



# XML Spaces Beyond Web Services

Patrick Thompson  
Chief Architect

Rogue Wave Software  
thompson@roguewave.com

# XML Tuple Spaces: Summary

- Secure XML communications over the Internet
- Primary characteristics:
  - Asynchrony
  - Security
  - Flexible document exchange (1-1, 1-many, many-many)
  - Identity-based addressing
  - Simplicity



# Many Applications

- General infrastructure technology with many uses
  - Asynchronous web services; “SOAP Spaces”
  - Application integration
  - E-procurement
  - Occasionally connected devices
  - Document routing

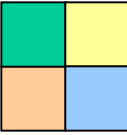






## Available Today

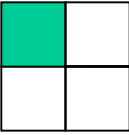
- “Ruple”
- Hosted space, downloadable space, code, examples, whitepapers available on Rogue Wave’s Technology Access Center:

[www.roguewave.com/developer/tac](http://www.roguewave.com/developer/tac)

# Today's Discussion Areas



-  XML Spaces overview
-  XML Spaces highlights
-  Scenarios using XML Spaces
-  Conclusion



# XML Spaces Overview

# XML Spaces: Distributed Communication



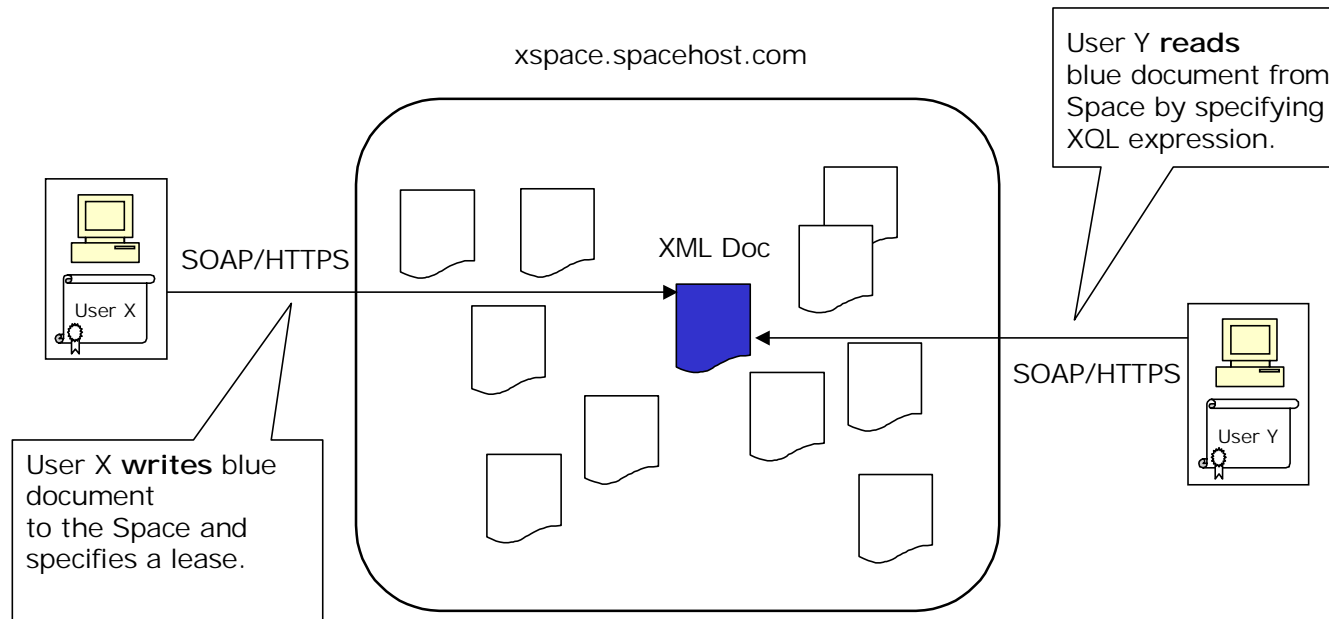
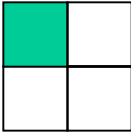
- Used for exchanging XML documents over the Internet (or intranet or extranet)
- Highlights:
  - Simple
  - Secure
  - Many-to-many interactions
  - Asynchronous
  - Document centric

# XML Spaces



- XML Spaces brings together:
  - Tuple spaces communication paradigm
  - Internet technologies (HTTP, DNS)
  - Web services (SOAP)
  - Security (SSL, X.509 digital certificates)
  - XML

# XML Document Space

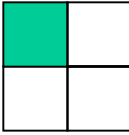


# XML Spaces Server

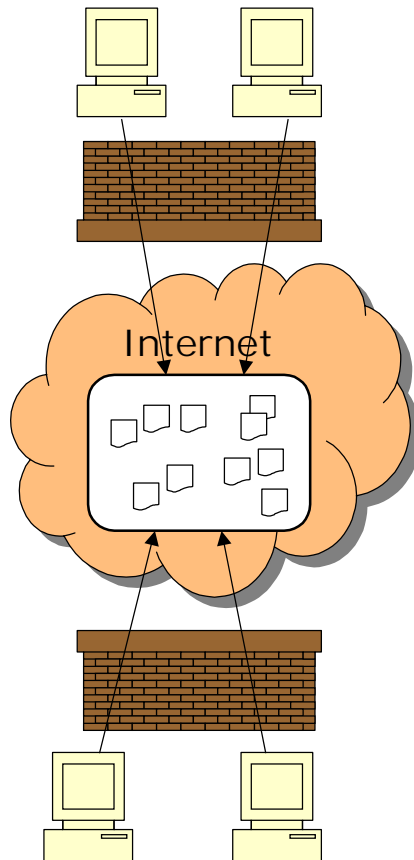


- An XML Spaces server is an intermediary between XML Spaces applications
- XML Spaces servers may be deployed:
  - On the internet for interactions between organizations
  - In the DMZ for interactions between an organization and the outside world
  - Behind a firewall for interactions within an organization

