Survey of Technologies for Web Application Development

Web-based application developers face a dizzying array of platforms, languages, frameworks and technical artifacts to choose from. We survey, classify, and compare technologies supporting Web application development. The classification is based on (1) foundational technologies; (2) integration with other information sources; and (3) dynamic content generation. We further survey and classify software engineering techniques and tools that have been adopted from traditional programming into Web programming. We conclude that, although the infrastructure problems of the Web have largely been solved, the cacophony of technologies for Web-based applications reflects the lack of a solid model tailored for this domain.

Introduction

- The Web, initially static, was developed in 1990 at CERN by Berners-Lee.
- Implementation became public domain in 1993, Mosaic led rush to the Web.
- Static content didn’t support the applications that were needed (online commerce, education, communication, etc.).

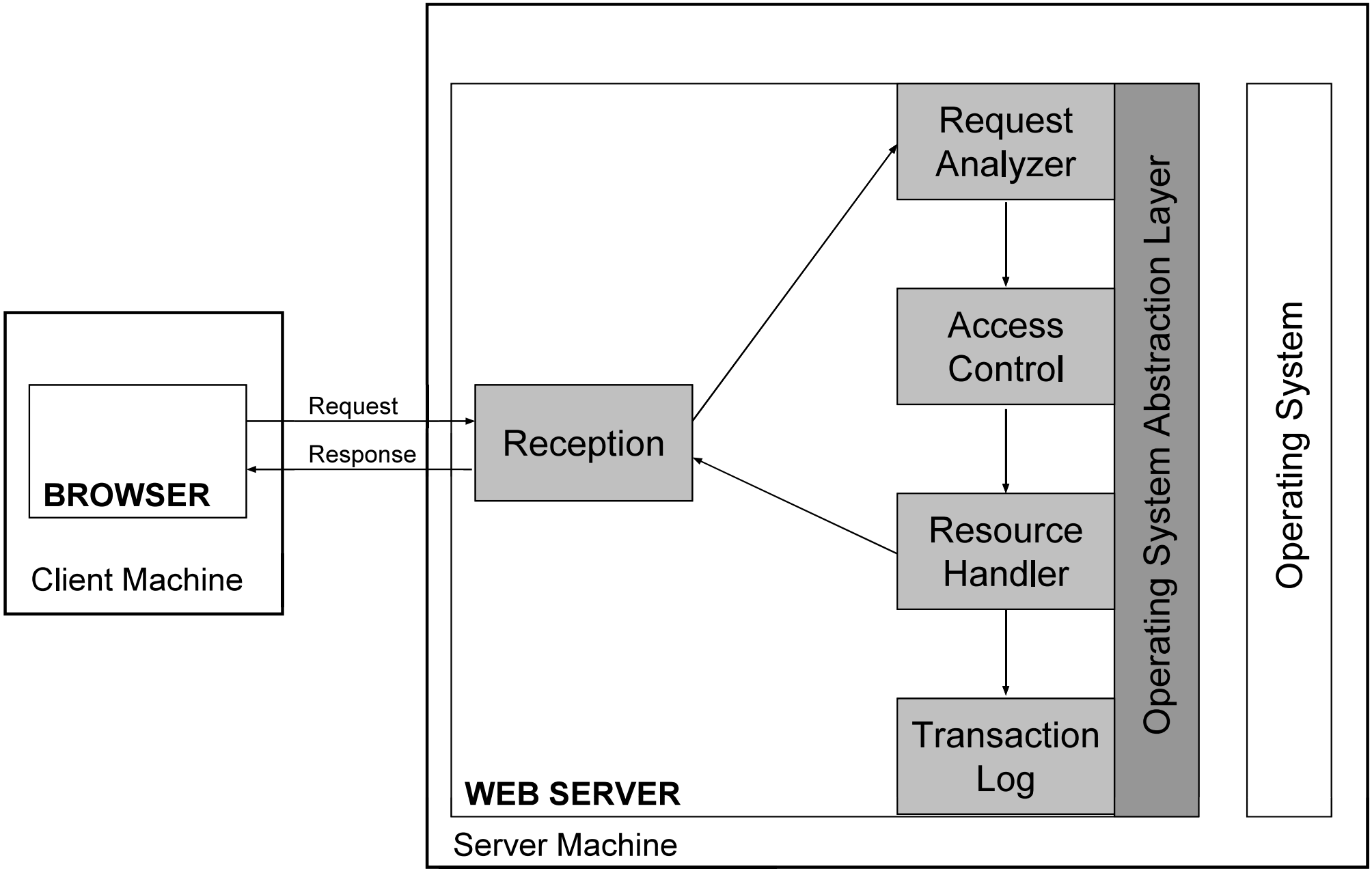
Factor	Objectives
Global distribution	Reliability Extensibility Scalability Performance Portability over heterogeneity Loose coupling Security
Interactivity	Dynamic page generation Data validation Handling of browser navigation anomalies State management Event-action
Human and socio-economic factors	Agile development Designer/programmer integration Learning effort Popularity Conformance to standards and practices

Table I. Web technology design factors.

