  - can forward the query to other, *domain expert librarians*.

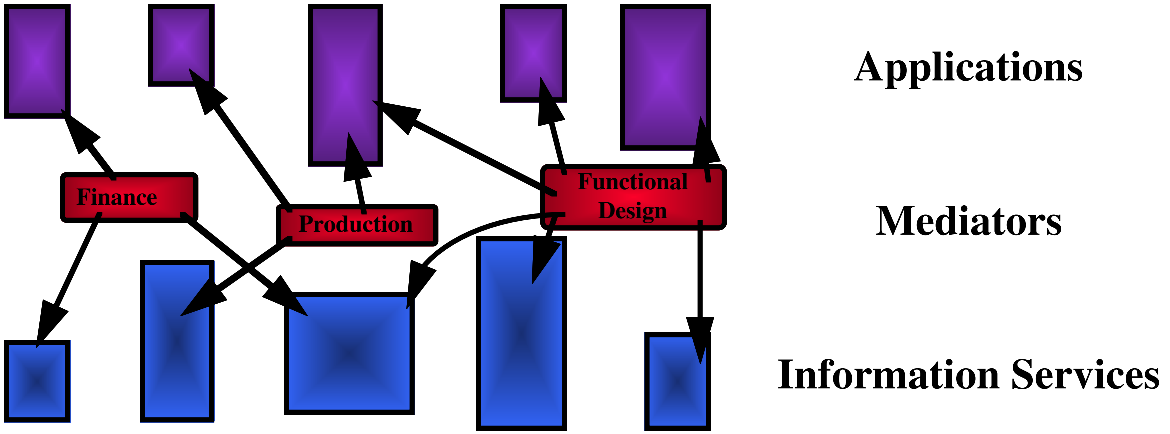

DBMS
More descriptive (less specific) links can help a Digital Library find related information and uncover new links **when the document is read**, rather than when it is written.
KQML
HTTP+ Server
KQML
KQML
MARION Digital librarian
Gopher Server
WAIS Source
DBMS
HTTP+ Server
KQML
KQML
More descriptive (less specific) links can help a Digital Library find related information and uncover new links **when the document is read**, rather than when it is written.
Gopher Server
WAIS Source
DBMS
More descriptive (less specific) links can help a Digital Library find related information and uncover new links **when the document is read**, rather than when it is written.
KQML
HTTP+ Server
KQML
Gopher Server
WAIS Source
Network

The Report Generator

- A general query is answered by a custom assembled report with
  - text extracted from other sources with summaries,
  - images,
  - reported statistics,
  - custom analyzed data
  - embedded live data, programs and agents
- How would this program find, recognize, and access the data?

Smart Information Agents

- Not just retrieval by users, but by intelligent information agents
  - Some directly representing users (e.g., librarians or knowbots)
  - Some serving other systems.
- Implemented using mediated architectures (a la Wiederhold)

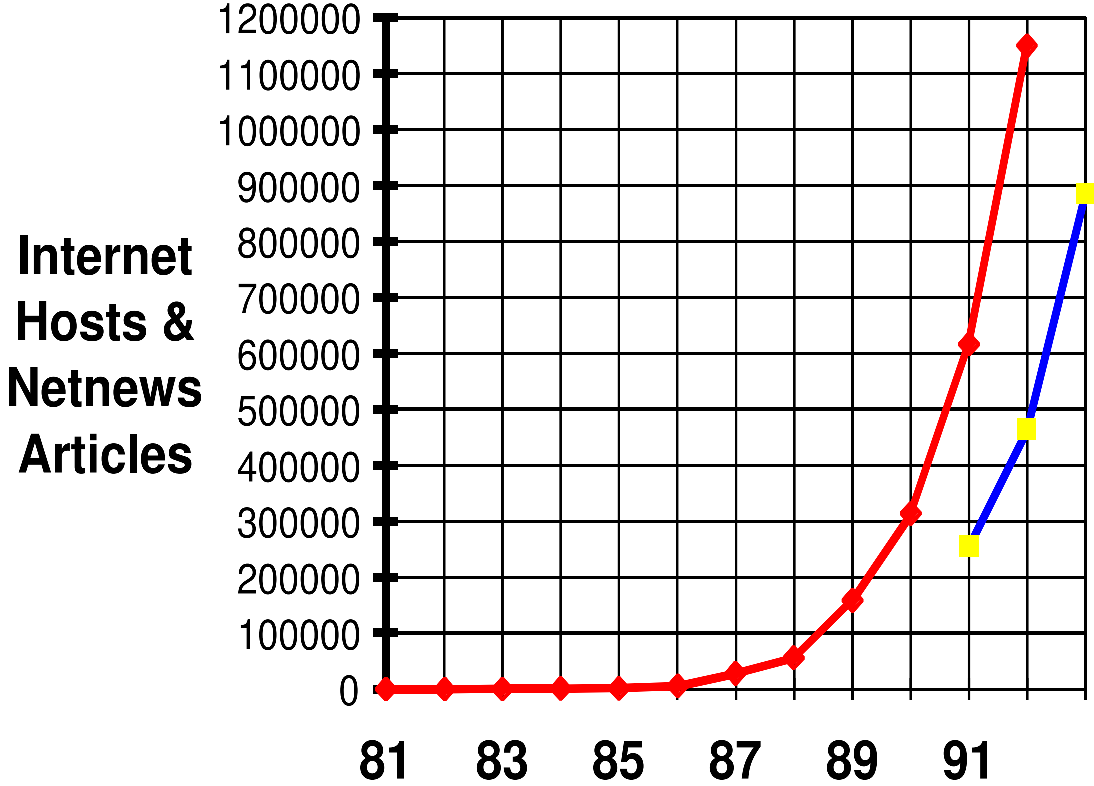


Where We Are Now ?