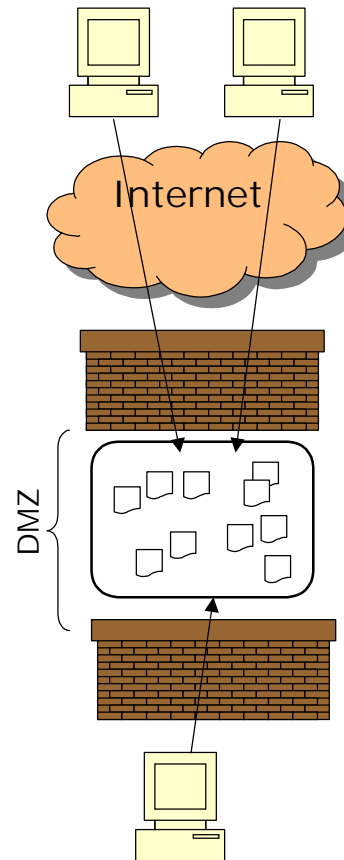
# XML Spaces Deployment



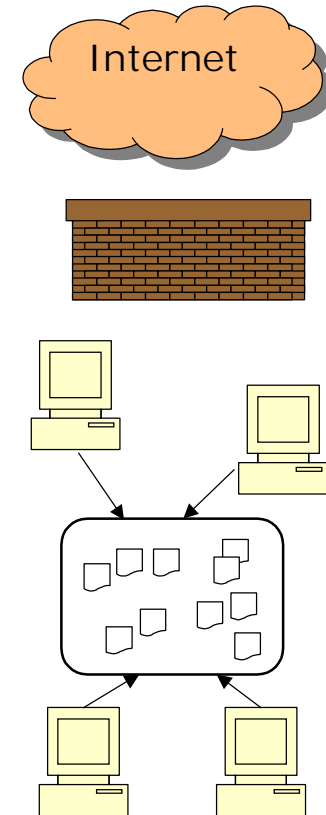
Internet Deployed Space



DMZ Deployed Space



Intranet Deployed Space

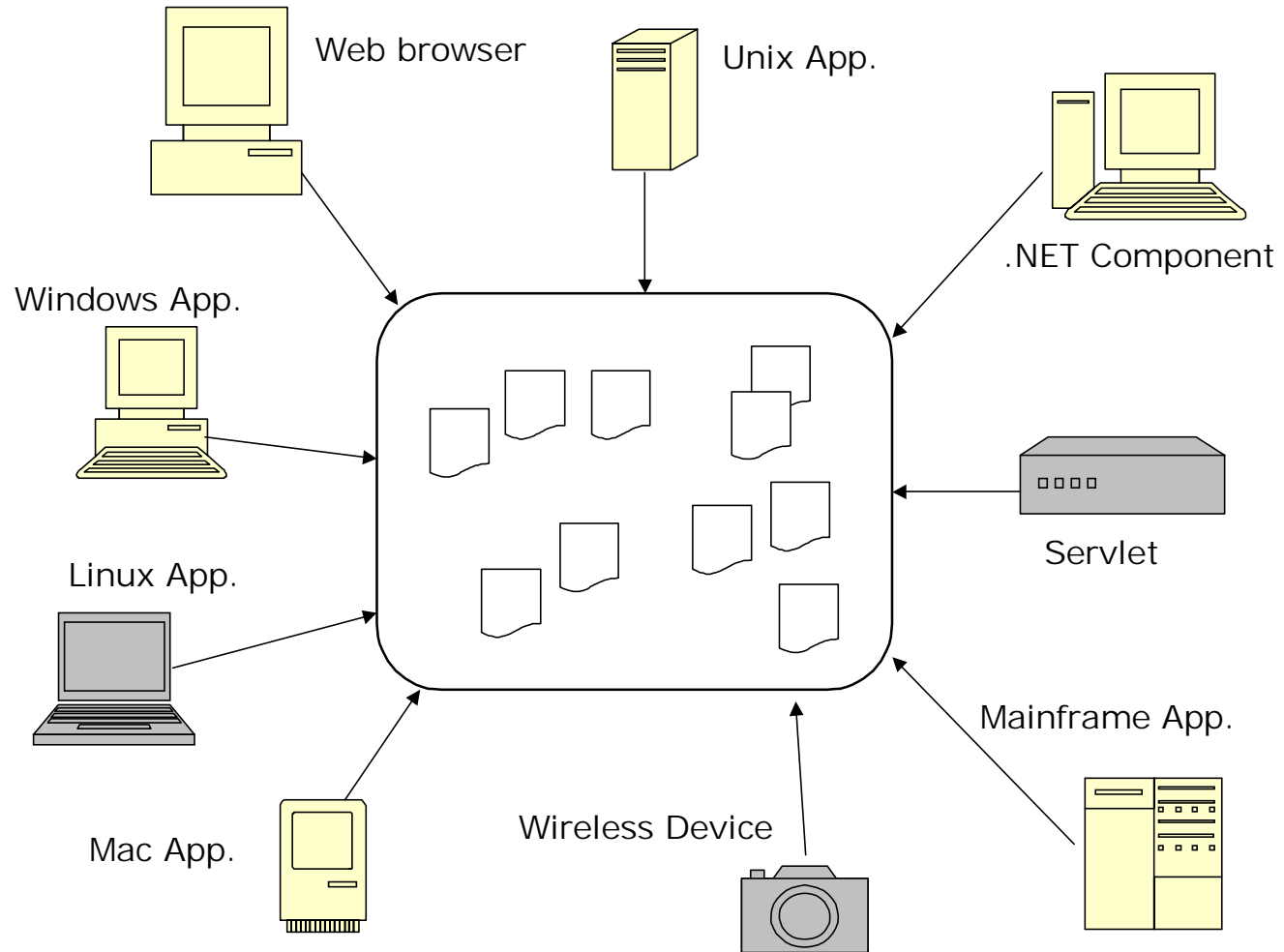
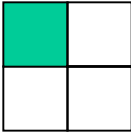


# XML Spaces Applications

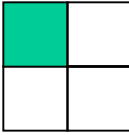


- XML Spaces accessible from any environment that supports SOAP over HTTP(S)
- Applications can interact with an XML Spaces server either:
  - Directly via SOAP/HTTP, or
  - Through a proxy that hides the SOAP/HTTP detail.
- XML Spaces applications interact with each other through the space.