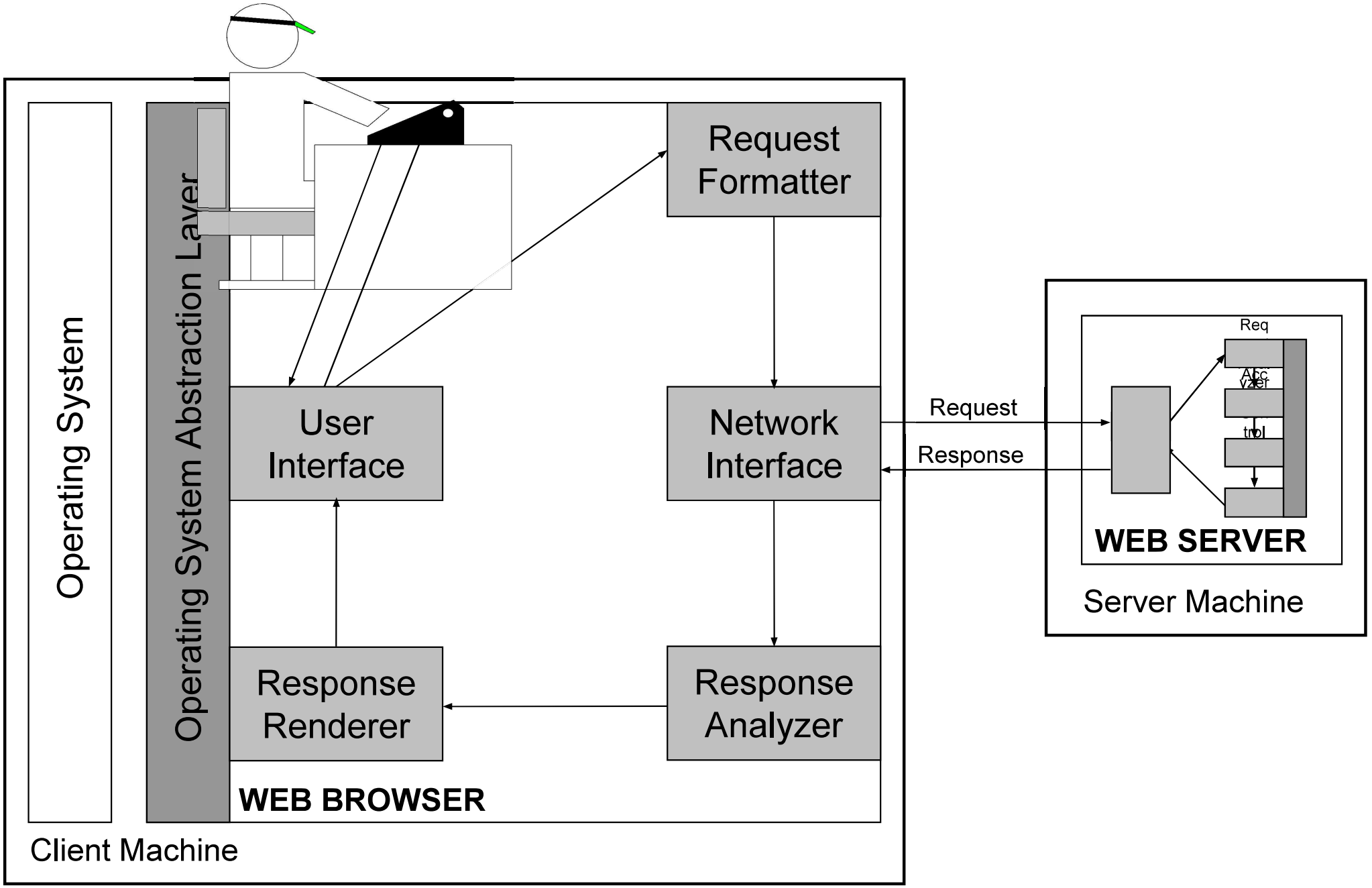
Support for Dynamic Content

- The HTTP request-response cycle provides an architectural foundation for distributed hypertext applications.
- Web servers and browsers communicate through message-passing, browser initiated requests by URI name for resources (HTML pages, etc.)
- Initial dynamic interaction supported by HTML forms and CGI.

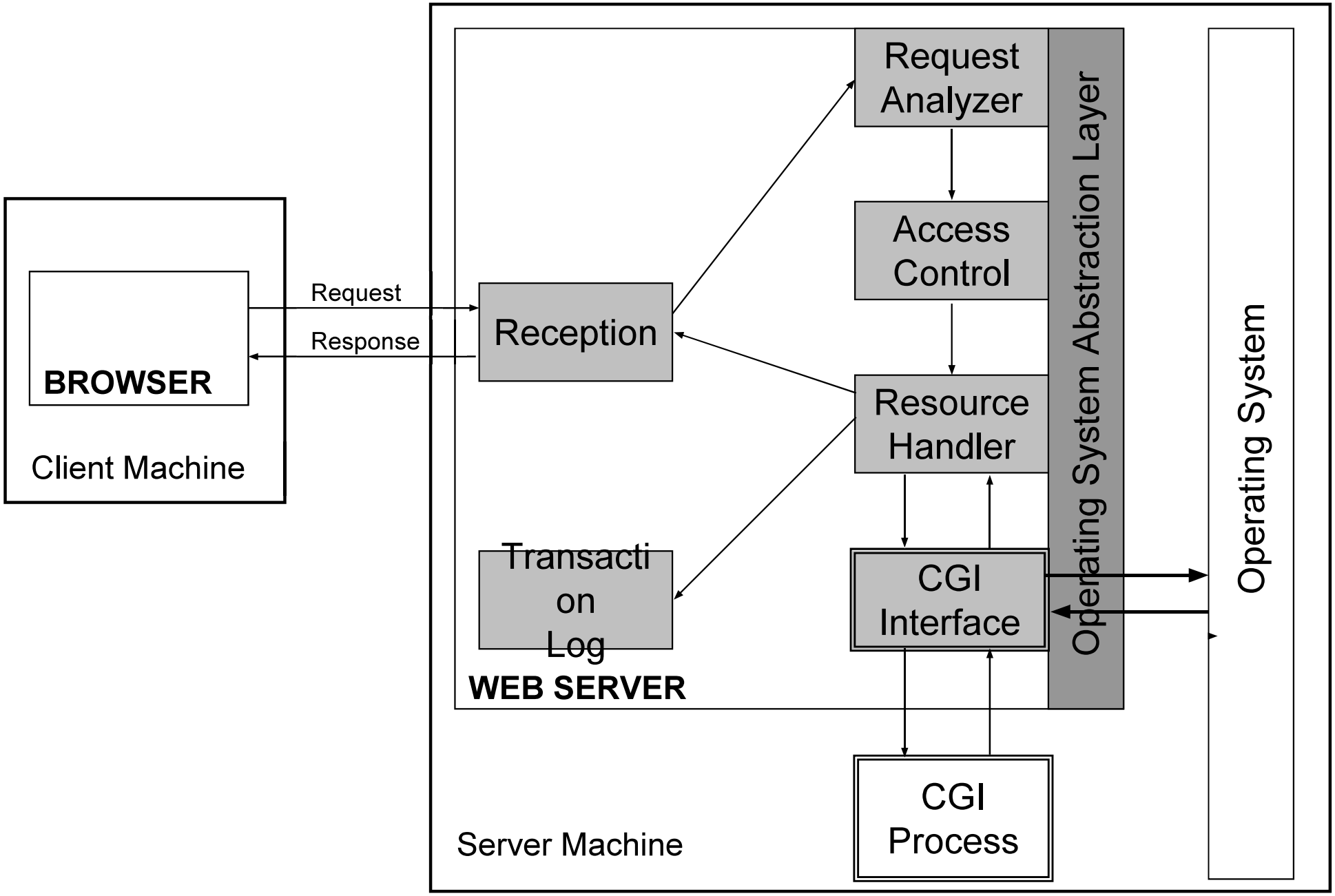
A Web server reference architecture (Hassan)



