

The Project

A Future Vision for the Web, and
Some Tools to Get There

Stephen Uhler
Sun Microsystems Laboratories
2001-0266

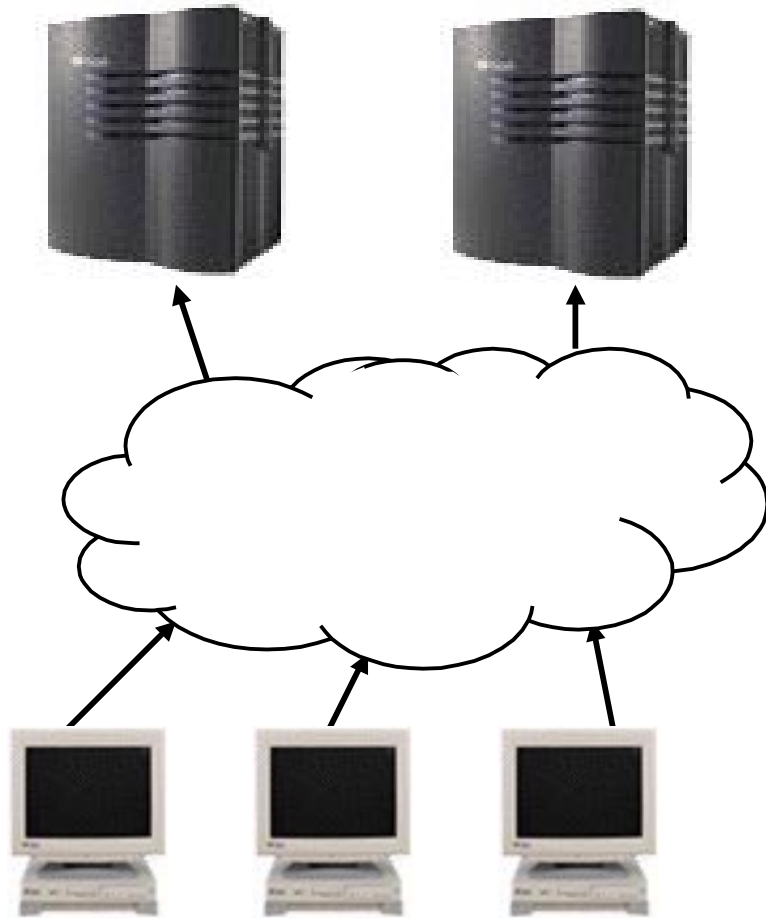
Overview

- ❖ A vision for the future of the Web
- ❖ An architecture for the vision
- ❖ Brazil - a sample implementation
- ❖ Sample applications
- ❖ Using the toolkit
- ❖ Summary

*“Research is like driving a car
at night. You can only see as far
as your headlights, but you can
make the whole trip that way.”*

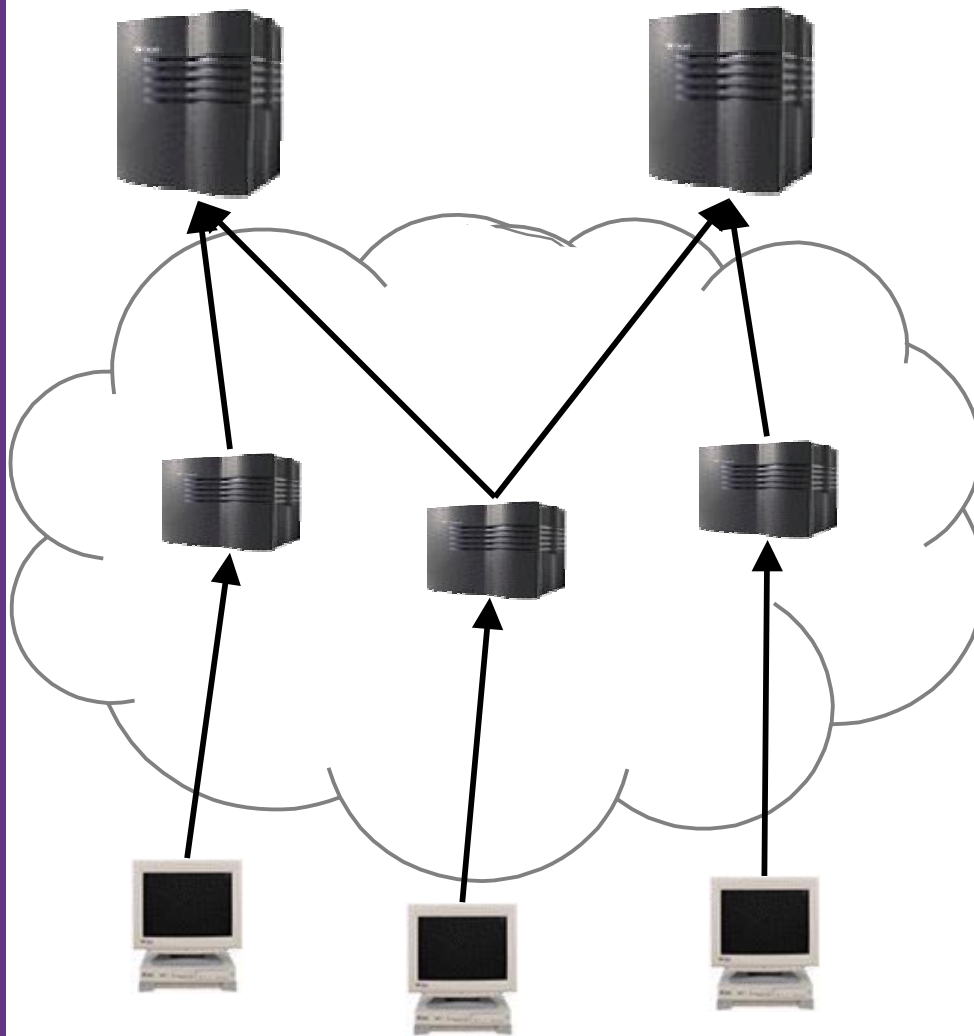
- E.L. Doctorow

The Web Today



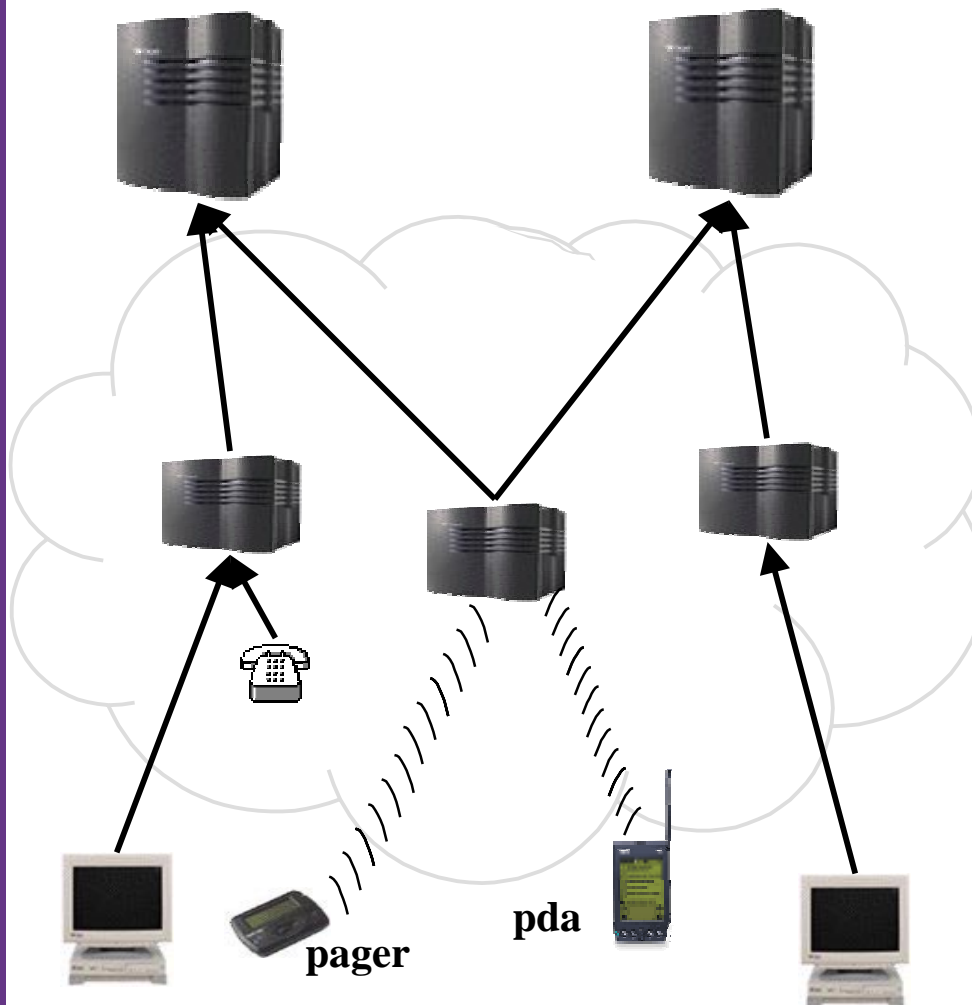
- ❖ Same client for every application
 - browser
- ❖ Content supplied by the server:
 - Web server
 - Business logic
 - Database/transactions

A Vision



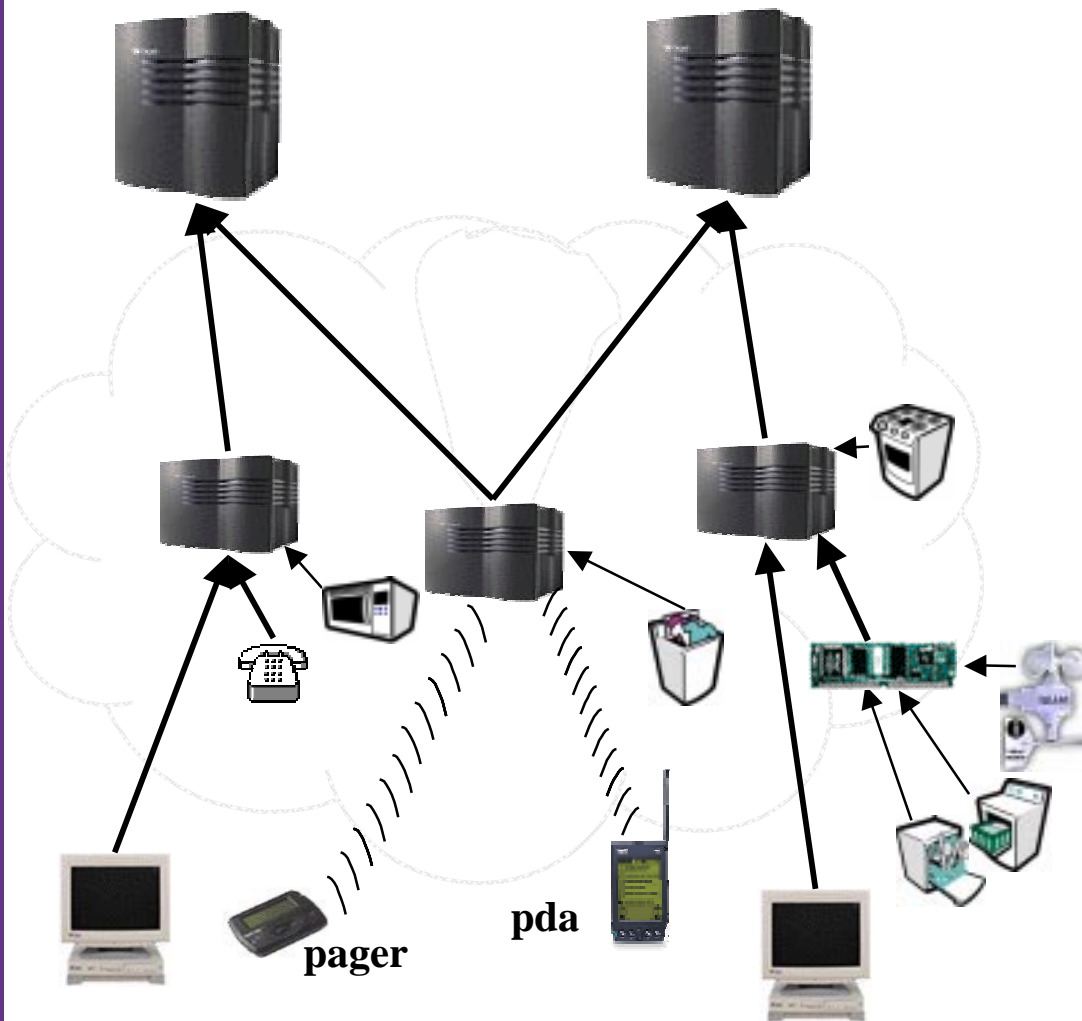
❖ Distribute the application throughout the network