- **A difficult administration problem is getting worse quickly.**
- Internet Growth is explosive
- Data sources are hard to use
- Browsing and Searching tools have inconsistent interfaces and are meant for human users, not programs.

Where We Are Now ?
Internet Growth

Many new users and new information suppliers



Where We Are Now ?
Information Sources

- **Current Information Sources hard to use**
  - Unindexed text *(FTP, Gopher, HTTP)*
  - Menu driven sources *(e.g., Weather Underground)*
  - Library catalogs and searches *(many, varied interfaces)*
  - Commercial Databases *(many, varied interfaces)*
- **Current browsing and searching tools designed for people (*e.g.*,** *Gopher, Mosaic, Archie, Veronica, WAIS)*
- **and present subject indices to us as documents to be read rather than as databases to be used.**

⇨ **We need tools for programs, not just people.**

Where We Are Now ?
Administration is Impossible

- Simple links: can’t refer to “nearest” or “cheapest” source.
- Stale links: absolute pointers can’t be maintained.
- Machines are renamed, retired, moved.
- Local changes have *global* impact
- Current approaches *will not* scale up.

⇨ We need “intelligent” links.

Requirements for the Future

- Standards for smarter communication protocols
- Directory and information services which utilize and observe these standards
- Mediation services which rely on, use, and advertise in these directories.

⇨ KQML, or something like it, is a part of this picture

-- KQML --

Knowledge Query and Manipulation Language

and the

Knowledge Sharing Effort

ARPA Knowledge Sharing Effort
Working Groups

For agents to communicate, they need a *Common Language*

Common Syntax

- **Interlingua Working Group**
- **Common Knowledge Representation Systems Specification (KRSS) Working Group**

Common Semantics

- **Sharable and Reusable KBs (SRKB) Working Group**

Common Pragmatics

- **Interfaces and Architectures Working Group**

Common Syntax

- A shared language for representing information.
- E.g., KIF, KRSL, the LOOM printed representation, object SQL, ...
- A universally accepted language is not likely soon.

➪ Interlingua Working Group developing KIF (Knowledge Interchange Language) based on FOL with extensions.

Common Semantics

- Actually shared ontologies
- ***Ontology*** **:** A common vocabulary and agreed upon meanings to describe a subject domain.
- Relatively independent of syntax.
- Too difficult for just one. There will be many *domain specific ontologies* published.

⇨ SRKB working group developing tools and methodologies for building sharable ontologies.

Common Pragmatics

You must follow the same rules for discourse in order for others to find and speak with you.

- Who to talk to
- How to find them
- How to get their attention
- How to initiate and maintain a conversation
- How to deal with what they say
- How to respond appropriately

⇨ Interfaces and Architecture working group developing KQML.

KQML
Knowledge Query and Manipulation Language

- A high-level communication language and protocol for exchanging information independent of content syntax and ontology.
- A language in which to wrap information offering a uniform view of an *information agent as a knowledge base*.
- An extensible set of performatives expressing a *belief* or an *attitude* toward some information

⇨ The language of Knowbots

A KQML Message

- Represents a single *speech act* or *performative*

ask	reply	forward	...
tell	deny	subscribe	
achieve	monitor	evaluate	

- with an associated *semantics and protocol*
- with a list of *attribute/value pairs* , including

:content	:receiver	:aspect
:language	:reply-with	:in-reply-to
:topic	:label	...
:ontology	:from	
:sender	:to	

KQML Message Examples

```
(ask  :content (geoloc lax (?long ?lat))
      :language loom
      :ontology std_geo
      :from map_agent
      :to geo_server
      :label fm-query-10-23-92-14-09-07)
(tell :content "geoloc(lax, [42.45,83.21)"
     :language standard_prolog
     :ontology std_geo)
```

Performative

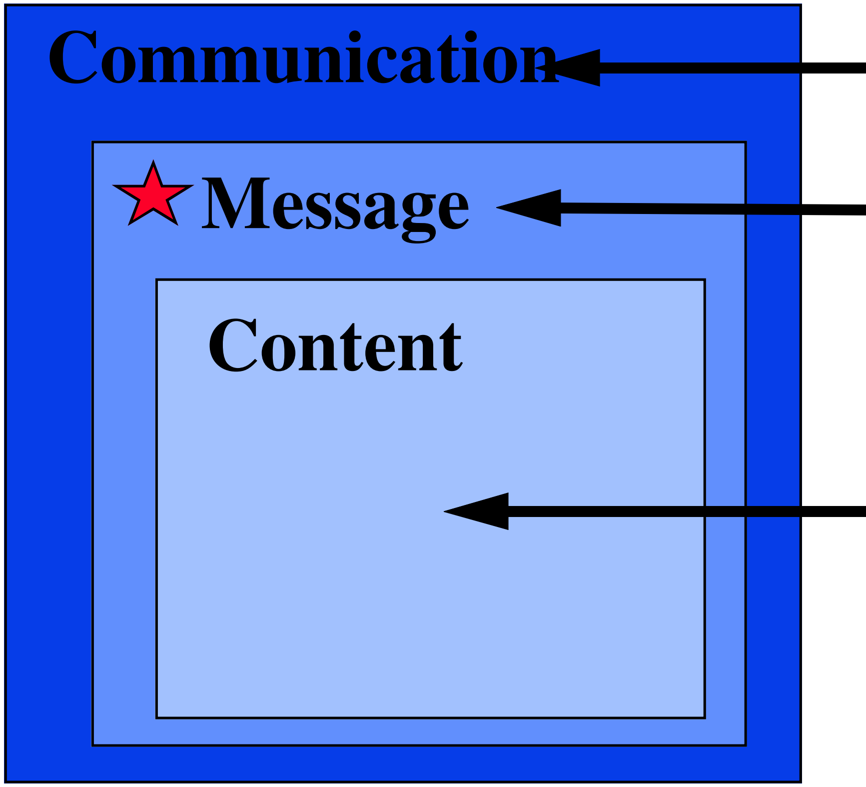
Parameter

Value

Current syntax is Lispy with provisions for binary data and structures.