A reference architecture for browsers



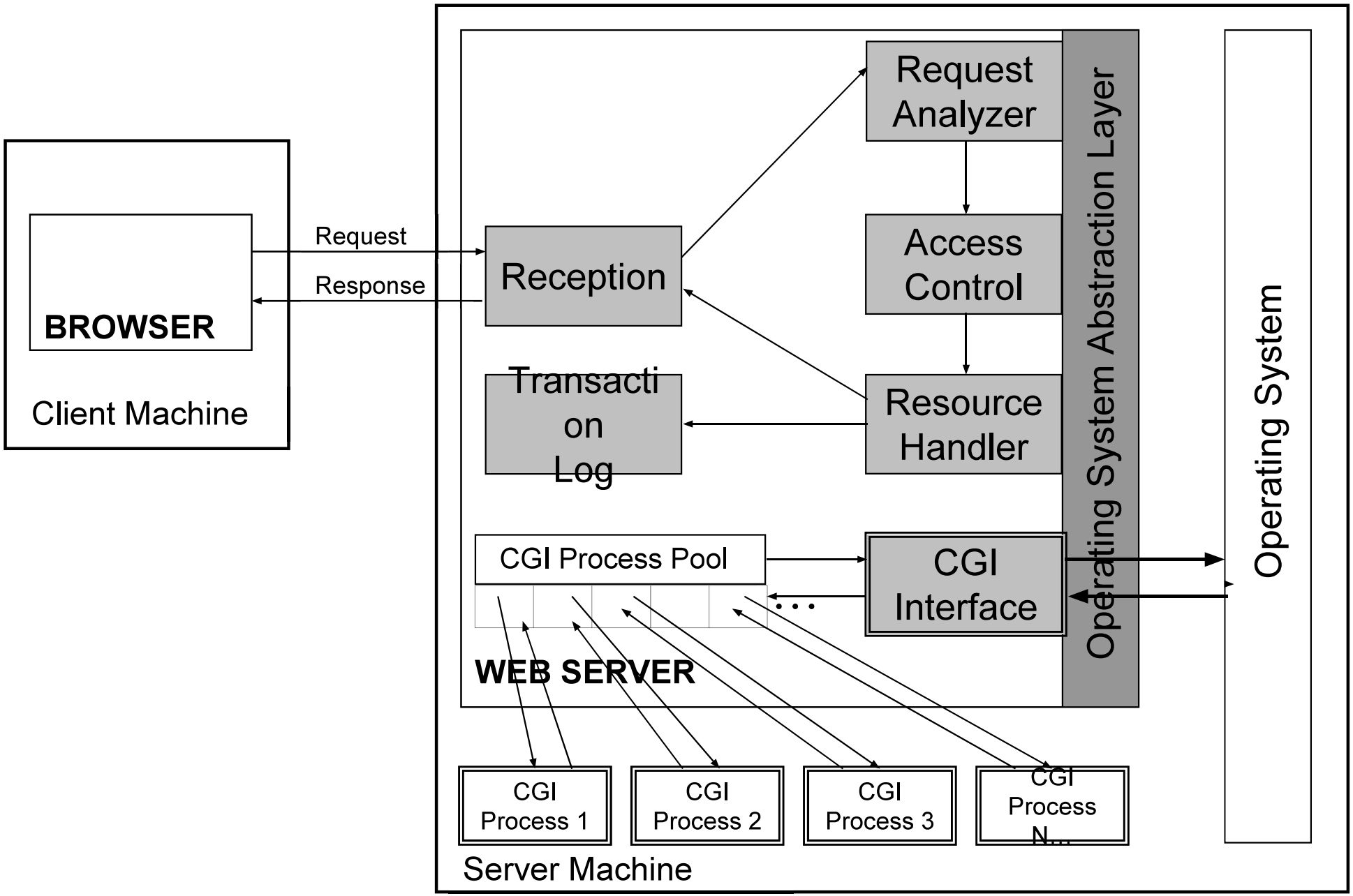
CGI


Request Analyzer
Access Control
Resource Handler
CGI Interface
Operating System Abstraction Layer
Operating System
Request
Response
Reception
BROWSER
Client Machine
Transacti on Log
WEB SERVER
CGI Process
Server Machine

CGI advantages / disadvantages

- Simple, implemented on all well-known Web servers out-of-the-box.
- Combined with scripting languages are a portable solution.
- Not process efficient.
- HTML generation from within code, not providing separation between the HTML designers and programmers.