A Vision



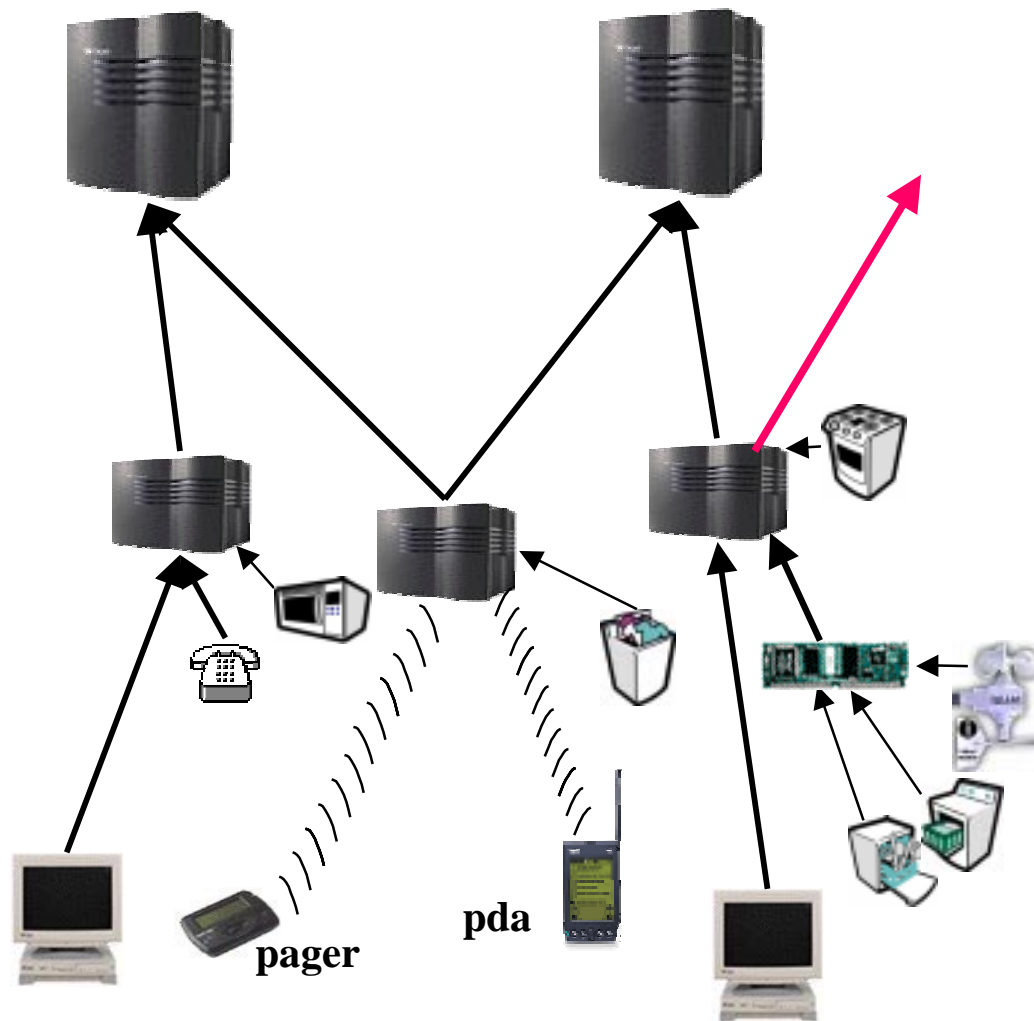
- ❖ Distribute the application throughout the network
- ❖ Support different clients

A Vision



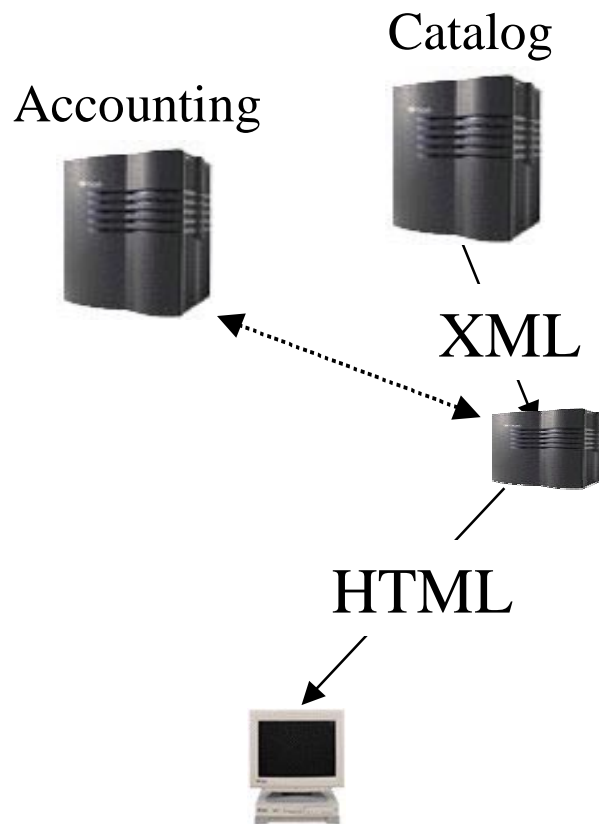
- ❖ Distribute the application throughout the network
- ❖ Support different clients
- ❖ Provide content aggregation for small data sources

A Vision



- ❖ Distribute the application throughout the network
- ❖ Support different clients
- ❖ Provide content aggregation for micro servers
- ❖ Extend reach of applications to external content

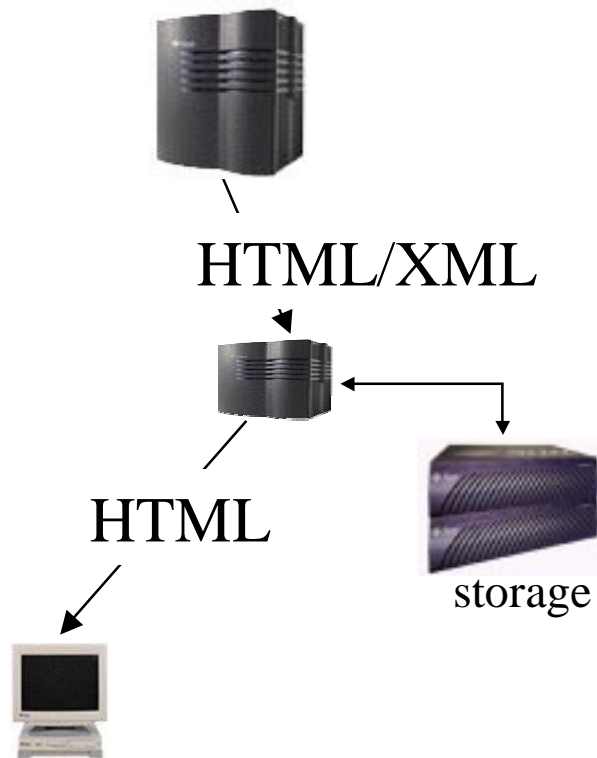
Some Examples



❖ Workflow

- Legacy application provides a generic catalog interface
- Portal proxies and filters content, mediating with the accounting system
- Company discounts, billing info added

Some Examples

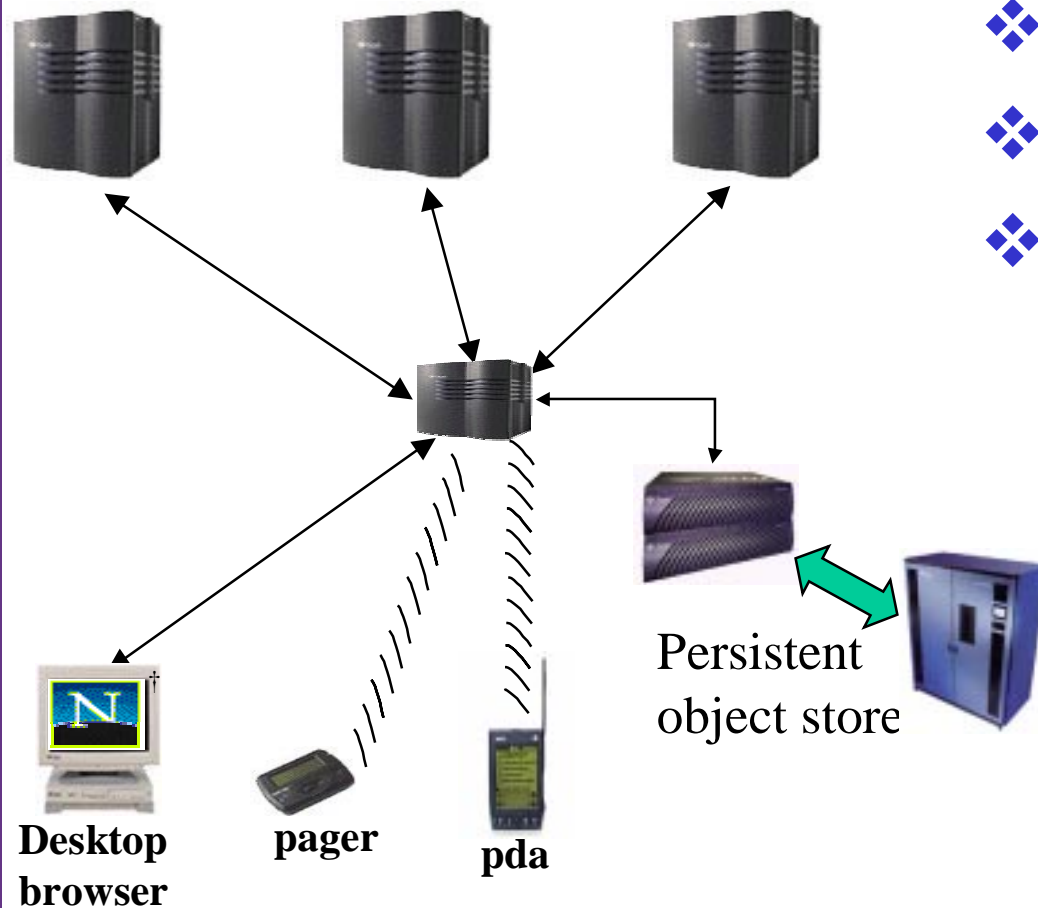


❖ Workflow

❖ Localization

- Legacy application provides a generic user interface
- Portal proxies and filters content, adding localization