KQML is a layered language

Logically, a KQML message is a *content expression* enclosed in a *speech act* inside a *communication packet* .



- Mechanics of communication
- Logic of communication (performative or speech act)
- Content of communication (in agreed upon language, e.g., KIF, KRSL, etc.)

KQML
Communication and Content Layers

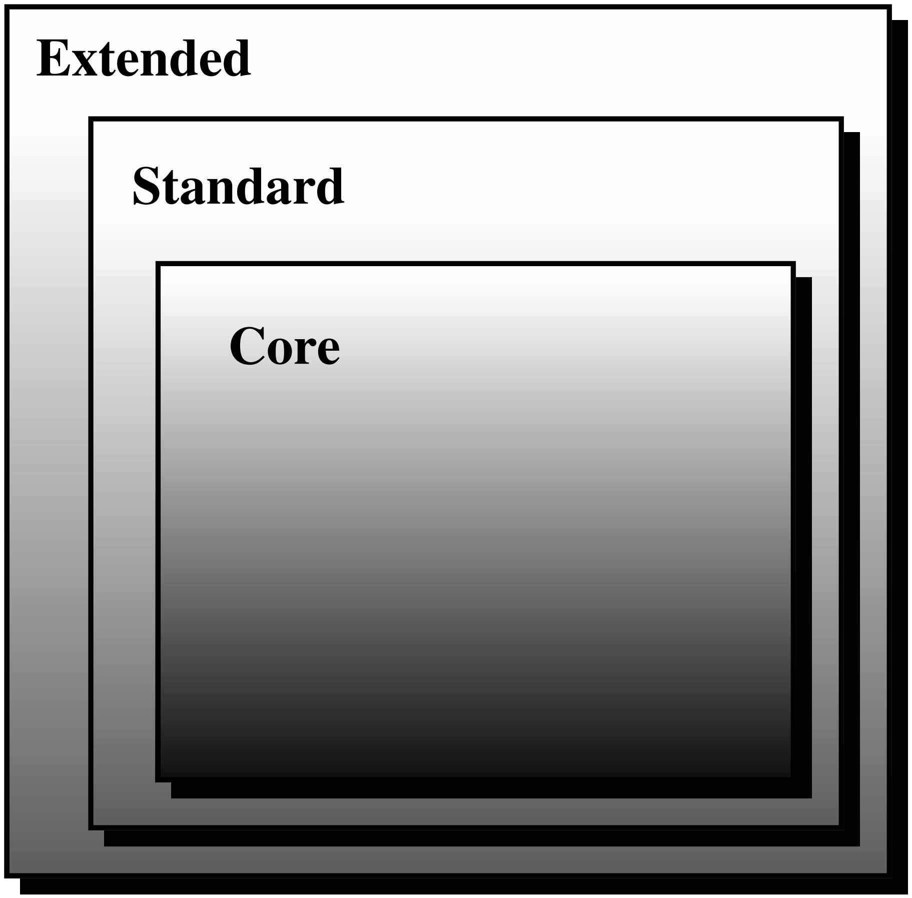
- **Communication layer**
  - Low level, packet oriented, not stream oriented.
  - Currently built on TCP/IP streams but could be built on RPC, UDP, Email, or other packet communication media.
- **Content layer**
  - Language variable -- specified in message layer
  - Two agents can communicate in any language they can agree on
  - KIF is a good default

KQML Message Layer

- Each KQML expression is a speech act described by a *performative*.
- Independent of the content language or communication layer.
- Based on an extensible set of well defined performatives such as ***ASK, TELL, SUBSCRIBE*****.**

```
(ASK :content (geoloc lax (?longitude ?latitude))
     :language LOOM
     :ontology ARPI-GEO)
```

KQML Performatives

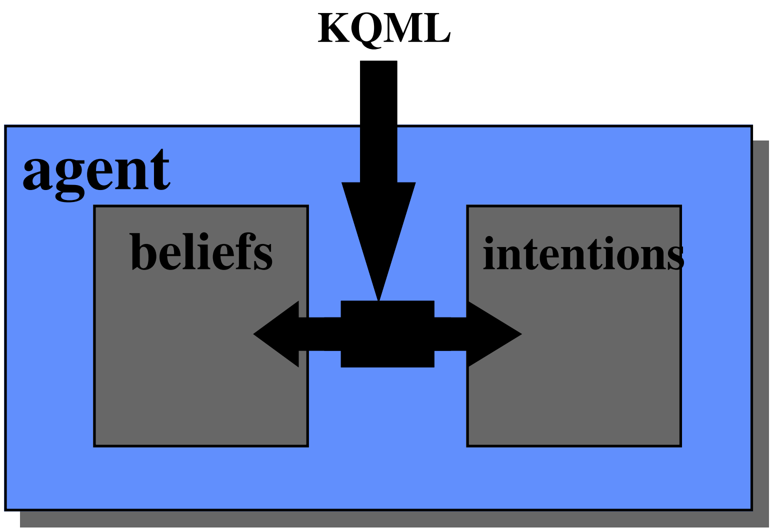


- **Core Performatives**
  A small set of primitive performatives formally defined which are used to define others.
- **Standard Performatives**
  Reserved names for performatives that most agents will want to handle.
- **Extended Performatives**
  Groups of agents can define and use extensions.