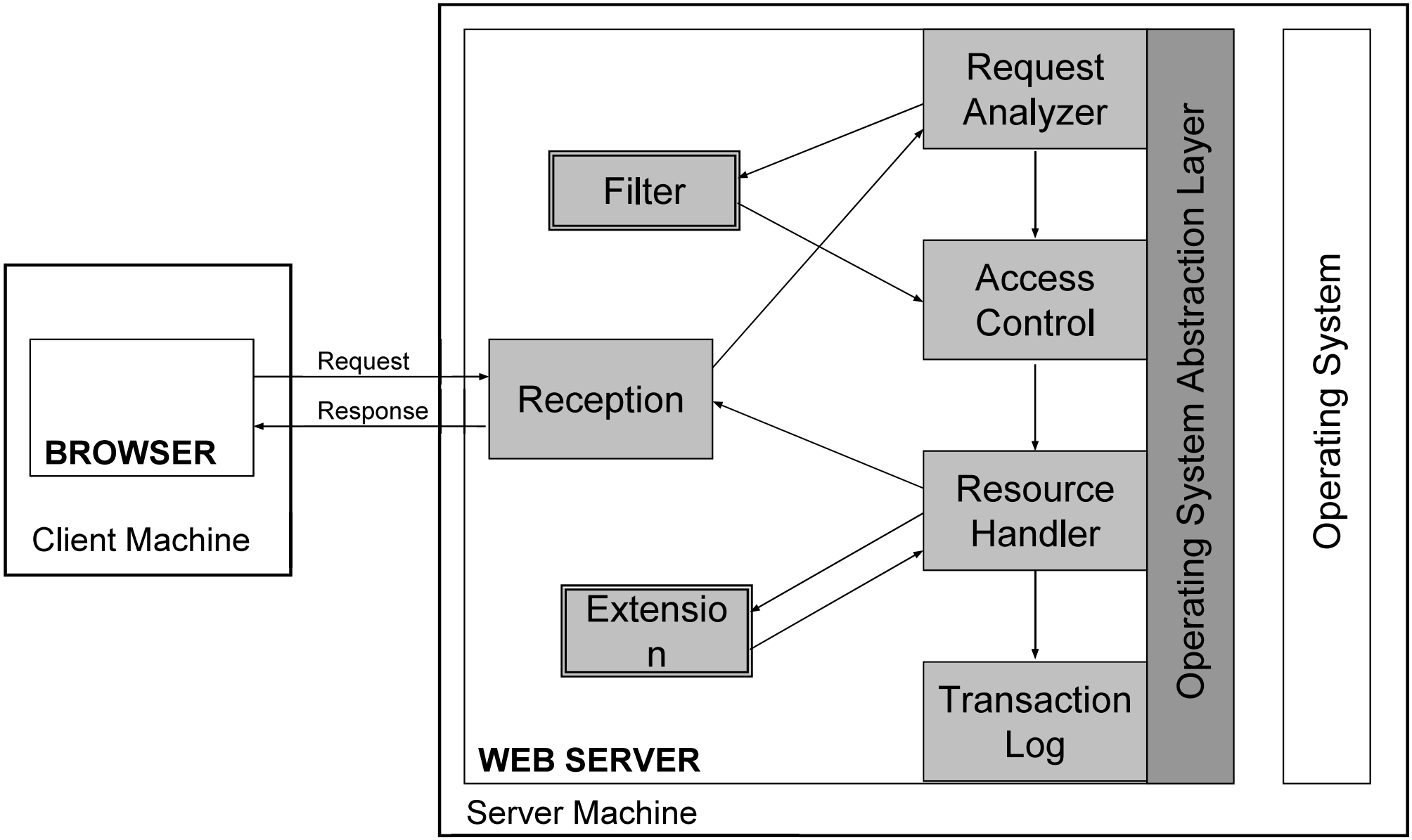
Scalable CGI



Scalable CGI advantages / disadvantages

- FastCGI is the most well-known implementation.
- Performance is very good, still better than more recent technologies.
- The usability disadvantages of CGI still apply, programmers are responsible for everything and must know details of HTTP.

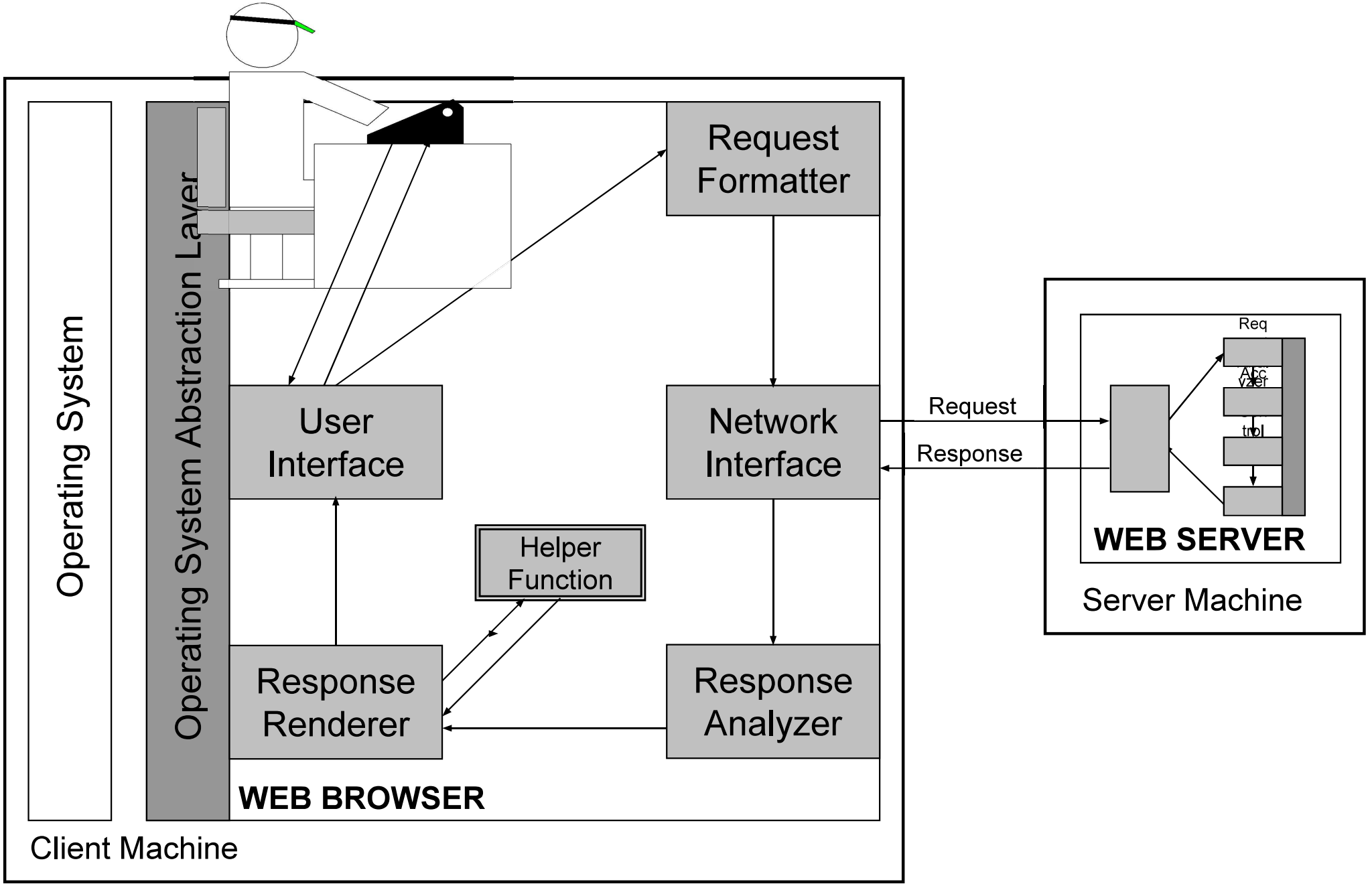
Web server APIs



Web Server APIs

- NSAPI, ISAPI, Apache API.
- Very efficient since compiled extension modules run within the Web server's address space…
- …but also dangerous since a bug in an extension module can crash the Web server.
- Not commonly used for applications, but for performance reasons, most server-side technologies that support dynamic content are based on Web server extension modules.