Some Examples



❖ Workflow

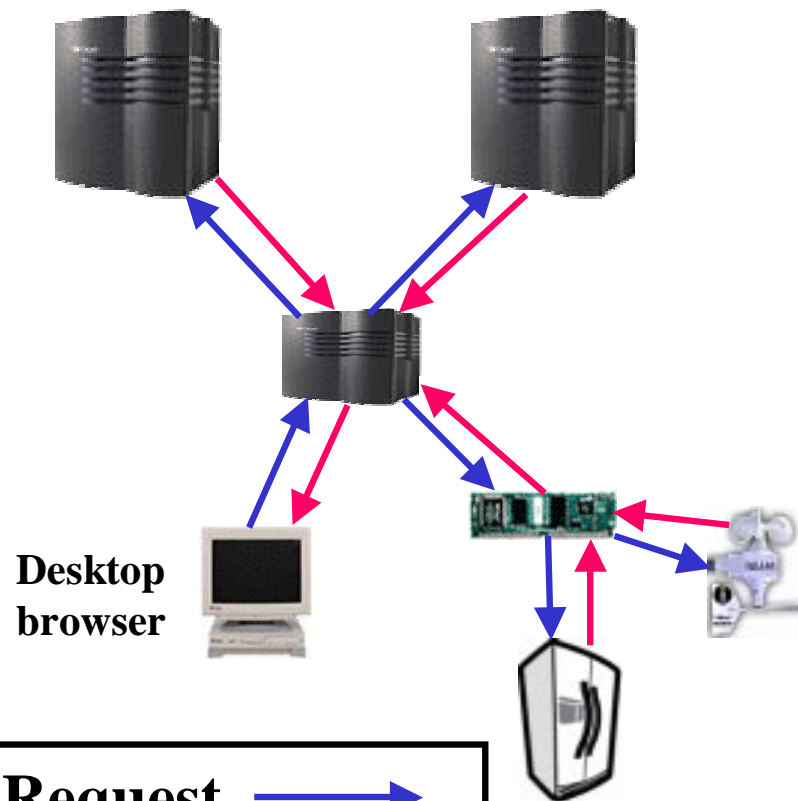
❖ Localization

❖ Personalization

➤ Content filtered by user preferences stored on portal.

➤ Transcoding for alternate GUI's

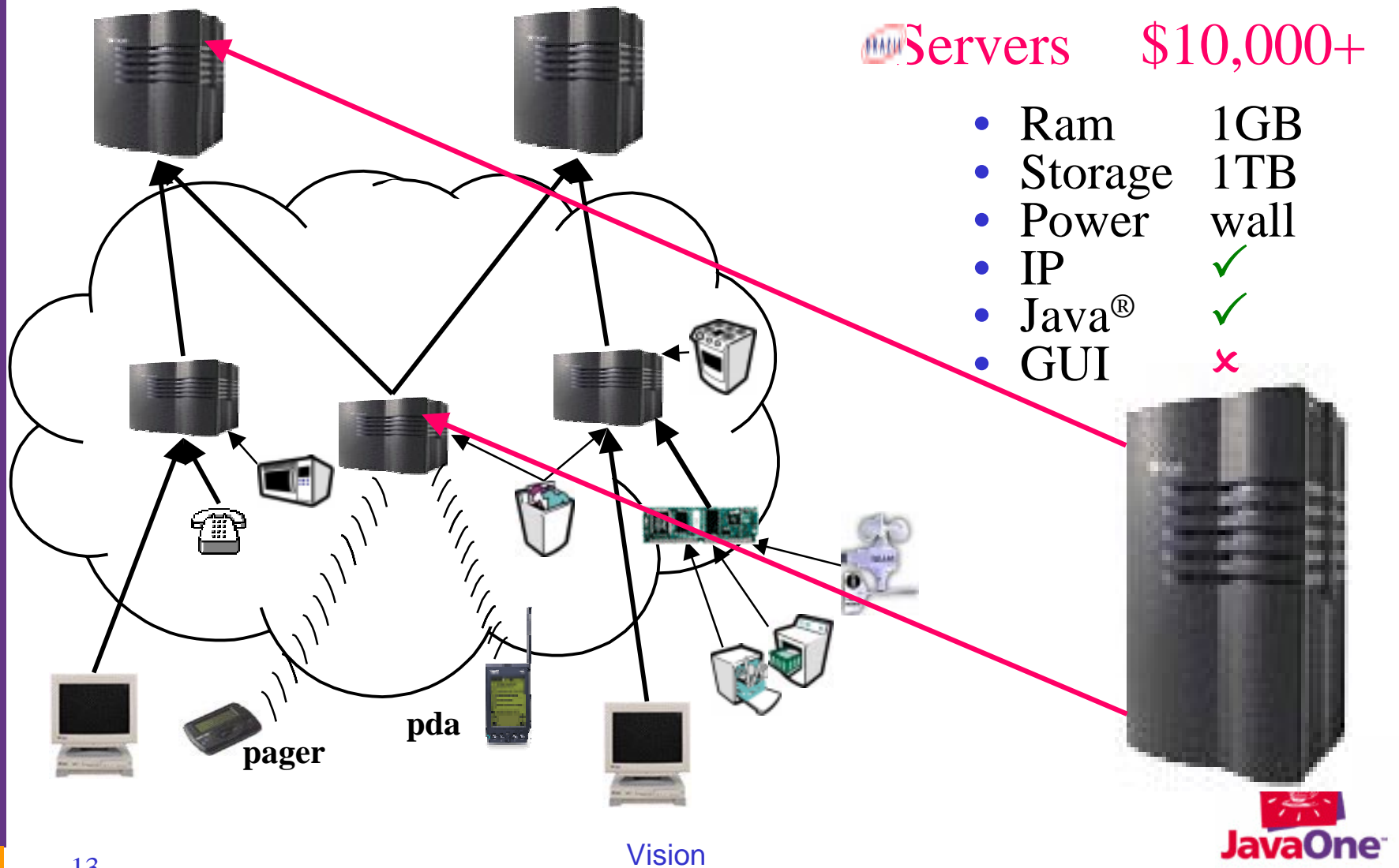
Some Examples



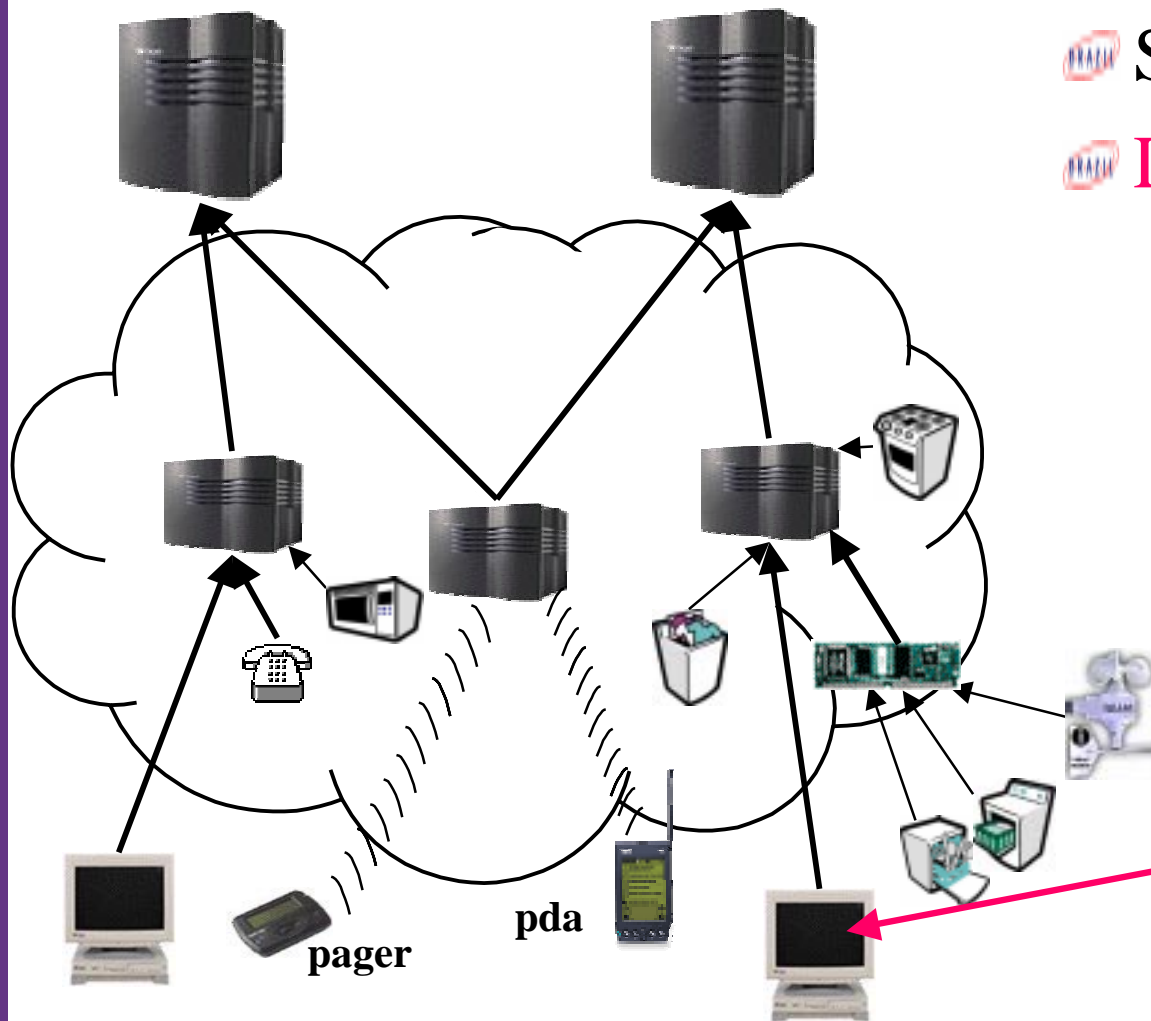
Request →
Response ←


- ❖ Workflow
- ❖ Localization
- ❖ Personalization
- ❖ **Device Integration**
 - Request filtered by portal
 - Content gathered from multiple sources
 - Integrated content delivered to client