Semantics of Core Performatives

KQML agents are modeled as having

- an information store (beliefs)
- a goal store (intentions)



- **TELL** - add content to the information store.
- **UNTELL** - remove content from the information store.
- **ACHIEVE** - add content to the goal store.
- **UNACHIEVE** - remove content from the goal store.

Some KQML Performatives

- **Basic query performatives**:
  evaluate, ask-if, ask-in, ask-one, ask-all
- **Multi-response query performatives**:
  stream-in, stream-all
- **Response performatives**:
  reply, sorry
- **Generic informational performatives**:
  tell, achieve, cancel, untell, unachieve
- **Generator performatives**:
  standby, ready, next, rest, discard, generator
- **Capability-definition performatives**:
  advertise, subscribe, monitor, import, export
- **Networking performatives**:
  register, unregister, forward, broadcast, route

KQML Communication Protocols

Protocols Include:

- simple synchronous transactions, e.g. query/reply, assert/acknowledge
- multi-response performatives, e.g. generators.
- asynchronous replies, e.g. subscriptions, monitoring
- communication services, e.g. forwarding, broadcasting

KQML Supports Multiple Architectures

- Agent Based Software Engineering

  The PACT testbed uses *forward, distribute*, and other performatives to support problem decomposition in CAD/CAM.

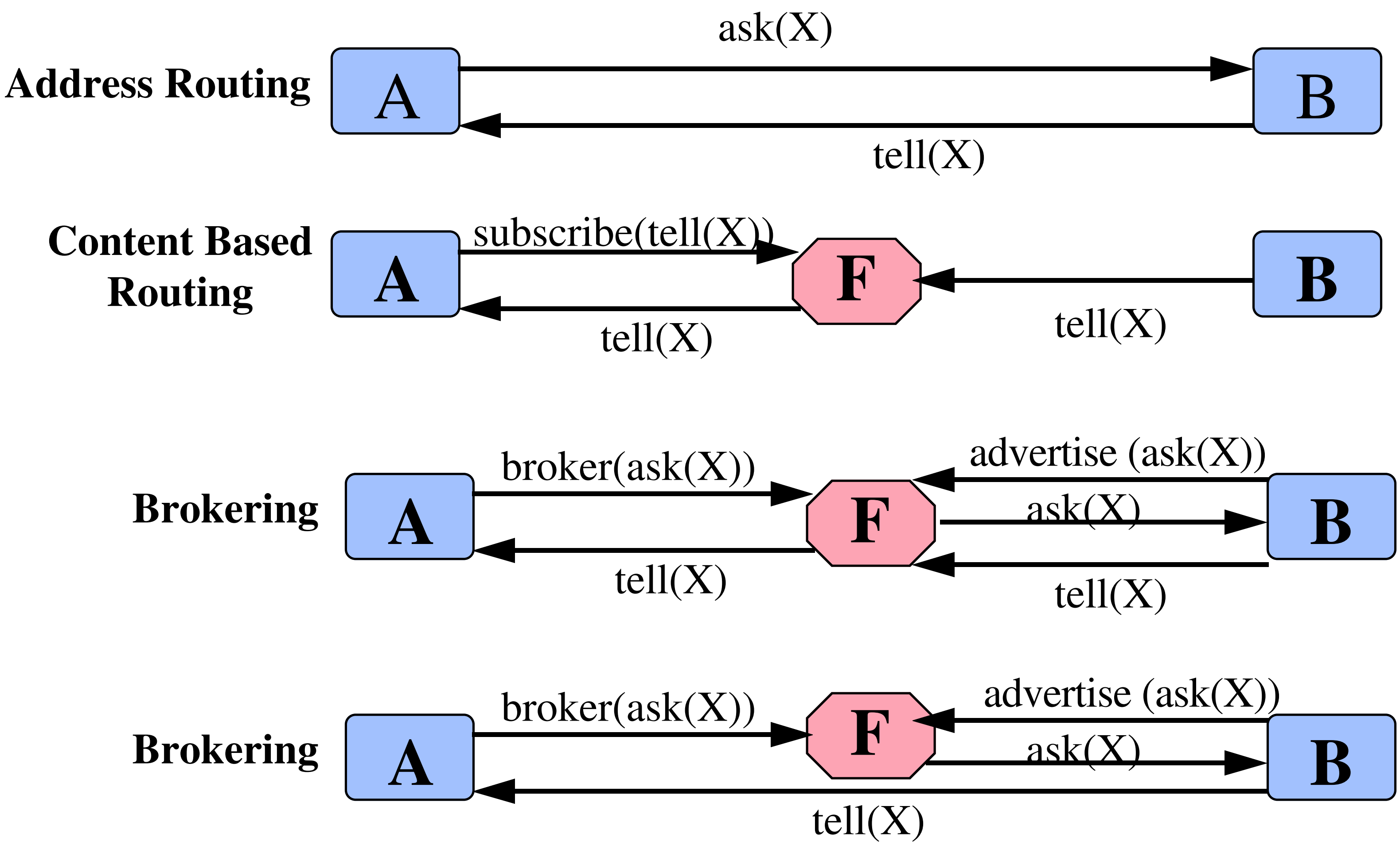
- Declarative Information Distribution

  Uses *routing agents* to support *declarative distribution* of information based on symbolic names of agents and declarations of topics.

- Information Bus

  Built on *publish and subscribe* primitives and the advertising of fixed hierarchical topic names.