# Possible XML Spaces Applications

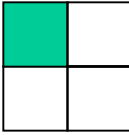


# Four Basic Operations



- **write()** – Writes a document to a space
  - **write**(xmlDoc, lease, auths);
- **read()** – Read a document from the space
  - xmlDoc **read**(xql, since);
- **readMultiple()** – Read multiple documents
  - xmlDoc **readMultiple**(xql, since);
- **take()** – Read and remove a document
  - xmlDoc **take**(xql, since);

# Write Example



*// Connect to the forum*

```
String m_url =
```

```
    http://forum.roguewave.com/forum/servlet/ForumServlet";  
Forum f = ForumFactory.createForum(m_url);
```

*// Set security policy*

```
Authorizations auths = new AuthorizationsImpl();  
auths.allowAllReaders(true);
```

*// Write a document to the forum*

```
String xmlDoc = "<color>purple</color>";  
int myLease = 604800; // 1 week  
int lease = f.write(xmlDoc, myLease, auths);
```

# Read Example



*// Connect to the forum:*

```
String m_url =
```

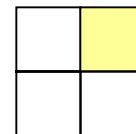
```
    "http://forum.roguewave.com/forum/servlet/ForumServlet";
```

```
Forum f = ForumFactory.createForum(m_url);
```

*// Query for document containing "purple" as value of <color>*

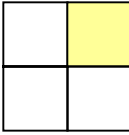
```
String query = "/color$eq$" + "purple\"";
```

```
Entry entry = f.read(query, 0);
```



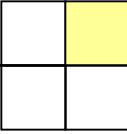
# XML Spaces Highlights

# Internet



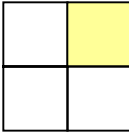
- XML Spaces is built on top of globally deployed Internet infrastructure
- Internet Standards based:
  - HTTP
  - SSL
  - DNS
  - SOAP

# Scalable



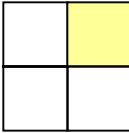
- The XML Spaces servers can be very scalable.
  - The server is stateless
  - Documents are independent of each other
- This allows for XML Spaces to be deployed into fault tolerant and scalable Web server farm environments

# XML



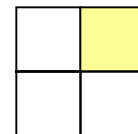
- XML Spaces is XML document centric
  - Documents are passed and stored natively
- Any XML document or XML fragment may be written to the space
  - No setup required to support new document types
  - Very easy to create or change document exchange patterns
- XQL is used for associative lookups in the space

# Security



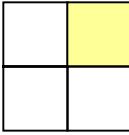
- XML Spaces are deployed on the Internet
  - Security is a key feature
- XML Spaces uses:
  - HTTPS (HTTP and SSL)
  - X.509 digital certificates
- Documents may be made available to all comers
- Or a document may be specifically targeted to one or more recipients based on digital certificates. To others, the document is invisible.

# Identity Based Addressing



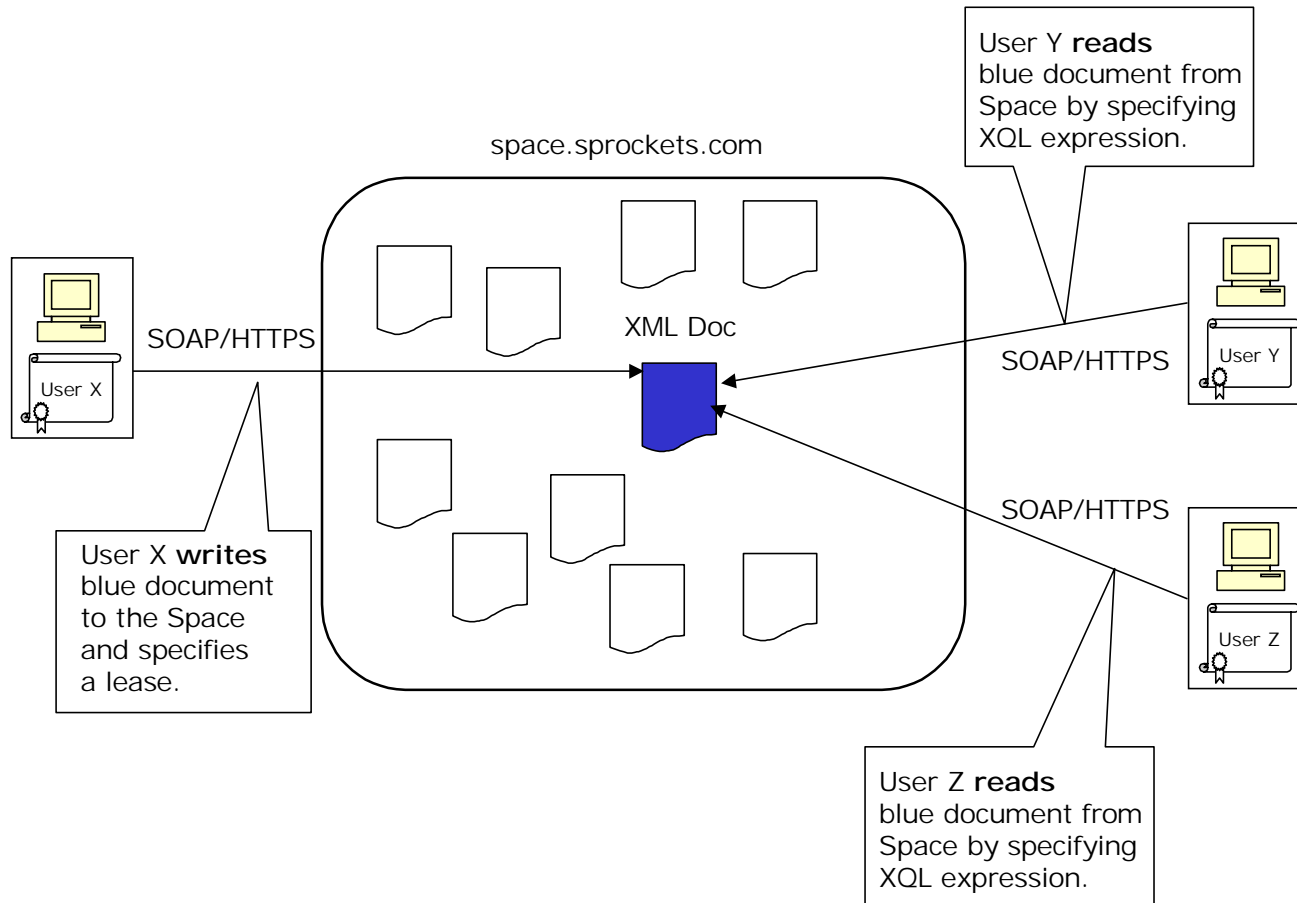
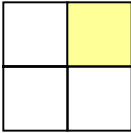
- X.509 digital certificates allow document based security
- They also provide an abstract addressing mechanism
  - Recipient isn't named by a fixed network address
  - Recipient is identified by abstract identity
  - Recipient can retrieve the document from wherever they are located