Breadth of Computing Environments



Breadth of Computing Environments



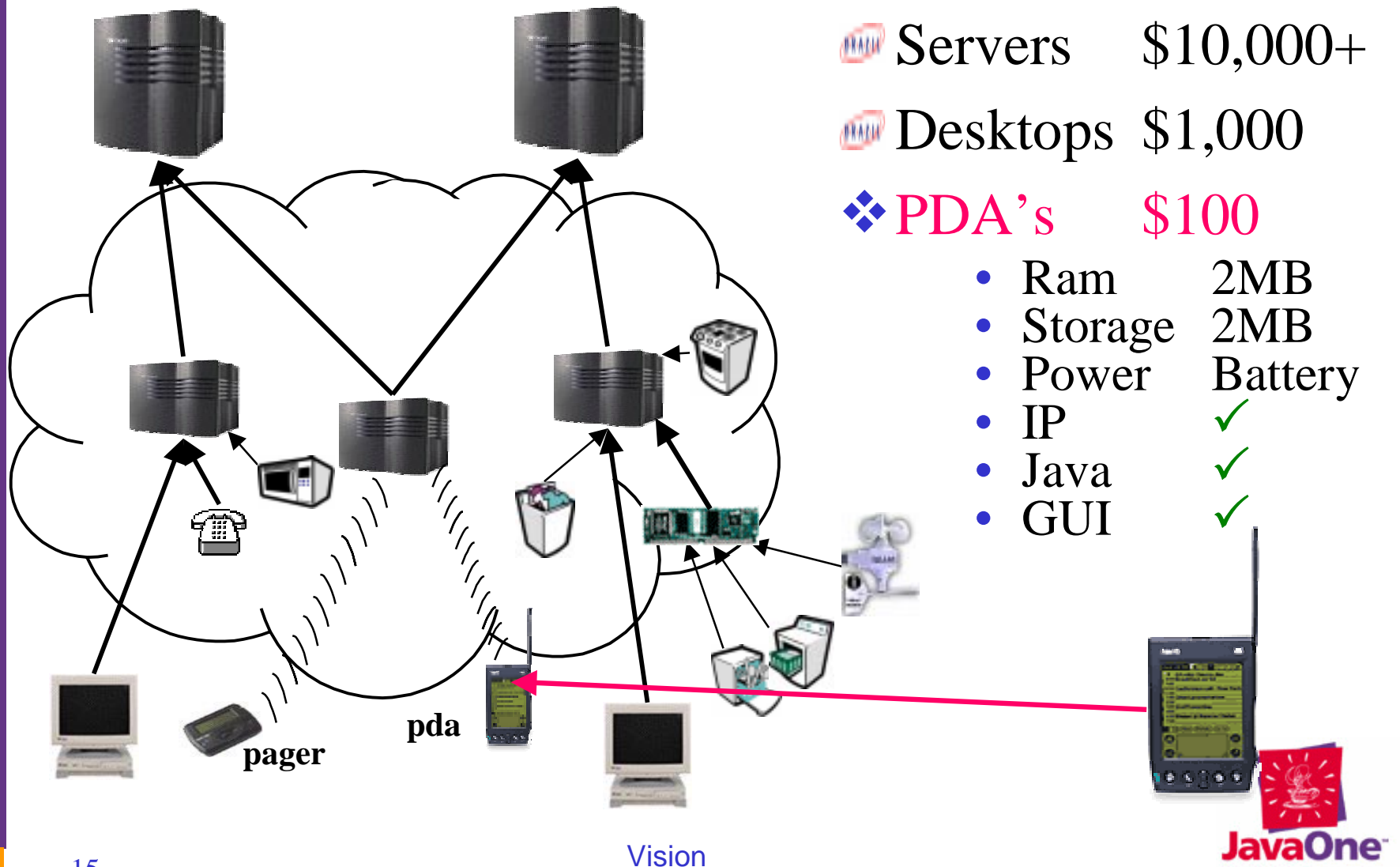
 Servers \$10,000+

 Desktops \$1,000

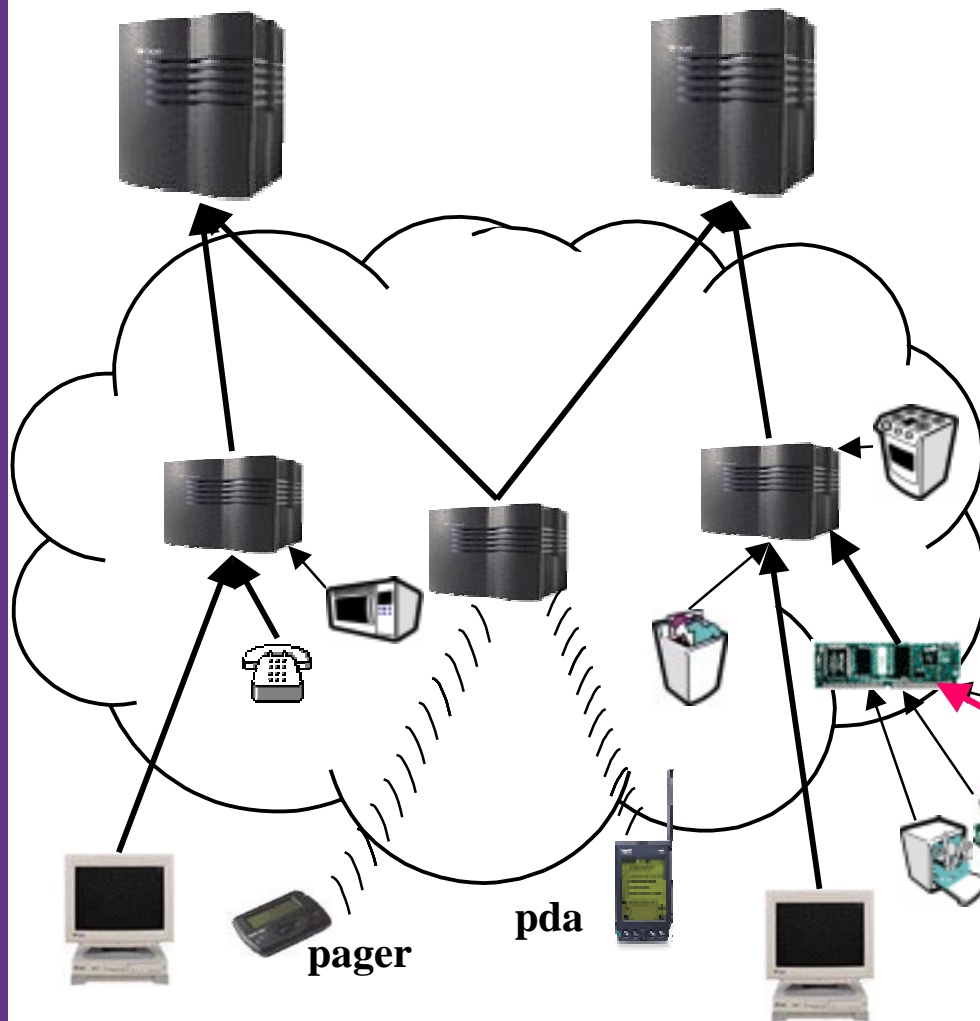
- Ram 64MB
- Storage 10GB
- Power wall
- IP ✓
- Java ✓
- GUI ✓