Facilitation Services



Content Based Routing of Messages

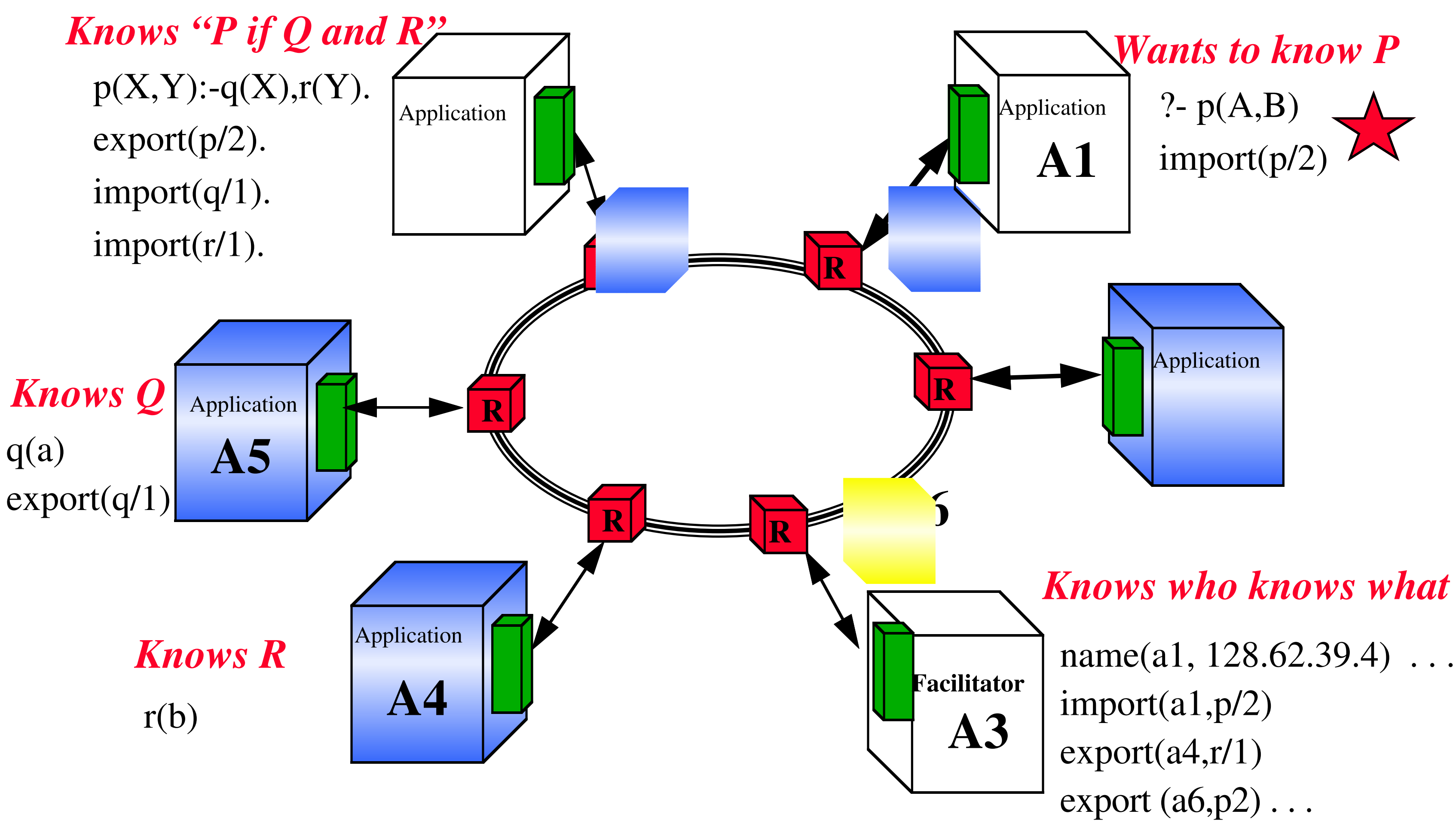
- **Applications send descriptions of their information needs and services to their local facilitator:**

```
(import
   :from (drpi.bbn.com 42)
   :content (ask :content (location ?x ?y)
   :language KRSL
  :ontology (and geoloc  planning))

(export
   :from (mapsrv.unisys.com 36)
   :content (tell :content (plan_update ?plan ?mod)
   :language KRSL)
```

- **Facilitators communicate with each other to share information or forward unanswerable queries.**