Browser extension interfaces



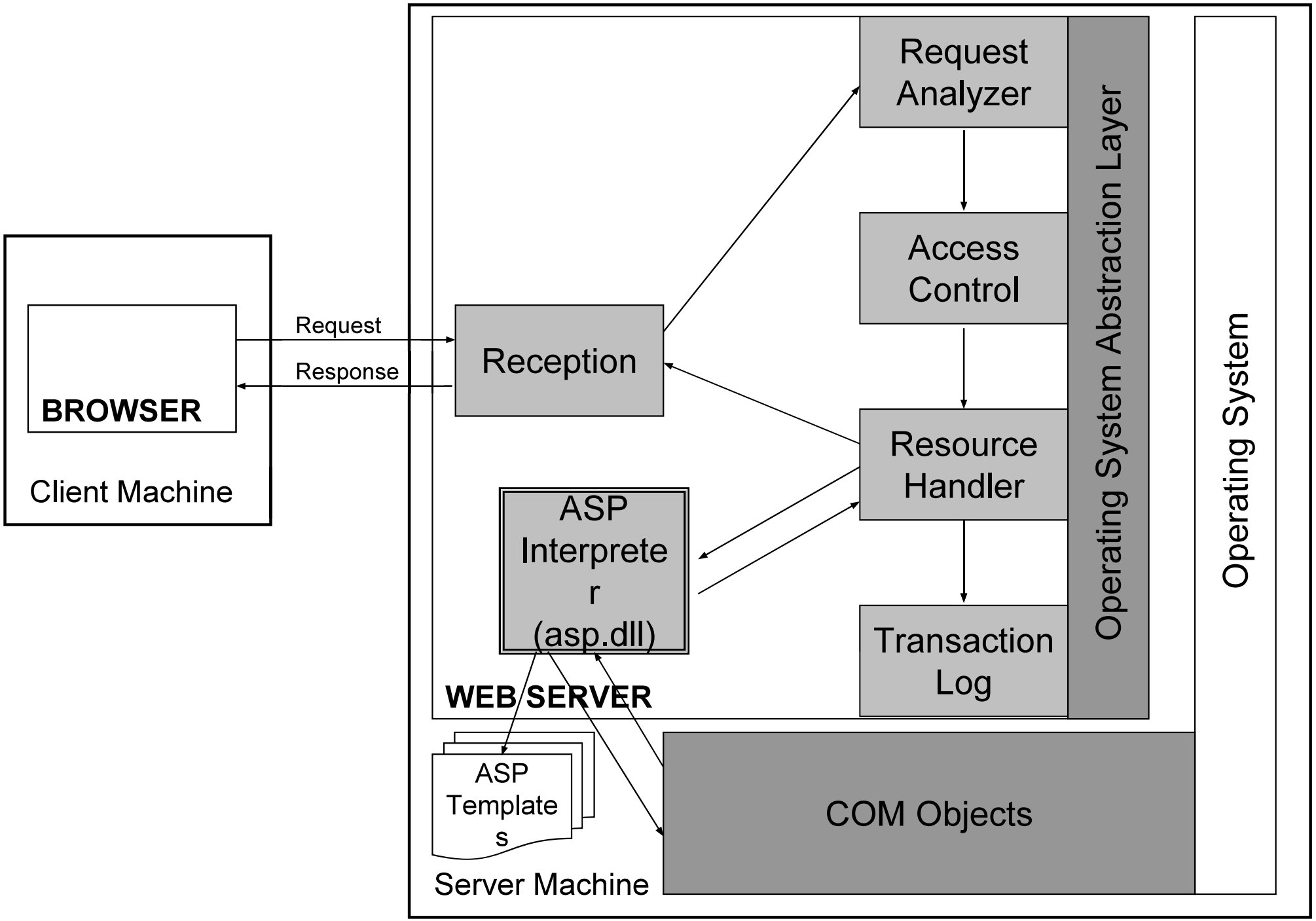
Browser Extension Interfaces

- CCI (obsolete)
- Plug-ins
- Client-side scripting (JavaScript, DOM)
- Java applets
- ActiveX
- Flash
- Rich internet applications (RIA)
  - Attempts to break free from page-centered interactivity constraints.

Interpreted Template-based Scripting

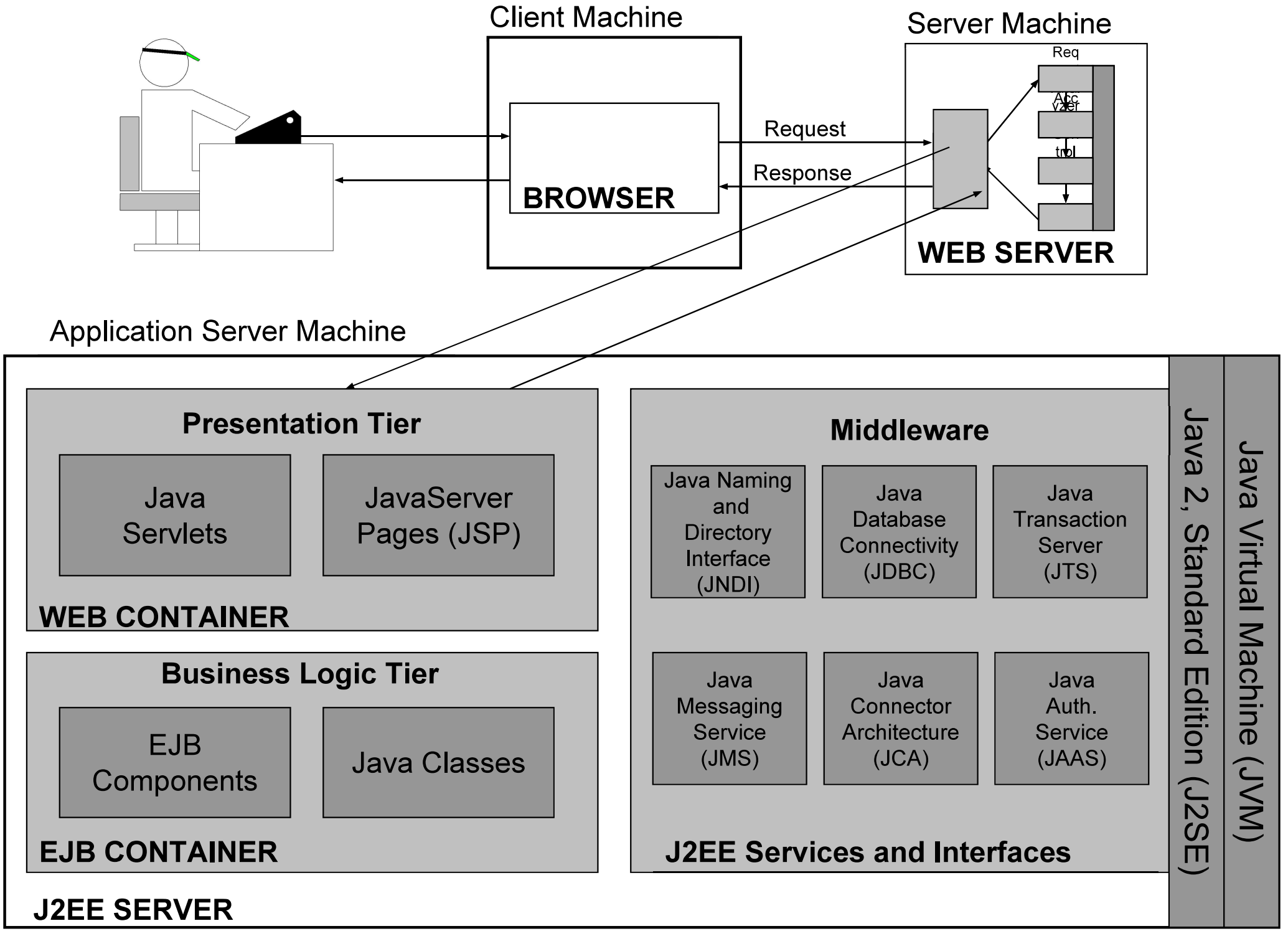
- Server-Side Includes (SSI)
- Extended SSI (XSSI)
- ColdFusion
- Server-side Java Script (SSJS)
- Active Server Pages (ASP)
- PHP