Breadth of Computing Environments



Breadth of Computing Environments



 Servers \$10,000+

 Desktops \$1,000

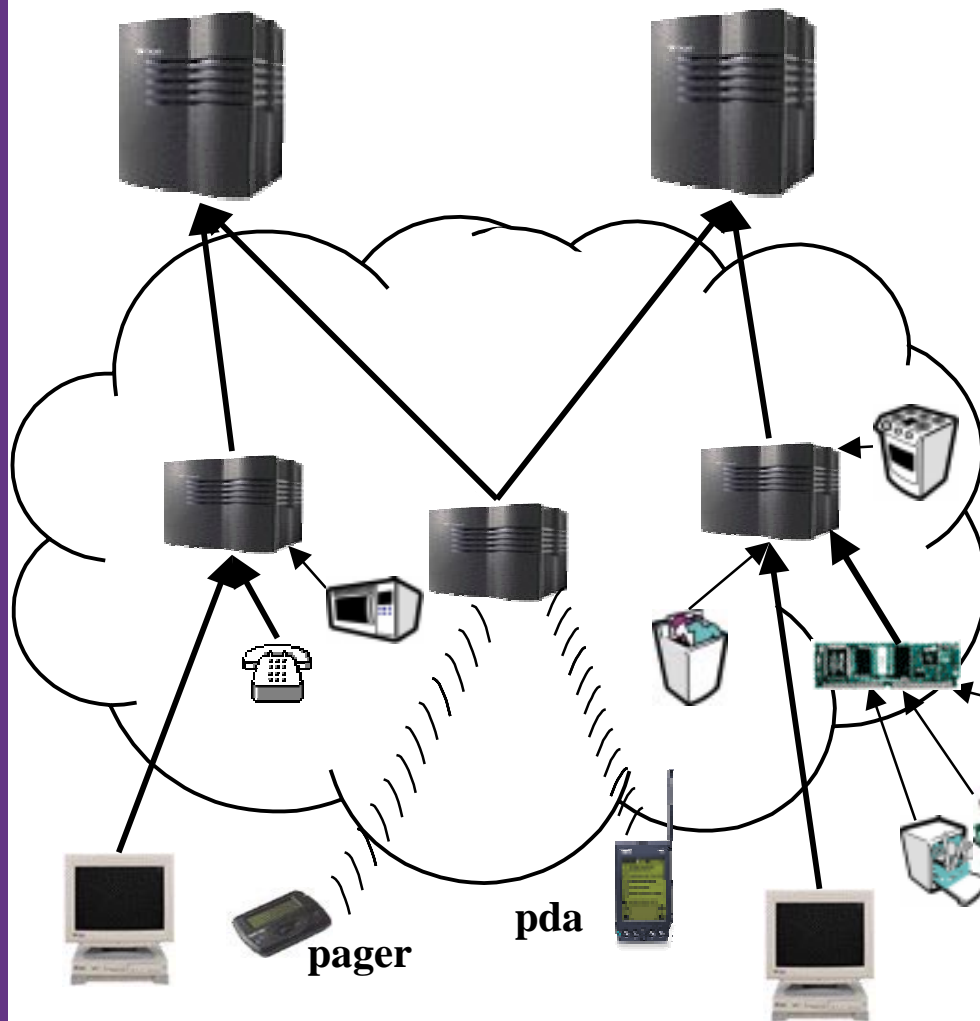
❖ PDA's \$100

 Embedded \$10

- Ram 0.5MB
- Storage 0.5MB
- Power wall
- IP ✓
- Java ✓
- GUI ✗



Breadth of Computing Environments



 Servers \$10,000+

 Desktops \$1,000

❖ PDA's \$100

 Embedded \$10

❖ **Micro** \$1

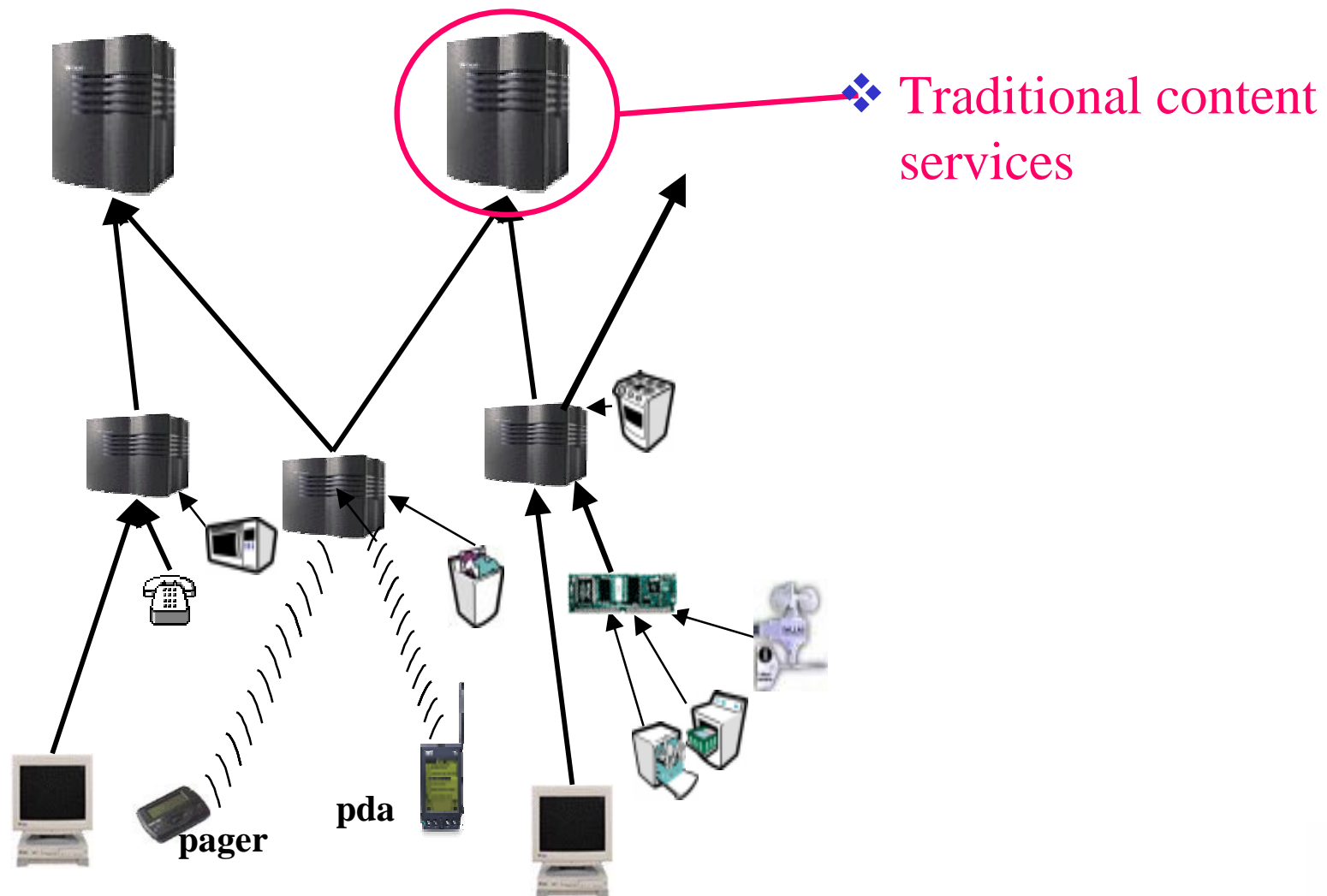
- Ram 0.05KB
- Storage 2KB
- Power battery
- IP ✗
- Java ✗
- GUI ✗



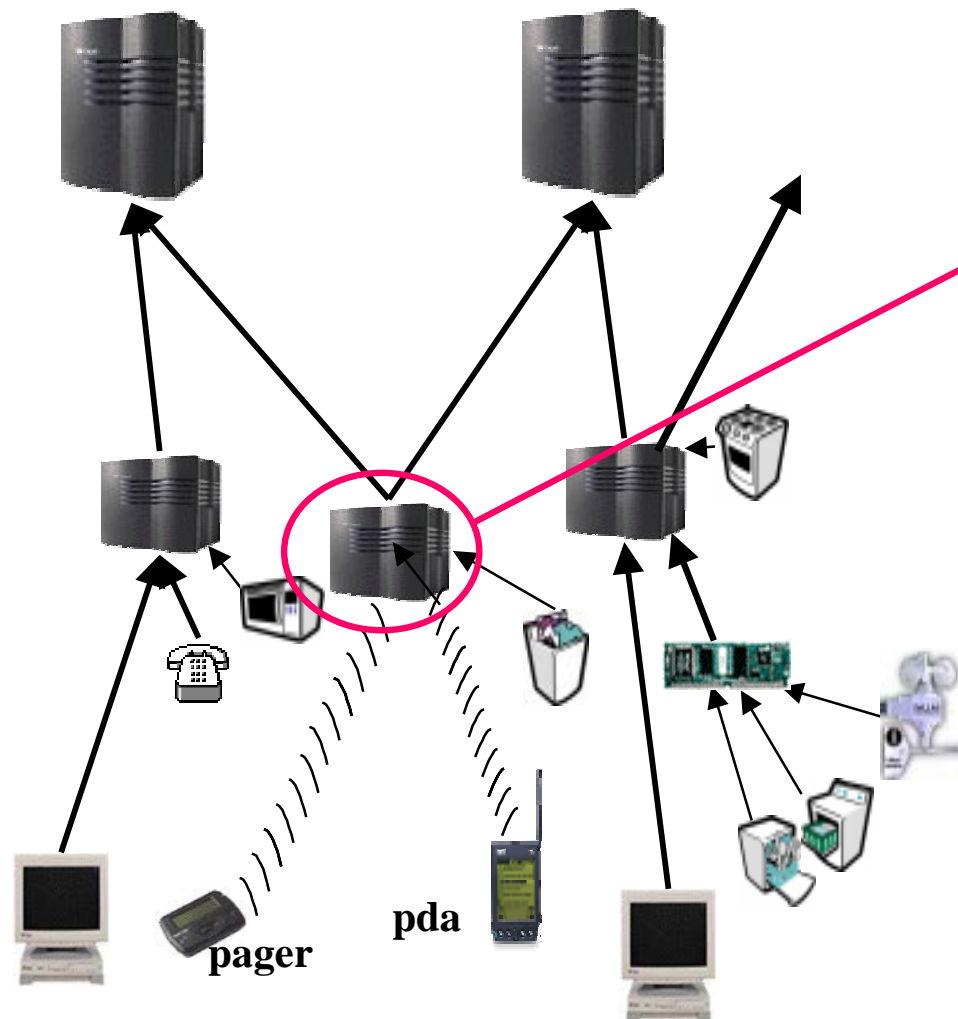


- ❖ A common infrastructure for a wide range of systems:
 - Small applications are simple to do
 - Large applications achieved by combining simple parts in consistent ways
- ❖ A methodology that encourages reuse at the appropriate granularity to manage complexity.
- ❖ Rich toolkit of powerful, composable parts.