Content Based Routing



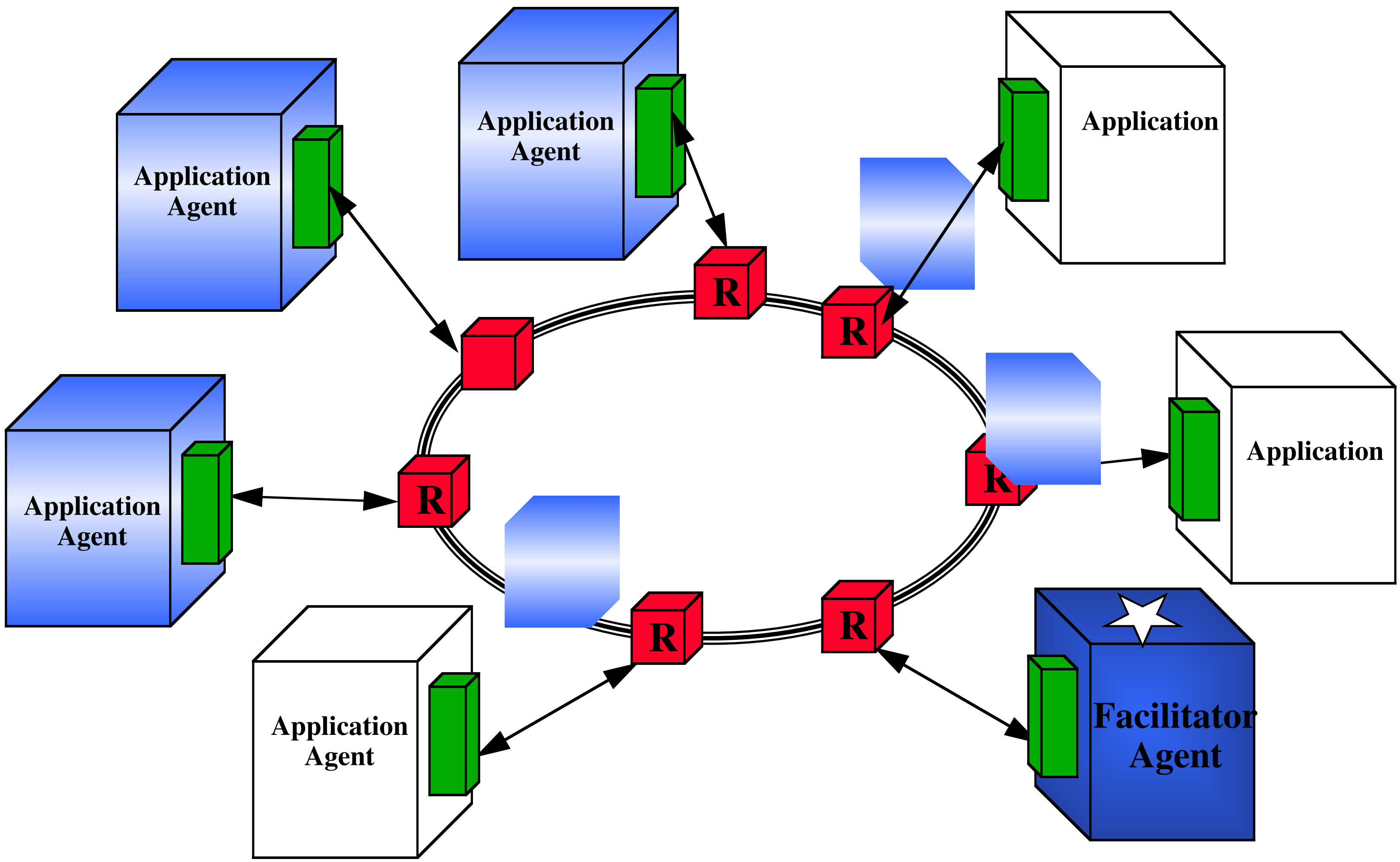
IMPLEMENTATIONS,

SOFTWARE ARCHITECTURES

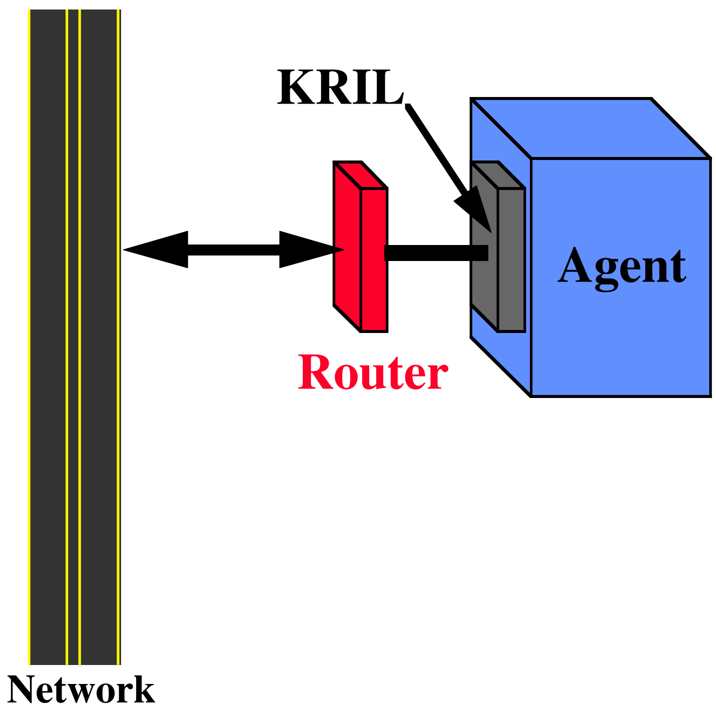
AND

APPLICATIONS

Unisys KQML Architecture

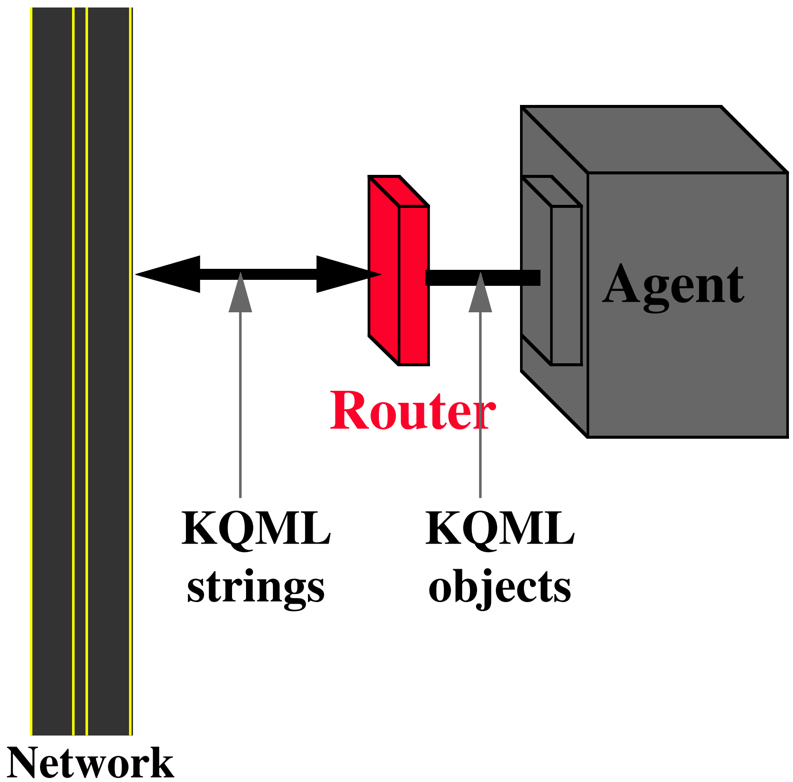


Unisys KQML Architecture



- **Routers**
  - Content independent message router.
  - One per agent - handles sending and asynchronous receiving of messages.
- **KRILs**
  - Knowledge Representation Interface Library.
  - API makes application look like a KB.
- **Facilitator Agents**
  - One per site - provides local DB of symbolically named services.
  - Eventually to hold and use a DB of topics and ontologies.
  - Can provide mediation and translation services.