# Document Exchange Patterns



- One-to-anonymous-many
  - Broadcast-like
- One-to-named-many
  - Multicast-like
- One-to-one

# One to Many



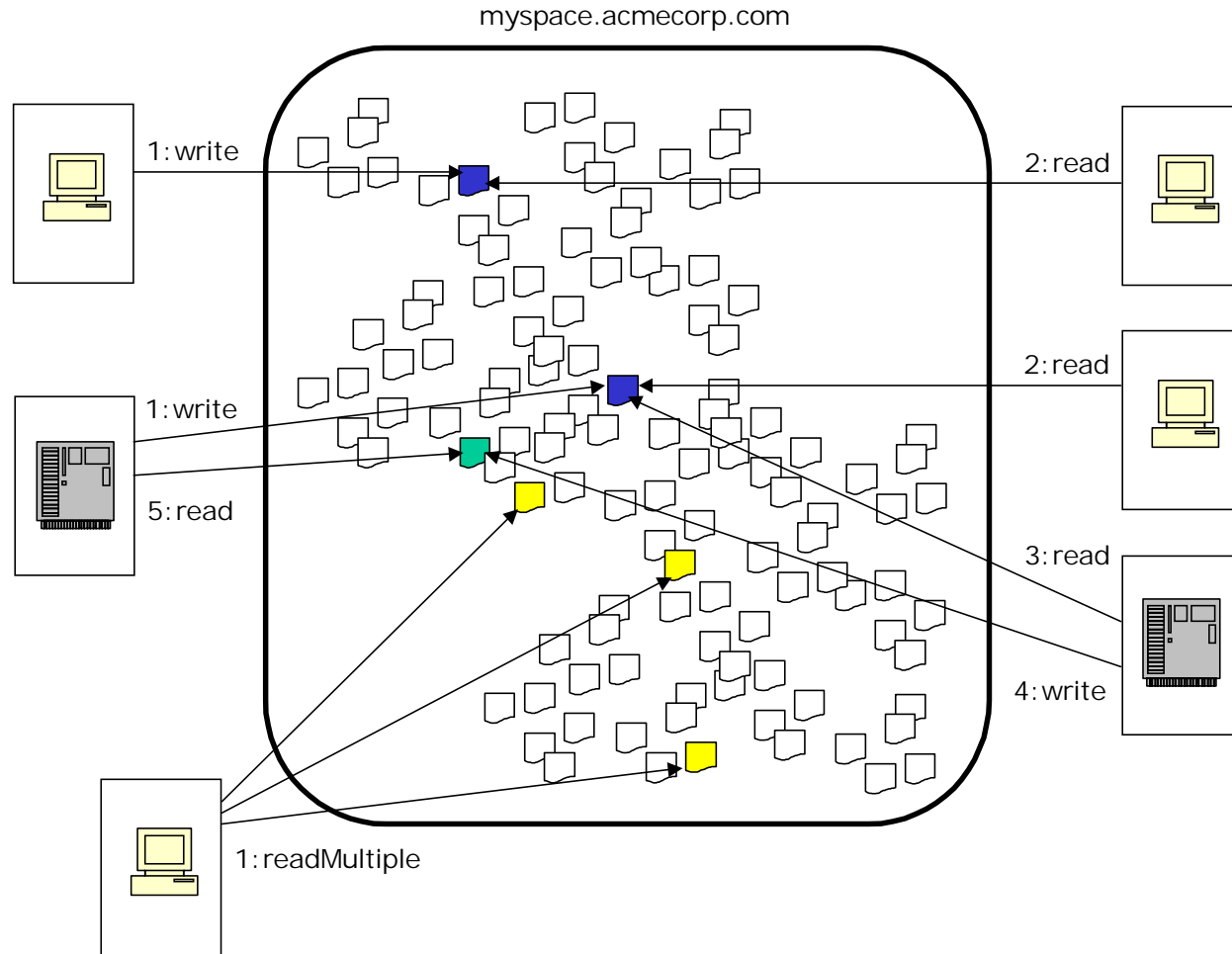
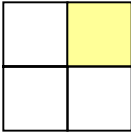
# Document Exchange Patterns

- A pattern of document exchange is a predetermined agreement between applications as to what documents will be exchanged and in what order. For example:
  - A vendor will write an RFQ document into the space, readable by multiple anonymous vendors
  - Vendors may respond to an RFQ by writing a Quote document into the space, targeted towards the buyer that issued the RFQ.
  - The buyer in response to a Quote may choose to write a Purchase Order document to the space, targeted towards the vendor that issued the Quote.
- It's obvious how to map document exchange patterns in the problem (business) domain to the solution (XML Spaces) domain.