Brazil



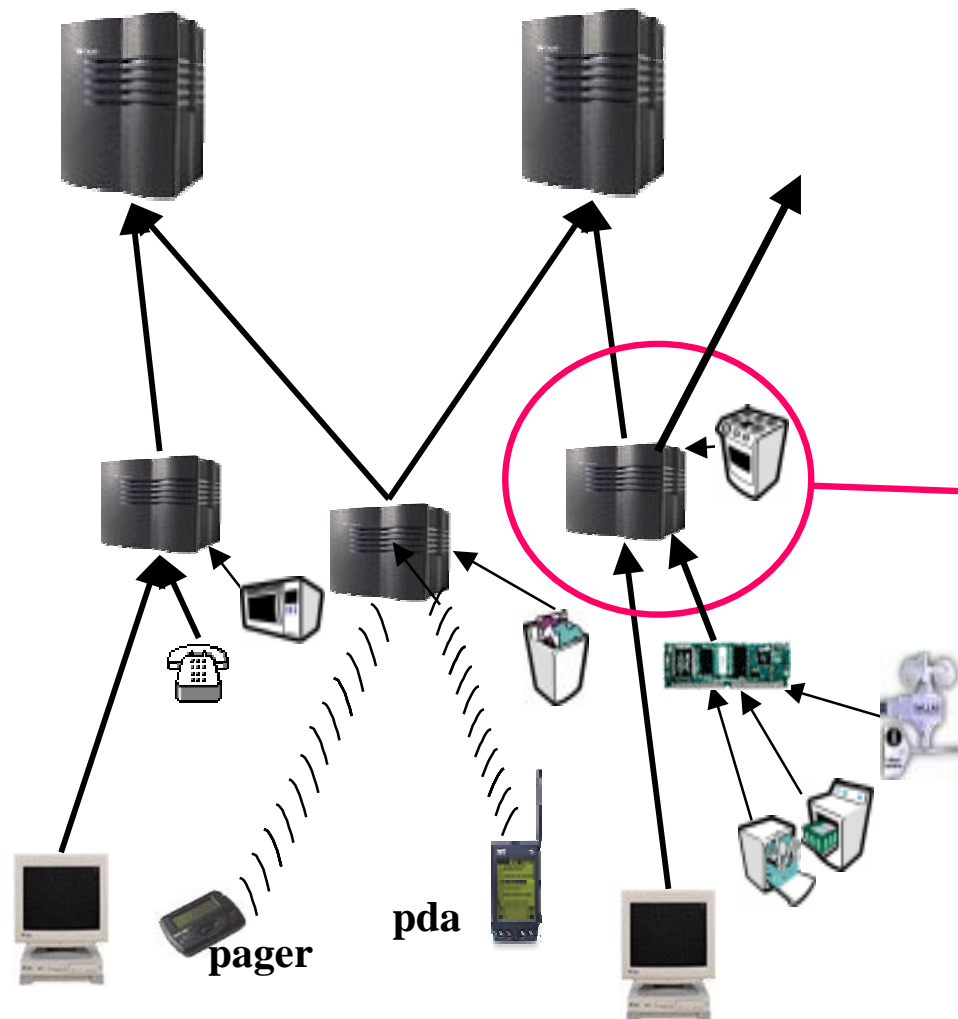
Brazil



❖ Traditional content services

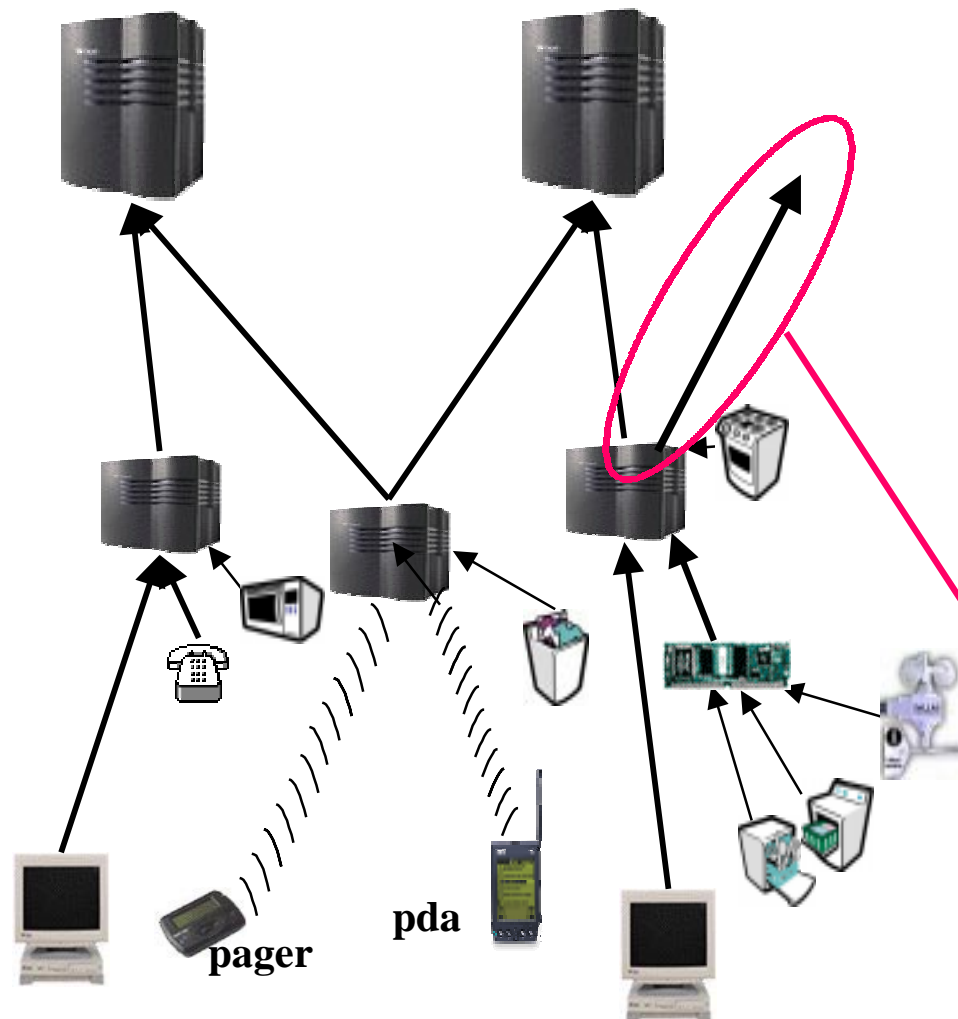
❖ Dynamic content transcoding for different GUI's

Brazil



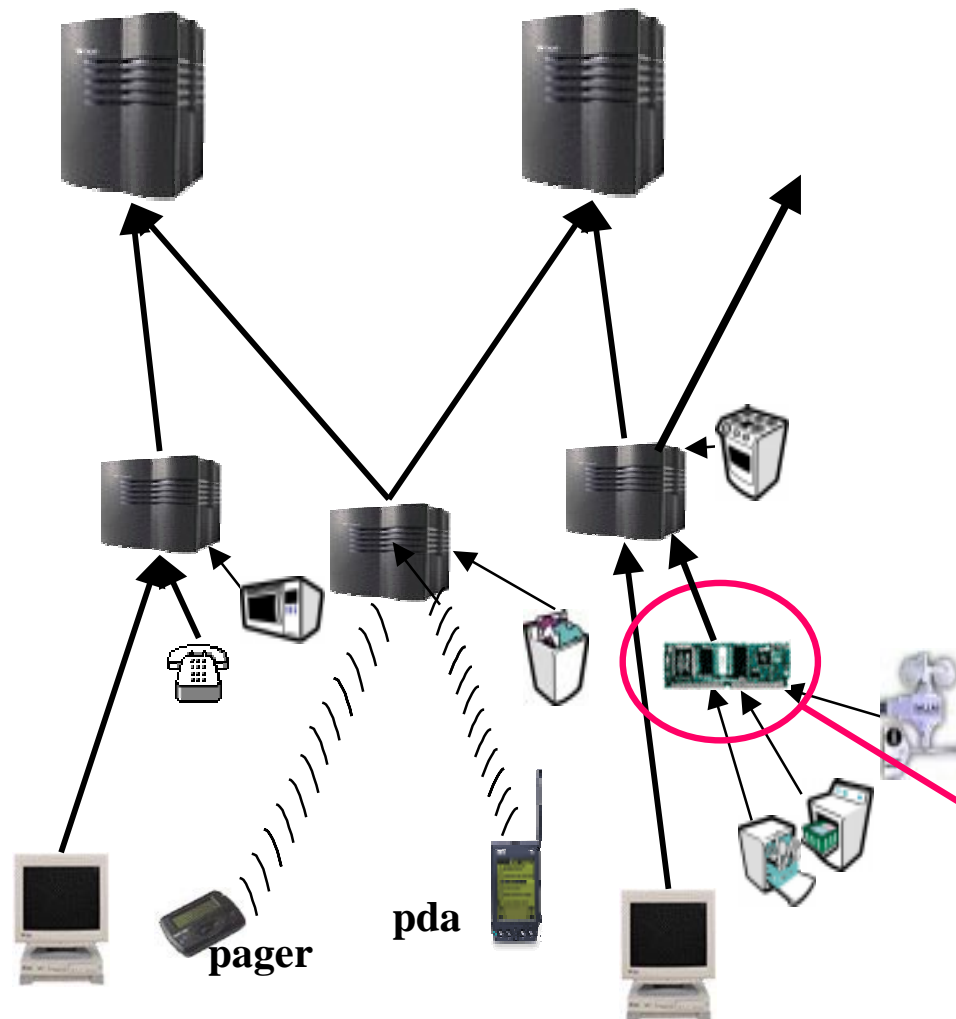
- ❖ Traditional content services
- ❖ Dynamic content transcoding for different GUI's
- ❖ Content aggregation for simple data sources

Brazil



- ❖ Traditional content services
- ❖ Dynamic content transcoding for different GUI's
- ❖ Content aggregation for simple data sources
- ❖ Extraction and integration of foreign content

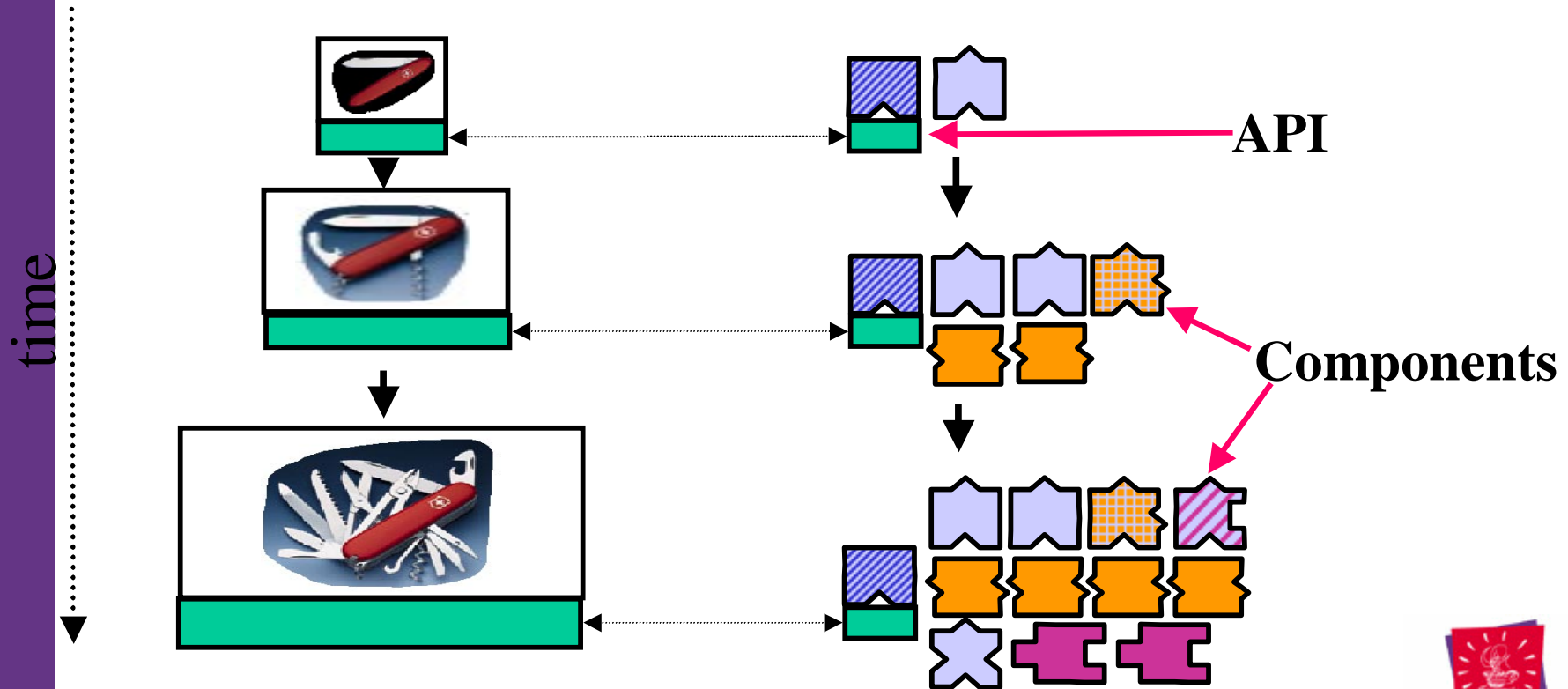
Brazil



- ❖ Traditional content services
- ❖ Dynamic content transcoding for different GUI's
- ❖ Content aggregation for simple data sources
- ❖ Extraction and integration of foreign content
- ❖ Communication with Sub-IP Devices via proxies

Architecture Comparison: How Complexity Increases

The "Other" Brand

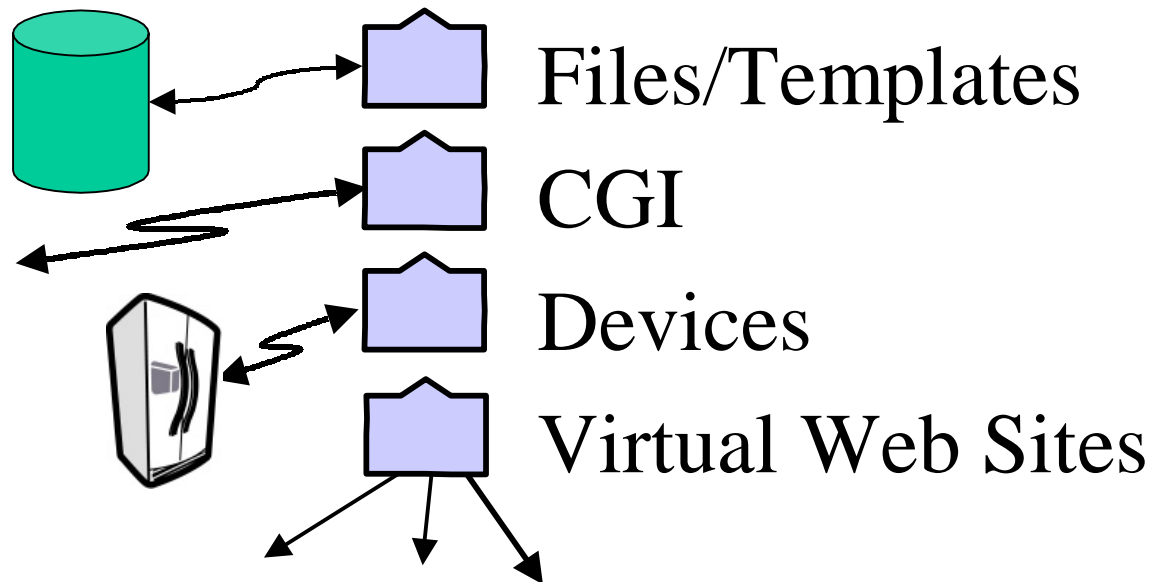


Architecture

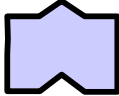
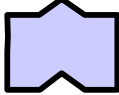
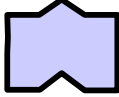
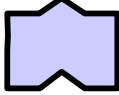
Brazil Functional Components