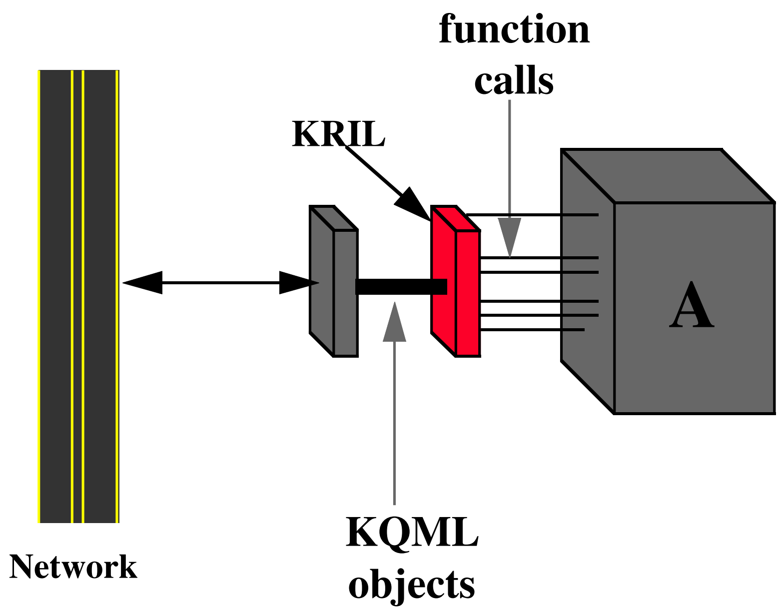
KQML Router



The Router provides an easy-to-use link between application and network.

- Provides both client and server capabilities
- Manages multiple simultaneous connections
- Handles some KQML interactions autonomously

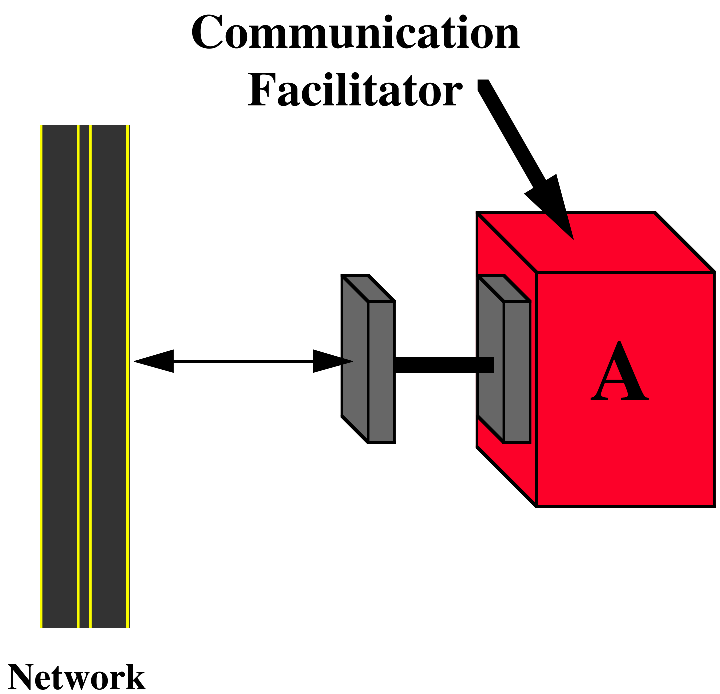
KQML KRIL



The KRIL is part of the application and has access to its internals.

- Provides internal access points to which the router delivers incoming messages.
- Analyzes outgoing messages for appropriate domain tagging and routing.
- Provides application specific interface and procedures for communication access.