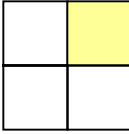
# Concurrent Document Exchange Patterns

- There can be multiple document exchange pattern instances passing through the space at one time
- The space does not know anything about document exchange patterns
- The semantics of document exchange patterns are left to the applications using the space medium

# Concurrent Document Exchange

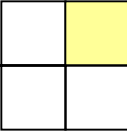


# Asynchronous



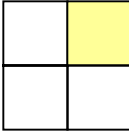
- A document is available in the space until
  - Its lease expires
  - It is removed from the space
- This allows for the sender and receiver(s) of a document to be decoupled in time
- Useful for intermittently connected devices

# Loose Coupling

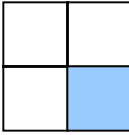


- XML Spaces applications don't rendezvous with each other in time or space
  - Don't have to know each other's address
  - Don't have to be connected at the same time
  - Separated by a “push-pull” buffer
  - They decide when they are ready to receive a document

# Simple

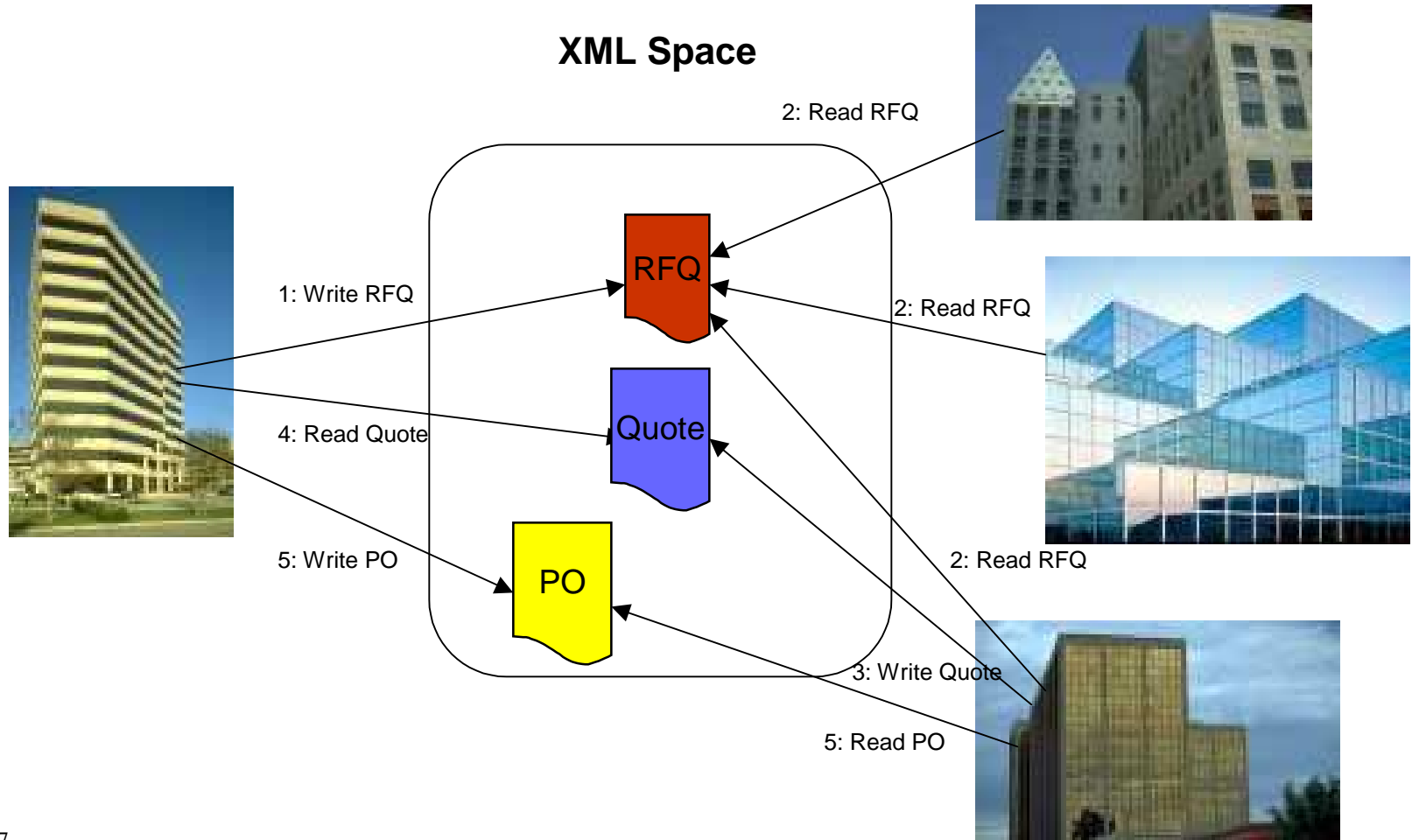
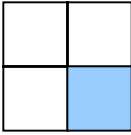


- XML Spaces supports a very simple, yet sufficient, programming model
- There are only 4 primitive operations, yet it supports very rich patterns of document exchange
- “Like the Internet, it dares to be simple”