- ❖ Content retrieval
- ❖ Request infrastructure
- ❖ Content transformation
- ❖ Special Purpose
- ❖ Support

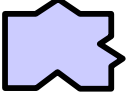
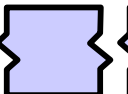

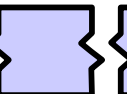
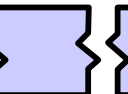

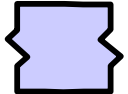
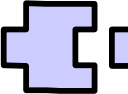
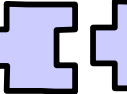
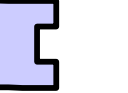


Content retrieval



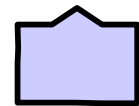
Request Infrastructure

-  Session management
-  Authentication
-  Access control
-  Request distribution

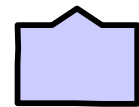
Content Transformation

-  Content filtering     
-  XML template processing   
-  Data extraction and manipulation
TCL, Python, SQL, LDAP, IMAP...
-  Content formatting: BSL
(Brazil Scripting Language)

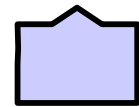
Special Purpose



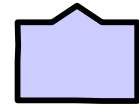
X10



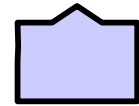
Jini®



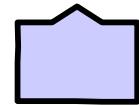
SmartCards



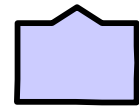
Certificate Authorities



ActiveX/Com connectivity



JRMS



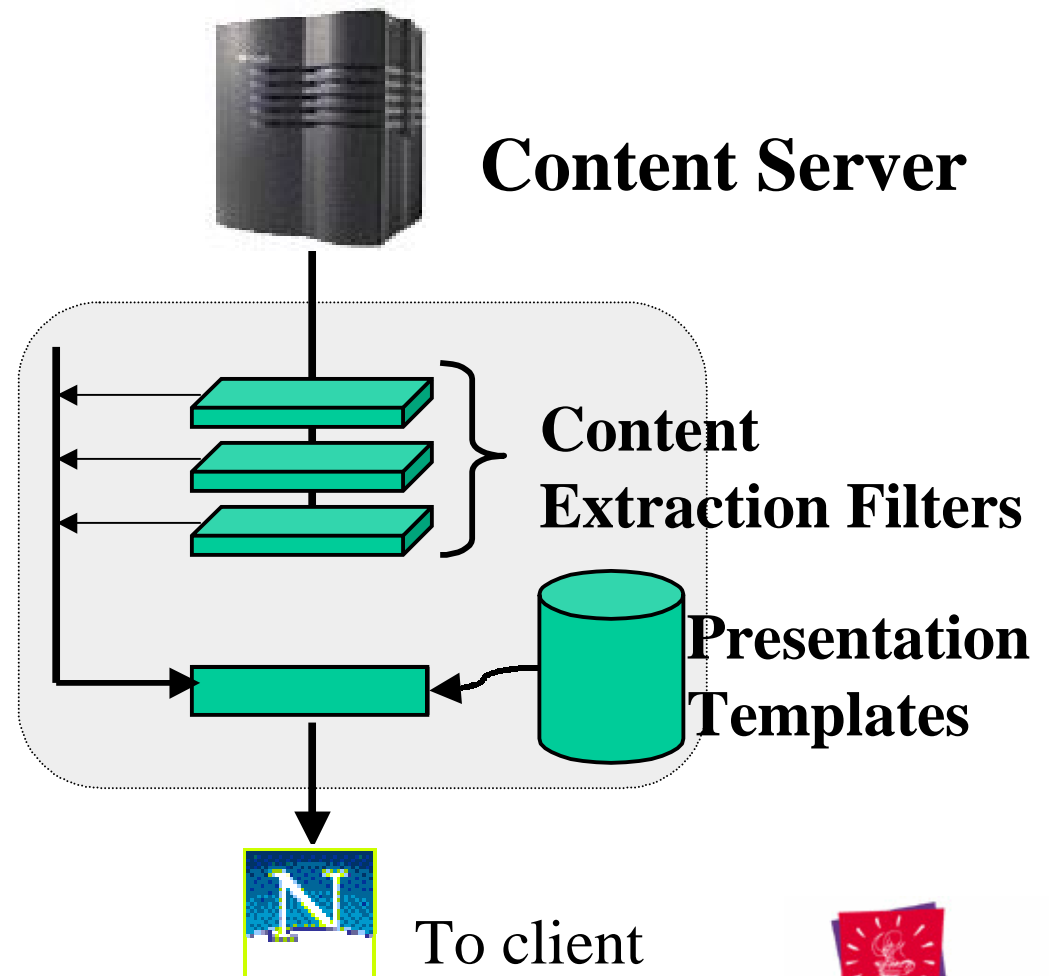
Gnutella

Support

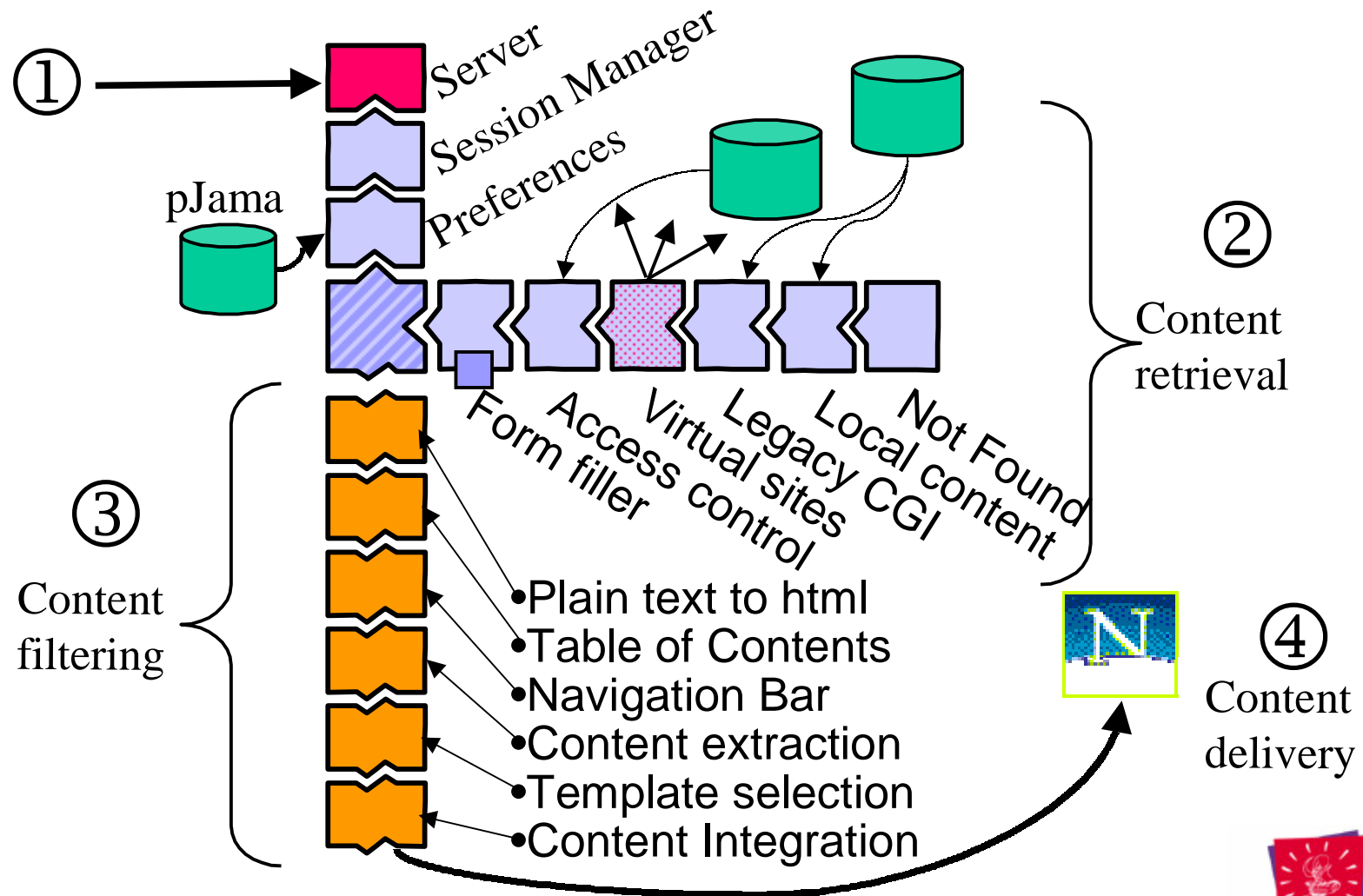
- XML processing
 - URL rewriting
 - Templates
- HTTP/HTML utilities
- Regular expressions
- Proxy services

Sample Portal Configuration (*conceptual view*)