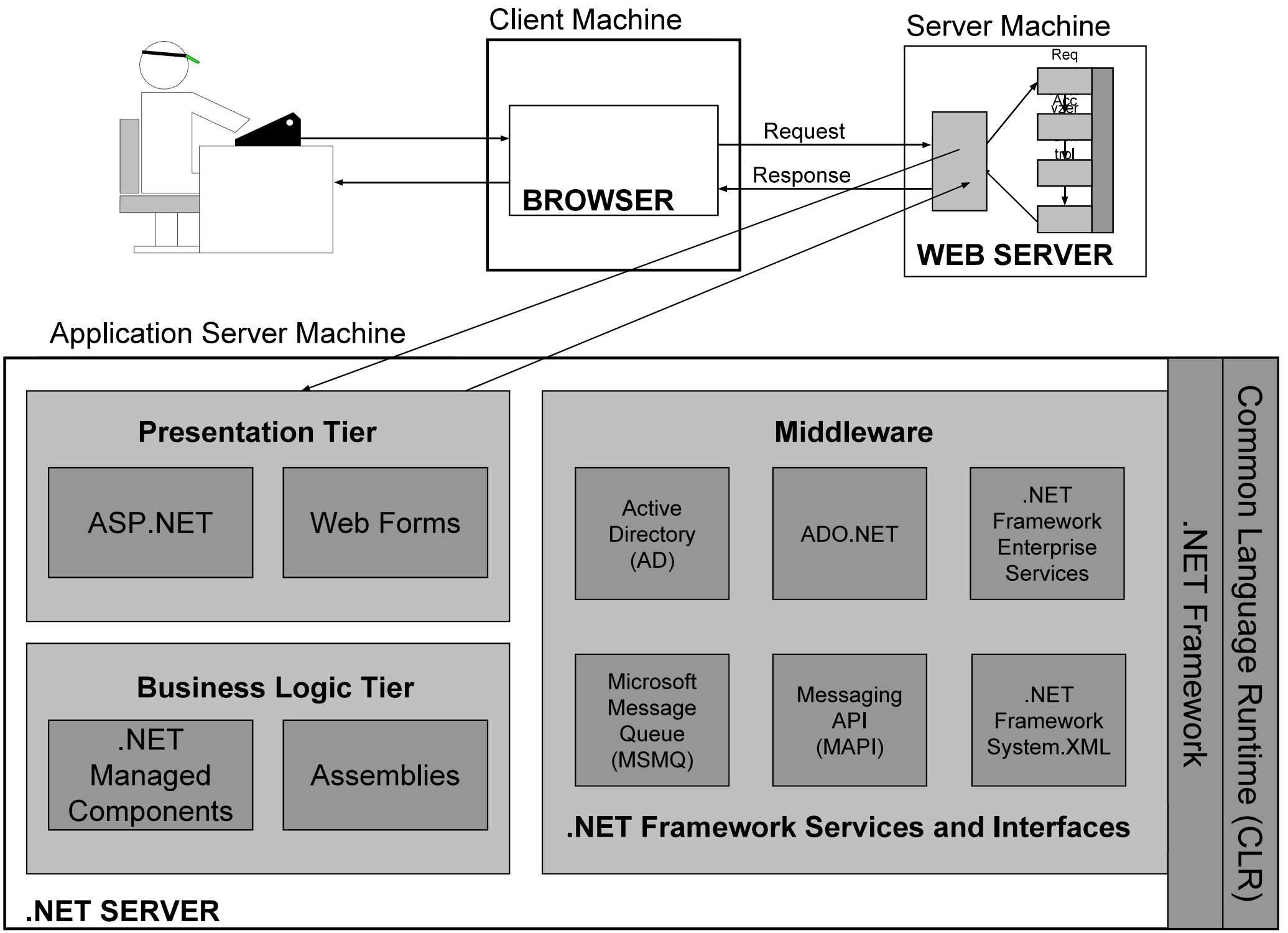
ASP


Request Analyzer
Access Control
Resource Handler
Transaction Log
Operating System Abstraction Layer
Operating System
Request
Response
Reception
BROWSER
Client Machine
ASP Interpreter (asp.dll)
WEB SERVER
ASP Templates
Server Machine
COM Objects

Scaling Up

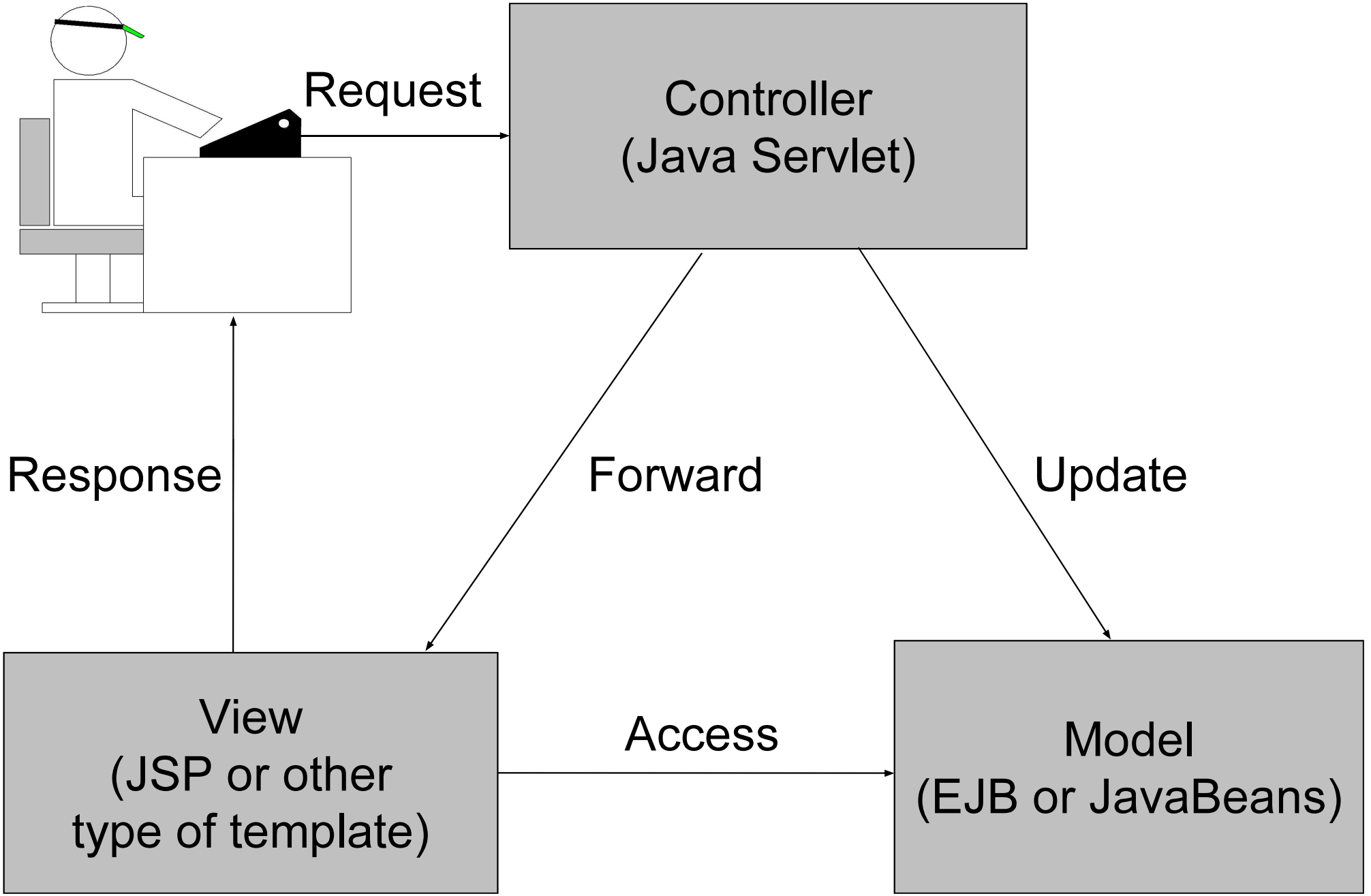
- Middleware
  - Reliability
  - Throughput
  - Integration
  - Security
  - Development
- Application servers and components
- Java
  - Servlets
  - JSP
  - J2EE
- .NET