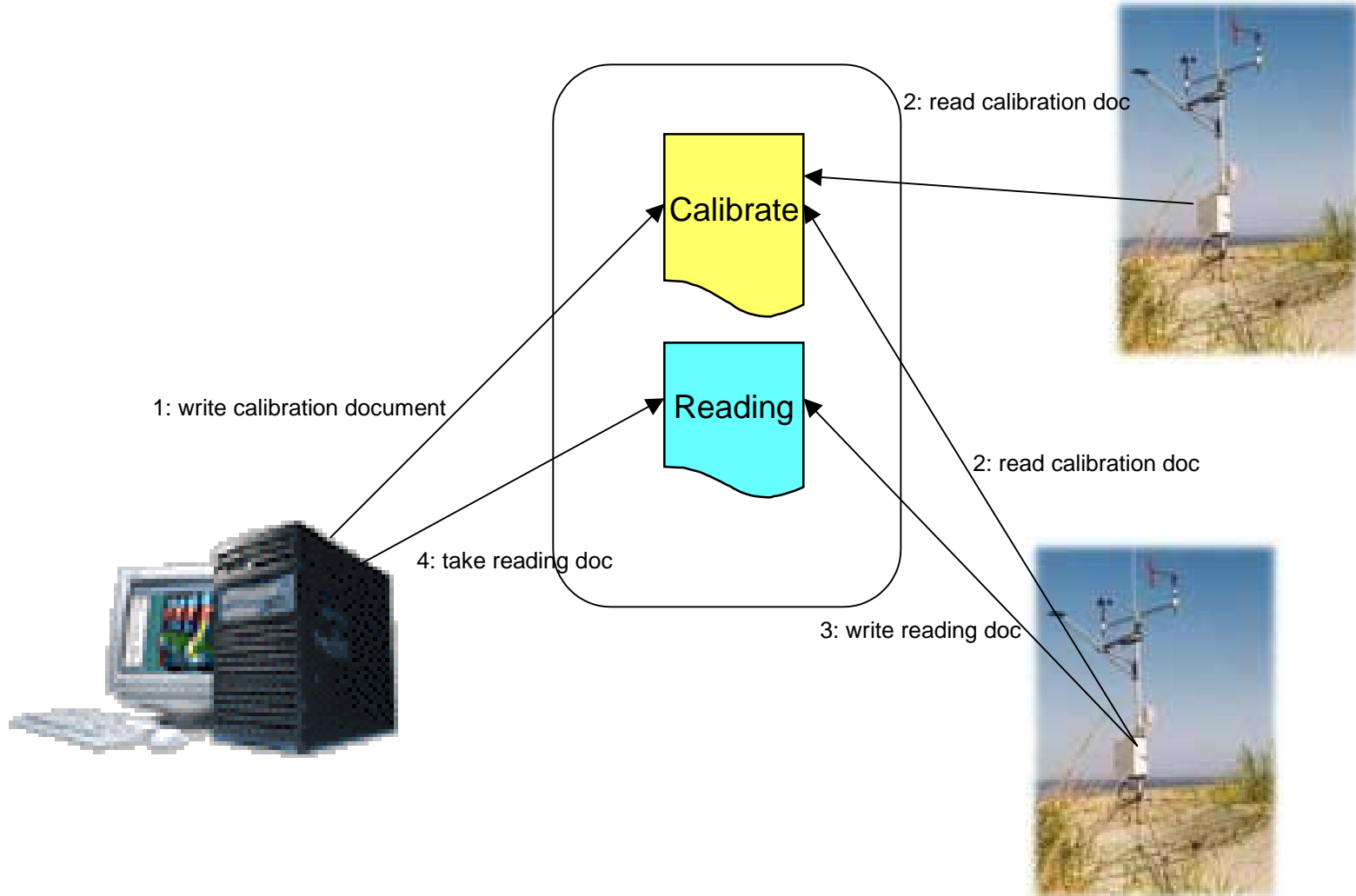
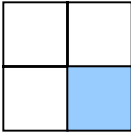