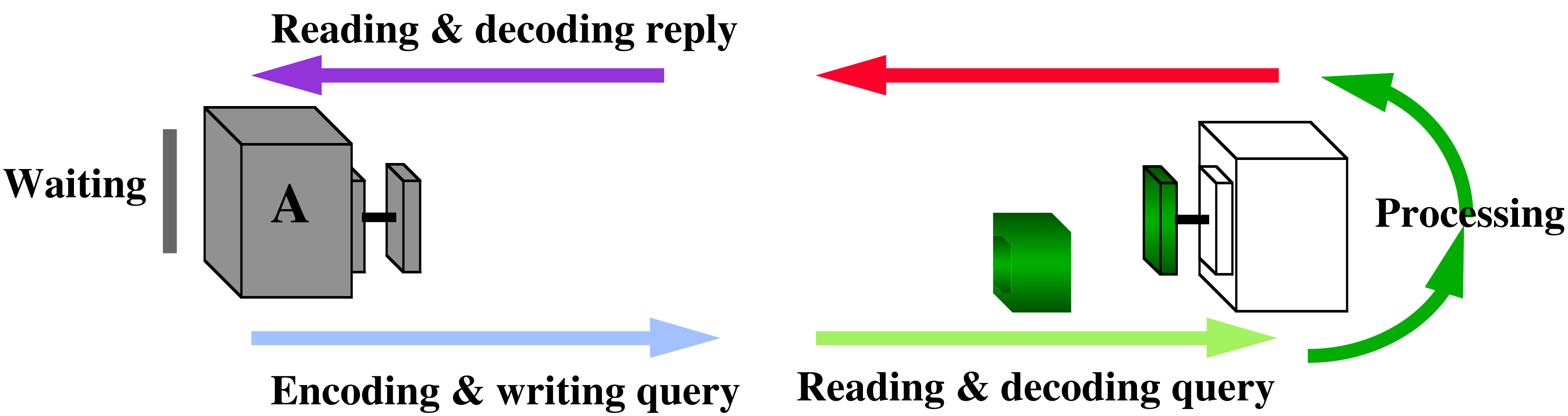
Facilitator Agents



A Communication Facilitator is an agent that performs various useful services, e.g.

- Maintains a registry of service names.
- Forwards messages to named services.
- Routes messages based on content.
- Provides "matchmaking" between information providers & clients.
- Provides mediation and translation services.

KQML Performance Model



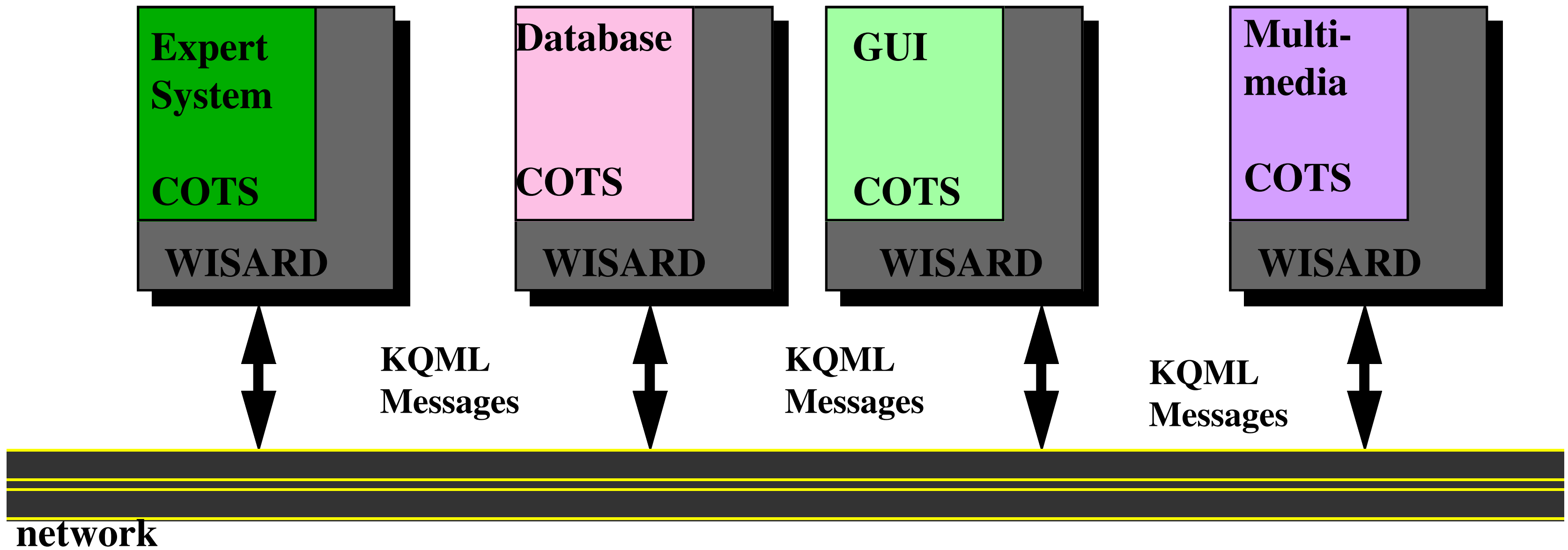
Agent A sends a KQML message to agent B and gets a response.

- Model has enabled us to tune software
- Compression enhancements
- ~ 20 round-trip empty messages/second

KQML Developers and Users

- Unisys, *Planning Initiative, IWSDB*
- Palo Alto Collaboration Testbed (PACT)
- Lockheed AI Center, *SHADE, Shared Knowledge-Based Technology for the Re-engineering Problem*
- The Aerospace Corp., *WISARD - A Distributed Architecture for COTS Integration*
- Lockheed AI Center, *Cosmos, A System for Accumulating and Managing Distributed Design Knowledge*
- ISX, *IWSDB, Integrated Weapon Systems Database*
- Stanford University
- University of Maryland

WISARD
A Distributed Architecture for COTS Integration



- Support the use of Commercial-off-the-shelf software in systems by wrapping them in a KQML speaking shell.
- Computer Systems Division, The Aerospace Corporation

KQML in the Future

- Diverse development and user community.
- Implementation efforts are application driven.
- Investigate the use of KQML in hypertext systems, digital libraries, NII, databases, ...
- Changes in the standards landscape: KQML on OMG's CORBA?

Summary and Conclusion

- The need for intelligent information agents (Knowbots)
- KQML as the language of Knowbots
- KQML supports many interesting agent architectures.
- KQML can be used for efficient communication
- Experimentation and experience will evolve the language

For More Information

- Mailing lists
  - kqml@isi.edu -- general discussion
  - kqml-users@isi.edu -- users and developers
- Information servers
  - **ftp:** ftp.cs.umbc.edu under pub/kqml
  - **www:** http://ftp.cs.umbc.edu/kqml.html
  - **gopher:** gopher.cs.umbc.edu under kqml
- Working group co-chairs
  - Tim Finin -- finin@cs.umbc.edu
  - Jay Weber -- weber@eitech.com