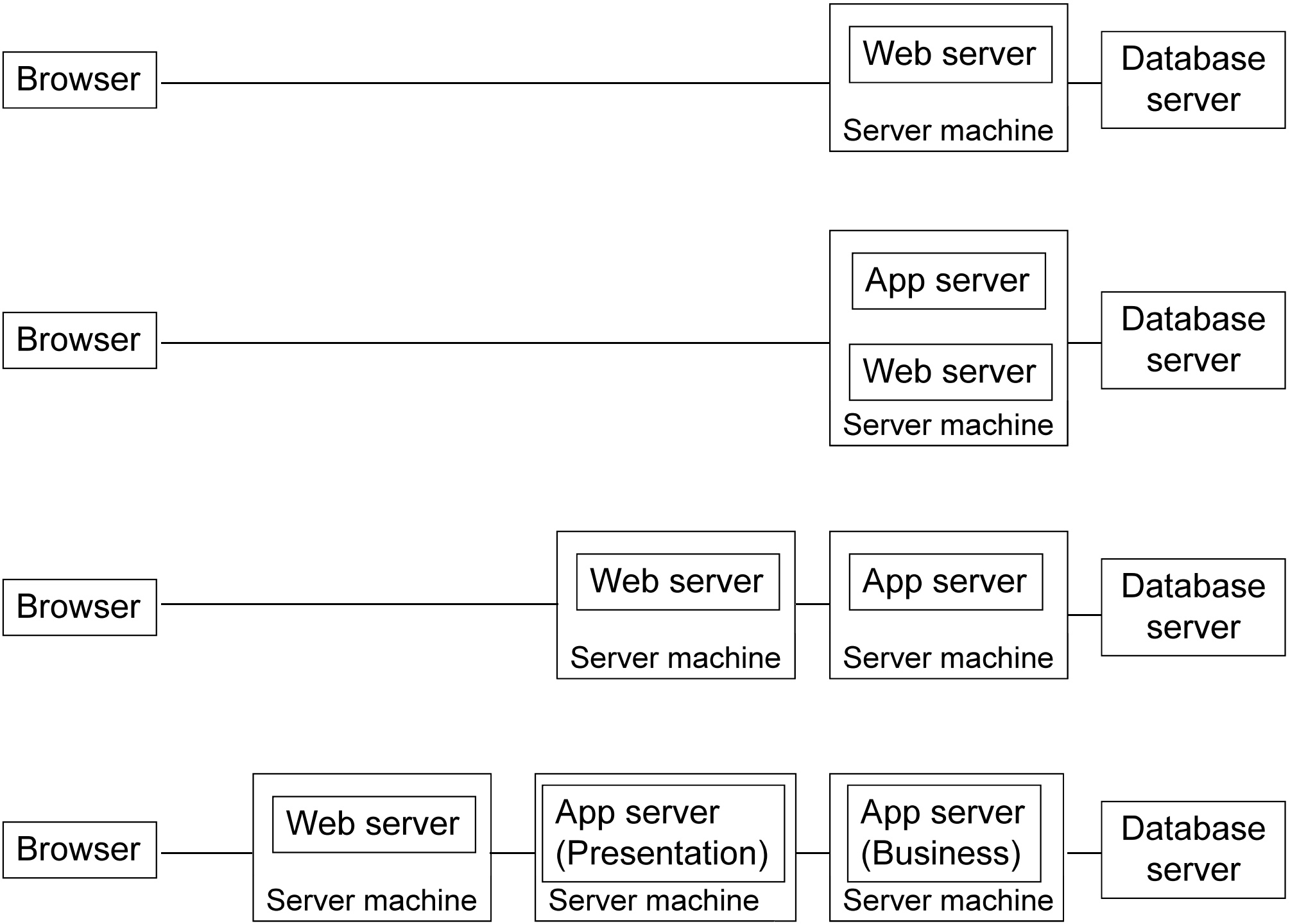

Client Machine
Server Machine
Req
Request
Response
BROWSER
WEB SERVER
Application Server Machine
Presentation Tier
Java Servlets
JavaServer Pages (JSP)
WEB CONTAINER
Business Logic Tier
EJB Components
Java Classes
EJB CONTAINER
Middleware
Java Naming and Directory Interface (JNDI)
Java Database Connectivity (JDBC)
Java Transaction Server (JTS)
Java Messaging Service (JMS)
Java Connector Architecture (JCA)
Java Auth. Service (JAAS)
J2EE Services and Interfaces
Java 2, Standard Edition (J2SE)
Java Virtual Machine (JVM)
J2EE SERVER


Client Machine
Server Machine
Req
BROWSER
Request
Response
WEB SERVER
Application Server Machine
Presentation Tier
ASP.NET
Web Forms
Business Logic Tier
.NET Managed Components
Assemblies
Middleware
Active Directory (AD)
ADO.NET
.NET Framework Enterprise Services
Microsoft Message Queue (MSMQ)
Messaging API (MAPI)
.NET Framework System.XML
.NET Framework Services and Interfaces
.NET Framework
Common Language Runtime (CLR)
.NET SERVER

Web programming vs. regular programming

- Web development traditionally lagged state-of-the-art, until J2EE.
- Approaches carried forward to the Web
  - Patterns
  - Tiered architectures
  - Frameworks
    - Persistence
    - Lightweight containers
    - WebMVC


Request
Controller
(Java Servlet)
Response
Forward
Update
View
(JSP or other
type of template)
Access
Model
(EJB or JavaBeans)


Browser
Web server
Server machine
Database server
Browser
App server
Web server
Server machine
Database server
Browser
Web server
Server machine
App server
Server machine
Database server
Browser
Web server
Server machine
App server (Presentation)
Server machine
App server (Business)
Server machine
Database server

Summary Classification

Technology	Tier	Implementations
Markup Standards	Client	HTML, XHTML, CSS
Protocol specifications	Server	HTTP, SSL, MIME, WebDAV
Web browser	Client	Internet Explorer, Opera, Mozilla, Netscape
Web server	Server	Apache, Internet Information Server, Sun Java Web Server, Zeus Web Server, Jigsaw

Table V. Foundational technologies for the Web.