# Scenarios

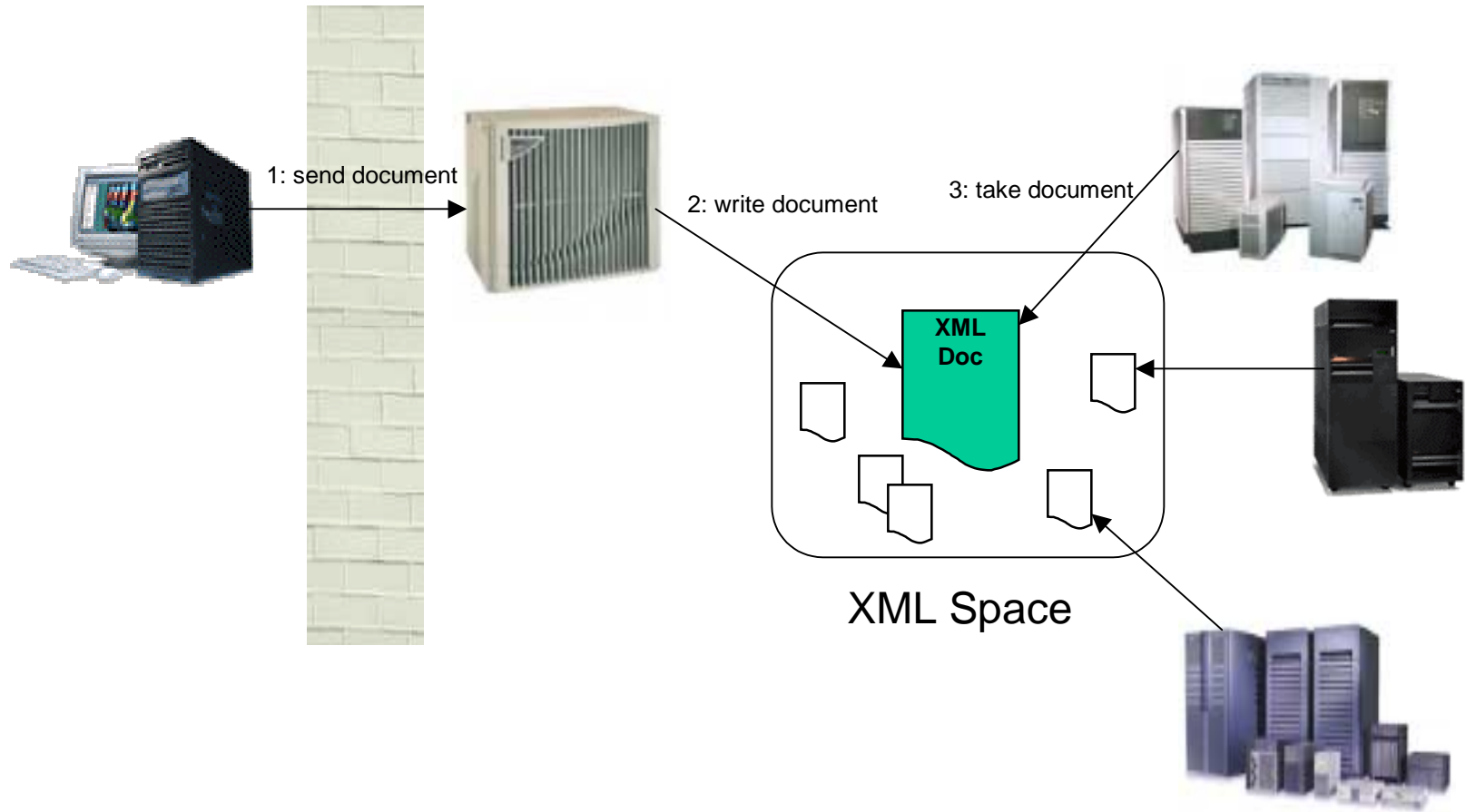
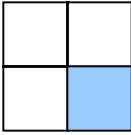
# Rich Document Exchange Patterns



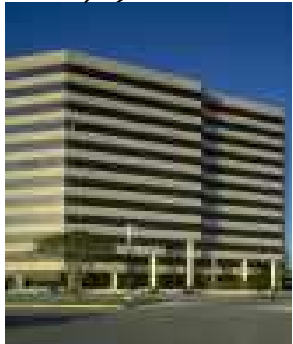
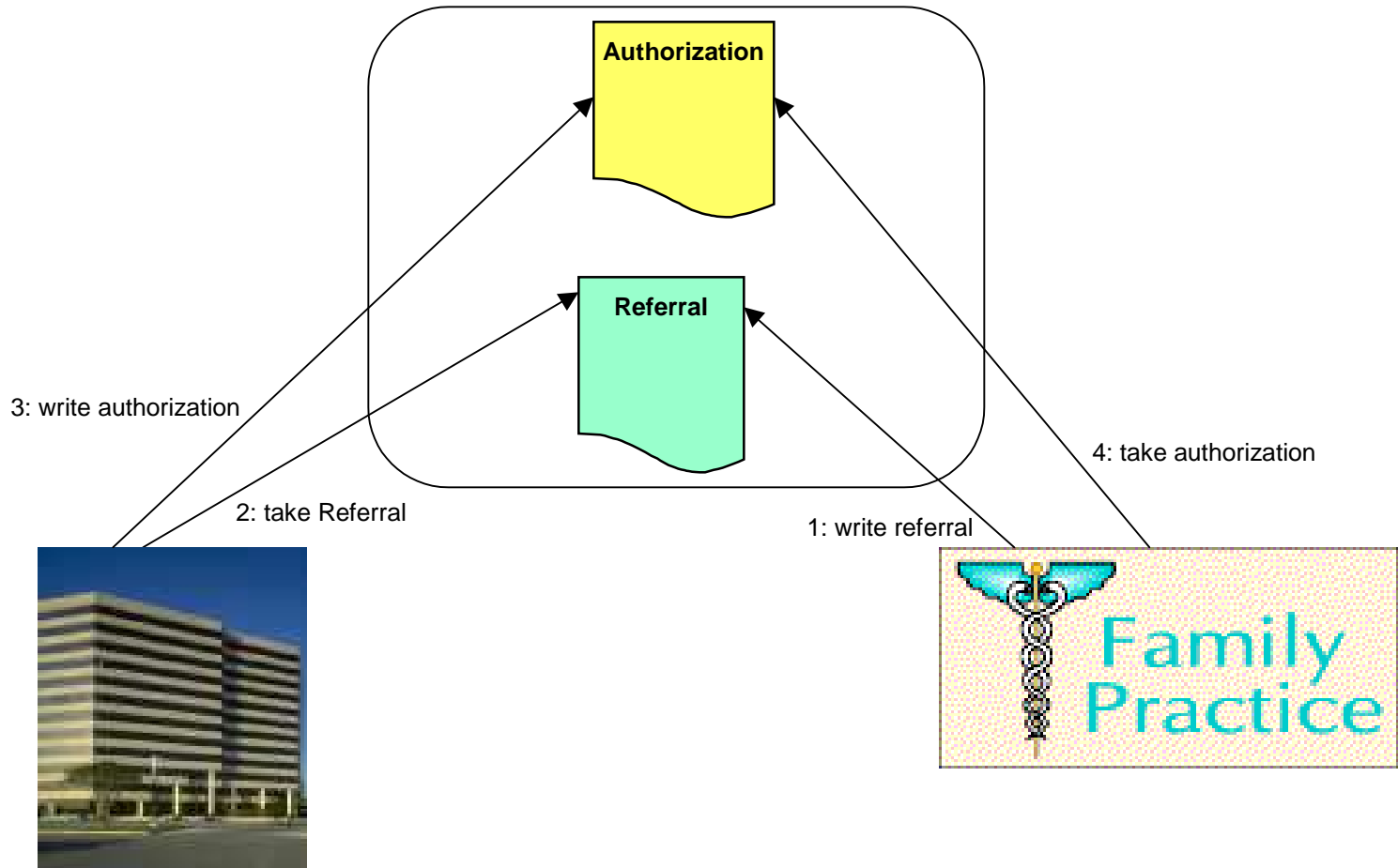
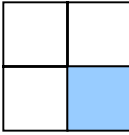
# Wireless Applications