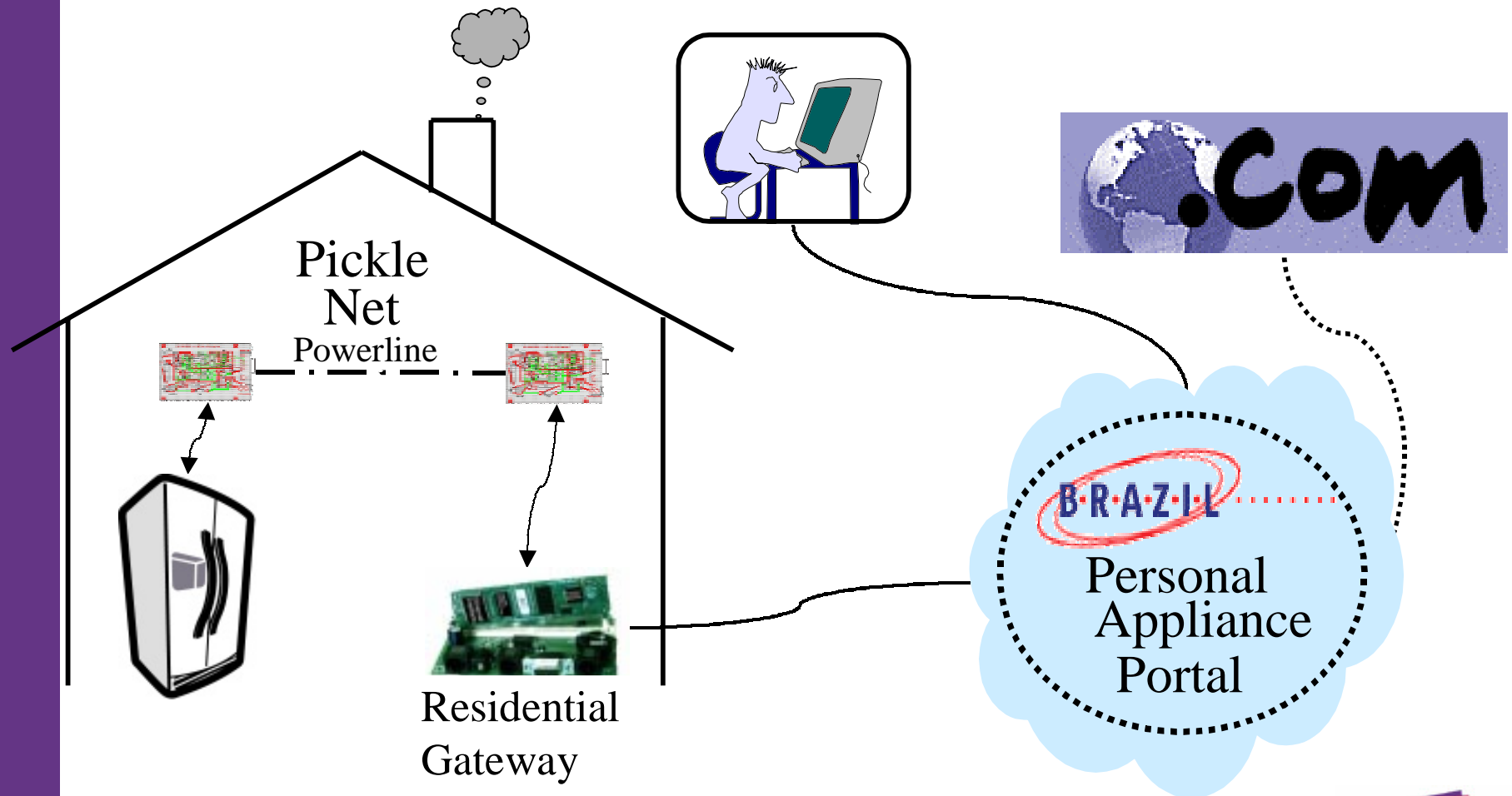
- ❖ Pages are retrieved from remote sites.
- ❖ Content is extracted
- ❖ A new page is created, combining extracted content with desired *look*.



Portal Configuration (concrete view)



FridgeNet (conceptual view)



FridgeNet in Operation

GE *We bring good things to life.*

HOME START SHOPPING WHAT'S HOT WHERE TO BUY
GE ANSWER CENTER® | ASK OUR TEAM OF EXPERTS

QUICK SEARCH **go**

Your Refrigerator
TFX30PB8WW
GE Profile Performance
29.8 Cu. Ft. Side-By-Side
Dispenser
Refreshment Center

You need a new water filter in **21 days.**

Your room temperature is: **70° F**

Your Frozen Food Compartment temperature is: **7° F**
[Change this setting](#)

Your Ice Bin is **100% FULL.**

Your Fresh Food Compartment temperature is: **39° F**
[Change this setting](#)

The state of your freezer door (open=on, closed=off) is **off.**

The state of your fresh food door (open=on, closed=off) is **off.**

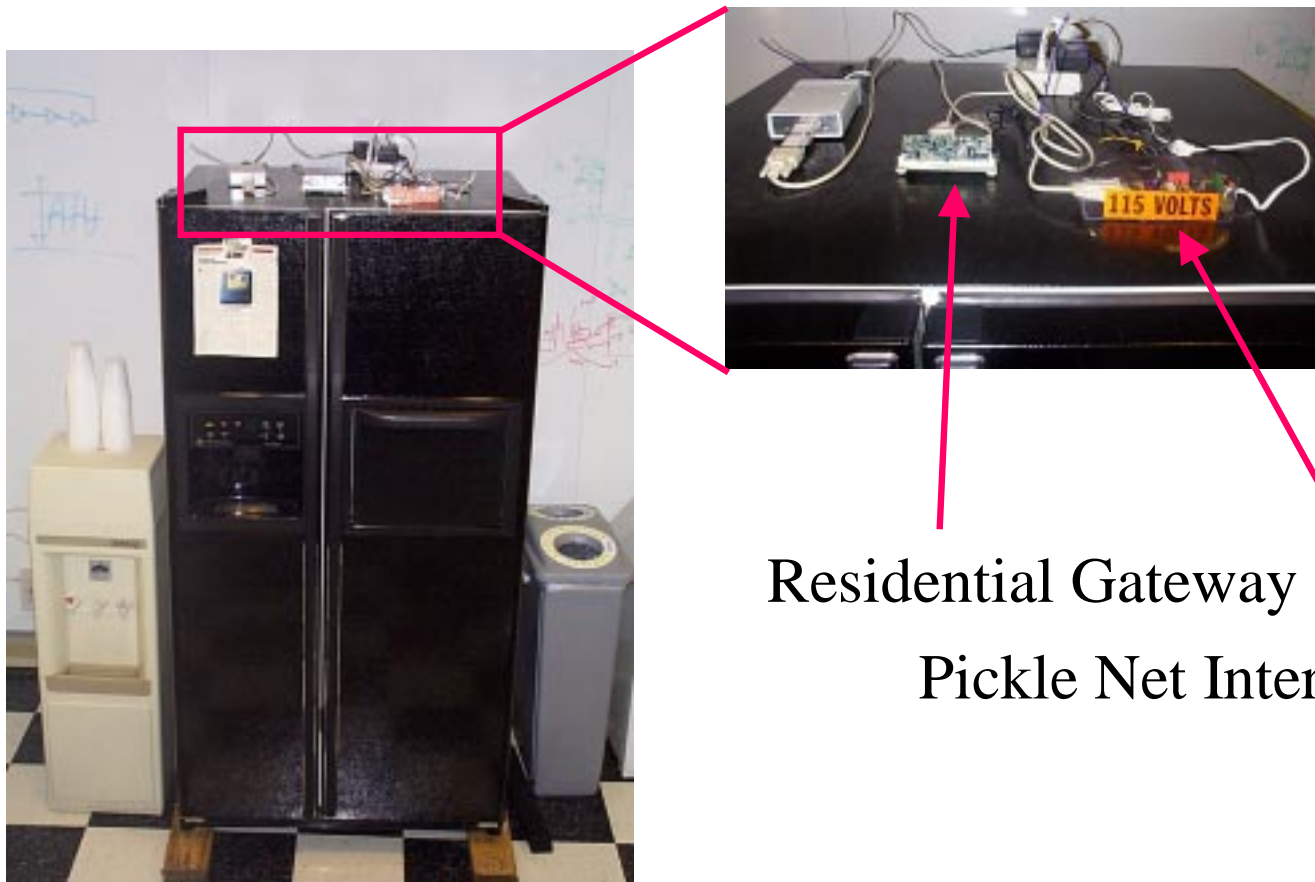
BRAND COLLECTIONS ▼ EASY FINDER **go**

© 2000 General Electric Company [Recalls](#) | [Contact Us](#) | [Y2K](#) | [Sitemap](#) | [Privacy Policy](#) | [Terms & Conditions](#)

Fridge.eng@Moscone



Fridge.eng @ home



Residential Gateway

Pickle Net Interface

How to get Started

- ❖ Download your own copy
 - <http://www.experimentalstuff.com/>
- ❖ View the built-in documentation
 - run: **java -jar brazil.jar**
 - Point your browser to: <http://localhost:8080/>
- ❖ Add **brazil.jar** to your CLASSPATH
- ❖ Unpack the archive
 - Sample applications
 - Source code

How to get Started

- ❖ New combinations of existing pieces
- ❖ Write your own application specific code

Provided by:

- Handlers* *Server*
- Filters *FilterHandler*
- Templates*. *TemplateFilter*
- Scripting *Tcl -or- Python Template*

Existing Application

```
Class MyApp {  
    ...  
    public static void main(String[] args) {  
        ...  
    }  
}
```

Hello World

(handler version)

```
Class MyApp implements Handler {
    public boolean
    init(Server server, String name) {
        return true;
    }
    public boolean
    respond(Request request) {
        request.sendResponse("Hello World!");
        return true;
    }...
    public static void main(String[] args) {
        new Server(new ServerSocket(8080),
            "MyApp", new Properties()).start();
    }
}
```


Hello world (*template version*)

❖ Source code

```
Class MyApp extends Template {  
    public void  
    tag_myapp(RewriteContext rc) {  
        rc.append("Hello World!");  
    }  
}
```

❖ Sample template

```
<title>World Existence Page</title>  
<body>  
    <h1><myapp></h1>  
</body>
```

Using Templates (*source code*)

❖ Application specific functionality

- Invent application specific “query” language
- Return “results” as name/value pairs
- No presentation markup

```
Class MyApp extends Template {  
    public void  
    tag_myapp(RewriteContext rc) {  
        String p=rc.get("prefix");  
        rc.request.props.put(p + ".h", "Hello");  
        rc.request.props.put(p + ".w", "World");  
    }  
}
```

Using Templates (*template*)

❖ Presentation

- Call application specific “query” language
- Use templates to generate presentation

```
<title>World Existence Page</title>  
<myapp prefix=world>  
<ul>  
  <foreach name=i glob=world.*>  
    <li><property name=i>  
  </foreach>  
</ul>
```

Using Templates (*configuration*)

❖ Integration

- Use server “properties” to tie functionality together

```
templates= MyApp \
    sunlabs.brazil.template.PropsTemplate \
    sunlabs.brazil.template.BSLTemplate

public static void main(String[] args) {
    Properties p = ;
    new Server(new ServerSocket(8080),
        "TemplateHandler", p).start();
}}
```



Summary

❖ Advantages

- **Composability:** Parts **do** fit together
- **Reusability:** Special purpose handlers reused
- *Slicers* and *Formatters* allow separation of content from presentation
- **Less structure:** Easy to get started

❖ Disadvantages

- **Performance:** Everything funnels through small interface
- **Security:** Uniform access to all information
- **Less structure:** Less framework for large systems

Some Numbers

❖ Size

- 25k lines of code¹ in core system
- 75k lines of code¹ in sample applications
- 30k – 250k bytes Jar file size

❖ Performance²

- 700 hits/sec (500 if every page is rewritten)
- 14Mbytes/sec throughput

❖ Distribution

- 10,000+ copies downloaded

¹Java Programming Language (JDK1.1)

²Sun 420R (*entry level server*)

References

- ❖ <http://www.experimentalstuff.com/>
- ❖ <http://www.sun.com/research/brazil>
- ❖ <http://fridge.eng/status> (*internal to Sun*)
- ❖ <http://www.javasoft.com/features/2000/08/brazil.html>
- ❖ <http://www.sun.com/research/features/myfridge/>
- ❖ [http://www.javaworld.com/
javaworld/jw-08-2000/jw-0811-javadev.html](http://www.javaworld.com/javaworld/jw-08-2000/jw-0811-javadev.html)

† Netscape Communications Corporation Logo is a registered trademark in the United States and other countries

‡ UltraSPARC is a registered trademark of SPARC International, Inc.

The End

Brazil Team:
Rinaldo DiGiorgio
Colin Stevens
Stephen Uhler