Technology	Exemplars	Key Properties and Common Usage
Browser interfaces		
State management	Netscape Cookie API	Used for client state persistence.
Web service client interfaces	SOAP, XML-RPC	Highly portable. Increased interactivity.
Scripting interfaces	XMLHttpClient	Highly portable. Increased interactivity.

Table VII. Client-based integration technologies.

Technology	Exemplars	Key Properties and Common Usage
Programming languages		
Natively compiled	C, C++	Highly portable, high performance, low usability. CGI scripting and web server extension.
Interpreted	Perl, Python, Ruby	Highly portable. Performance varies. CGI scripting.
Byte-code compiled	Java, .NET languages	Highly portable. Good performance. Extensive language run-time libraries.
Components		
Complex components	CORBA, COM, EJB, .NET managed components	Enterprise systems development.
Simple components	JavaBeans, Spring, picoContainer, Java classes	Enterprise systems development. Improved usability.
Middleware		
XML transformation	XSLT, Cocoon	XML Web publishing.
Messaging	MSMQ, MQSeries, J2EE JMS	Enterprise systems development.
Security	J2EE JAAS	Enterprise systems development.
Data access	ODBC, JDBC, ADO.NET	Highly portable. Highly scalable.
Object-relational mapping	Hibernate, JDO, TopView, iBatis	Highly portable. Highly scalable. Improved usability.
Web services	SOAP, XML-RPC	Highly portable.

Table VIII. Server-side integration technologies.

Technology	Exemplars	Key Properties and Common Usage
Browser extension		
Browser specific APIs	CCI, ActiveX, Netscape Plug-API	High performance, low usability. Extend browser capabilities.
Client dynamism		
Directly interpreted	JavaScript, VBScript, DOM, SVG	Highly portable due to Web standards acceptance. Increased interactivity.
Byte-code	Macromedia Flash, Java applets	Highly portable depending on plug-in spread. Marketing presentation development.
Rich interfaces		
Browser alternatives	CURL, Sash Weblications, Konfabulator, XAML	Used for applications with high interactivity requirements that need to access the Internet.
Client frameworks	Jakarta Commons http-Client	Used for browser alternative development.
Forms interfaces	InfoPath, XForms	Used by forms-based business systems.
Dynamic assembly	Edge Side Includes (ESI), Client Side Includes (CSI)	Improved cached content delivery.

Table IX. Client-based dynamic content generation technologies.

Technology	Exemplars	Key Properties and Common Usage
Server extension		
Server-specific	ISAPI, NSAPI, Apache API	Proprietary. Scalable. Complex. Implement extended server capabilities.
Gateways		
Simple	CGI	Portable. Not scalable. Low usability. Small-scale applications with simple navigational requirements.
Scalable	Fast CGI	Portable. Scalable. Low usability. Medium to large-scale applications with simple navigation requirements.
Interpreters		
General purpose	Server-side JavaScript, mod-perl	Medium to large-scale applications with simple navigation requirements.
Template-based	SSI, XSSI, ColdFusion, ASP, PHP, JWIG	Portable. Medium to large-scale applications with simple navigation requirements.
Extended servers		
Servlet engines	Tomcat, Resin	Highly portable. Scalable. Medium to enterprise-scale applications. Frameworks.
Template engines	JSP, Velocity, WebMacro, XMLC, FreeMarker	Portable. Scalable. Generally used as the view component of MVC implementations.
Content management	Zope, Cocoon	Scalable. Web publishing.
J2EE application servers	WebSphere, jRun, JBOSS, WebLogic	Highly portable. Highly scalable. Complex. Enterprise systems.

Table X. Server-side dynamic content generation technologies.

Technology	Exemplars	Key Properties and Common Usage
Frameworks		
Scripting language-based	Twisted (Python), WebWare (Python), Snakelets (Python), Rails (Ruby)	Portable. Small to large-scale applications with complex navigation requirements.
Application-driven	Struts, Spring MVC, WebWork, Maverick, Barracuda	Portable. Scalable. Enterprise web sites with complex navigation requirements.
Page-driven, component-based, desktop model	Echo, wingS, WebCream, Wi.Ser	Portable. Complex. Ease transition to Web programming.
Page-driven component-based, Web model	WebObjects, Tapestry, ASP.NET WebForms	Portable. Scalable. High usability. Rapid development. Simple to medium-complexity navigation requirements.
Application/Page Hybrid-driven component-based	JavaServer Faces	Portable. Scalable. High usability. Enterprise web sites with complex navigation requirements. Rapid development.
Portal composition	Java Portlet specification, WebParts, Tiles, SiteMesh	Portable. Scalable. Enterprise portal development.

Table XI. Server-side development approaches.

Model Driven Development		
Data-centered	AutoWeb, RMM, OOHDM, Oracle Web Development Suite, WebML, WebRatio	Data-intensive applications.
GUI-centered	CodeCharge, CodeSmith, DeKlarit, Fabrique	Interation-intensive applications.
MDA-compliant	Oracle ADF, IBM/Rational Rapid Developer	OMG standard.
Development Environments		
Authoring tools	Adobe PageMill, Amaya, Microsoft Office (Word, Excel, PowerPoint)	Static content creation and editing.
Web site management	FrontPage, NetObjects Fusion, MacroMedia Dreamweaver	Mostly static web sites with simple navigation requirements.
Full-fledged environments	Microsoft VisualStudio.NET, Sun Java Studio Creator, WebSphere Application Developer	Programming-intensive.
Others		
Programming languages	MAWL, <bigwig>, JWIG, functional continuations, Cocoon Flow	Research projects.

Table XI. Server-side development approaches.

Conclusions

- Infrastructure
  - Scalability problem is largely solved.
    - Google
    - Load balancing
- Application Development
  - Current methods don't scale well for people.
  - Can be many modes to deal with.
  - May be able to formulate a simpler, more concise model for Web application development.