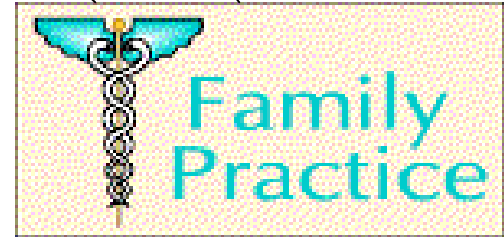
# Internal Document Routing



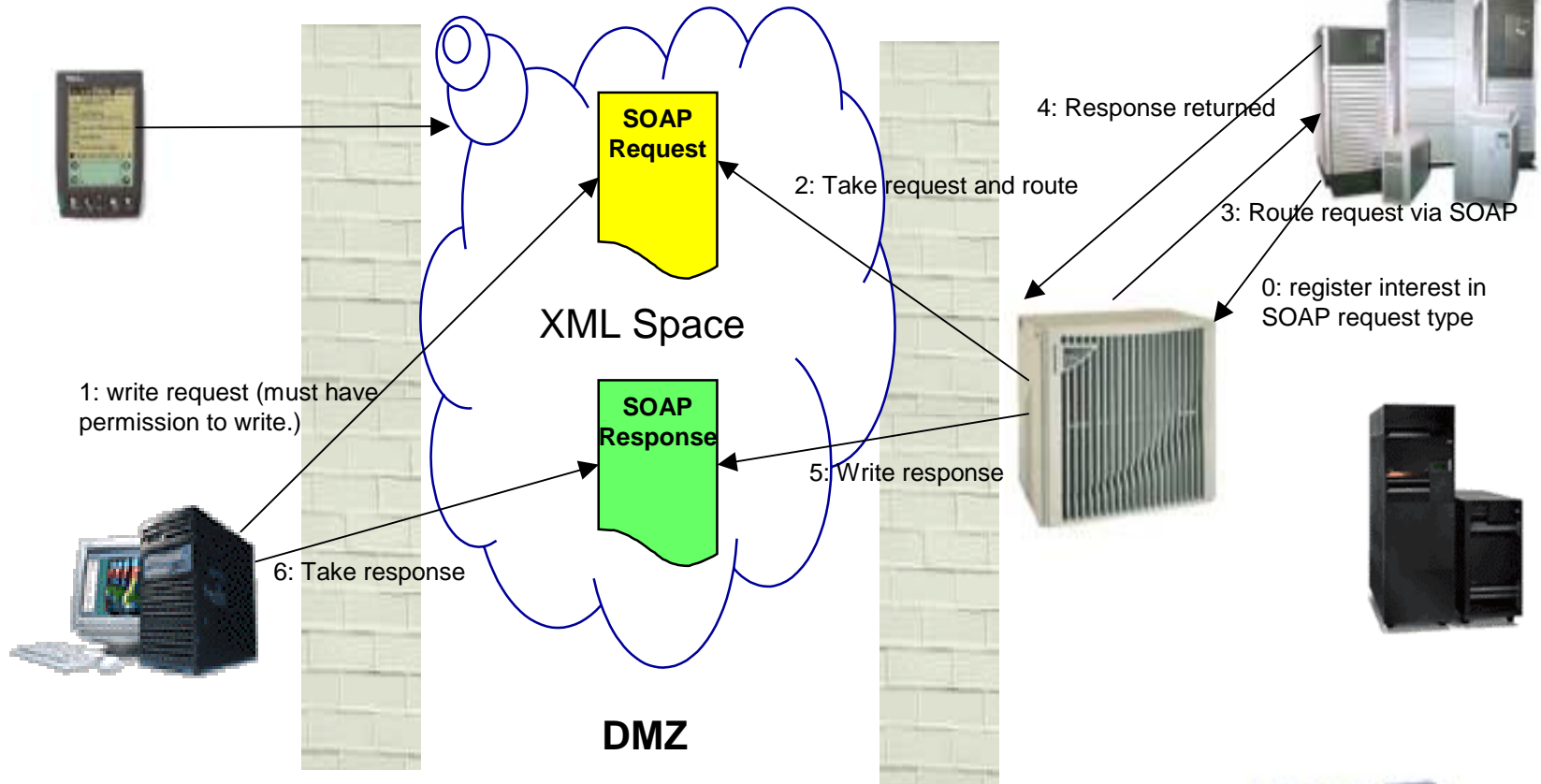
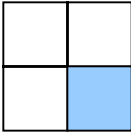
# Secure Applications



**HMO**



# SOAP Space





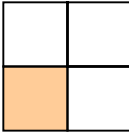
# Conclusion

## Summary

- XML tuple spaces is a new communication paradigm that brings together tuple spaces, XML, the Internet, security and web services to support rich document exchange patterns.
- Simple yet powerful.
- Extends the web services model with looser coupling identity-based addressing, asynchrony, arbitrary XML support, many-to-many interactions, and document level security.



## Available Today



- An XML Spaces implementation is available today from Rogue Wave. It is called “**Ruple**”.
- It provides XML Spaces functionality as described in this presentation, plus MIME support.
- Ruple can be downloaded from:
  - [www.roguewave.com/developer/tac/ruple](http://www.roguewave.com/developer/tac/ruple)