Disruptive Middleware: Past, Present, and Future

Lucy Cherkasova Hewlett-Packard Labs

From Internet Data Centers to Data Centers in the Cloud

- Data Centers Evolution
  - Internet Data Centers
  - Enterprise Data Centers
  - Web 2.0 Mega Data Centers

Performance

and

Modeling

Challenges

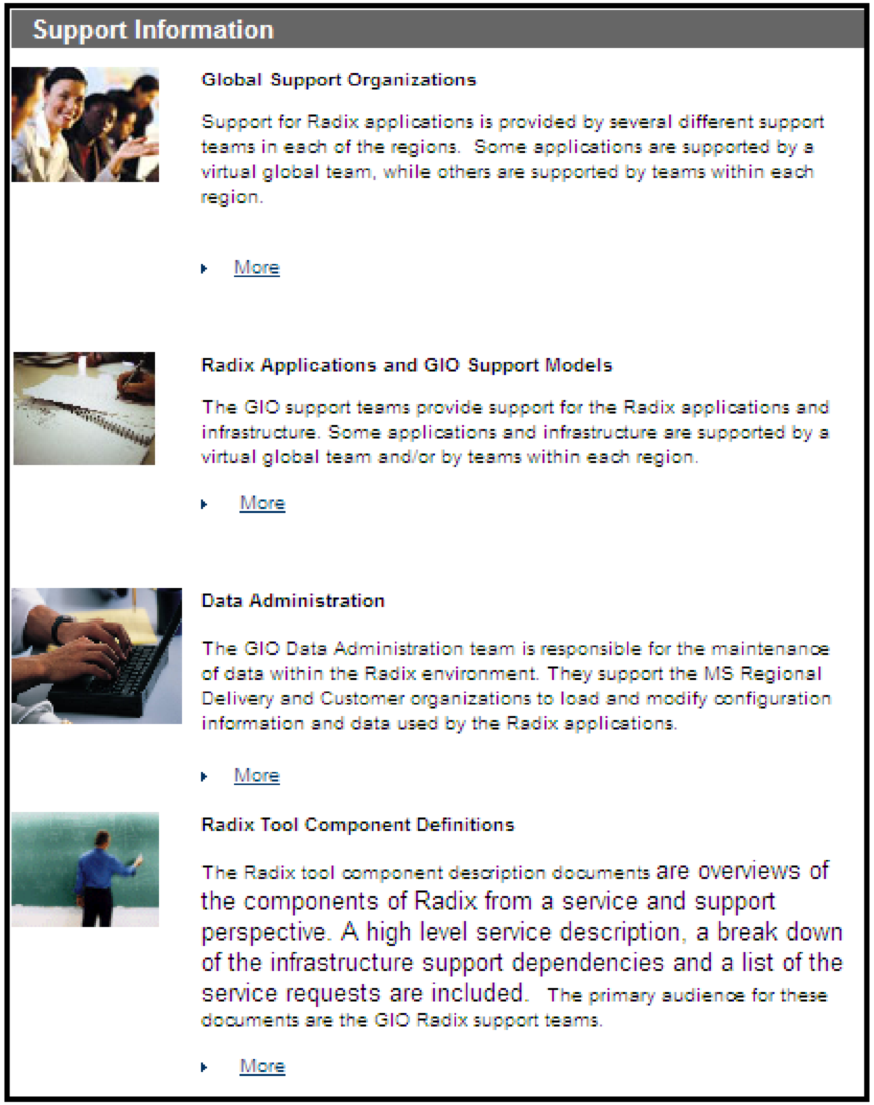
Data Center Evolution

- Internet Data Centers (IDCs first generation)
  - Data Center boom started during the dot-com bubble
  - Companies needed fast Internet connectivity and an established Internet presence
  - Web hosting and collocation facilities
  - Challenges in service scalability, dealing with flash crowds, and dynamic resource provisioning
    - New paradigm: everyone on the Internet can come to your web site!
  - Mostly static web content
    - Many results on improving web server performance, caching, and request distribution
  - Web interface for configuring and managing devices
  - New pioneering architectures such as
    - Content Distribution Network (CDN),
    - Overlay networks for delivering media content

Content Delivery Network (CDN)

- High availability and responsiveness are key factors for business Web sites
- "Flash Crowd" problem
- Main goal of CDN's solution is
  - overcome server overload problem for popular sites,
  - minimize the network impact in the content delivery path.
- CDN: large-scale distributed network of servers,
  - Surrogate servers (proxy caches) are located closer to the edges of the Internet.
- Akamai is one of the largest CDNs
  - 56,000 servers in 950 networks in 70 countries
  - Deliver 20% of all Web traffic

Retrieving a Web Page



Support Information

Global Support Organizations

Support for Radix applications is provided by several different support teams in each of the regions. Some applications are supported by a virtual global team, while others are supported by teams within each region.

▸ More

Radix Applications and GIO Support Models

The GIO support teams provide support for the Radix applications and infrastructure. Some applications and infrastructure are supported by a virtual global team and/or by teams within each region.

▸ More

Data Administration

The GIO Data Administration team is responsible for the maintenance of data within the Radix environment. They support the MS Regional Delivery and Customer organizations to load and modify configuration information and data used by the Radix applications.

▸ More

Radix Tool Component Definitions

The Radix tool component description documents are overviews of the components of Radix from a service and support perspective. A high level service description, a break down of the infrastructure support dependencies and a list of the service requests are included. The primary audience for these documents are the GIO Radix support teams.

▸ More

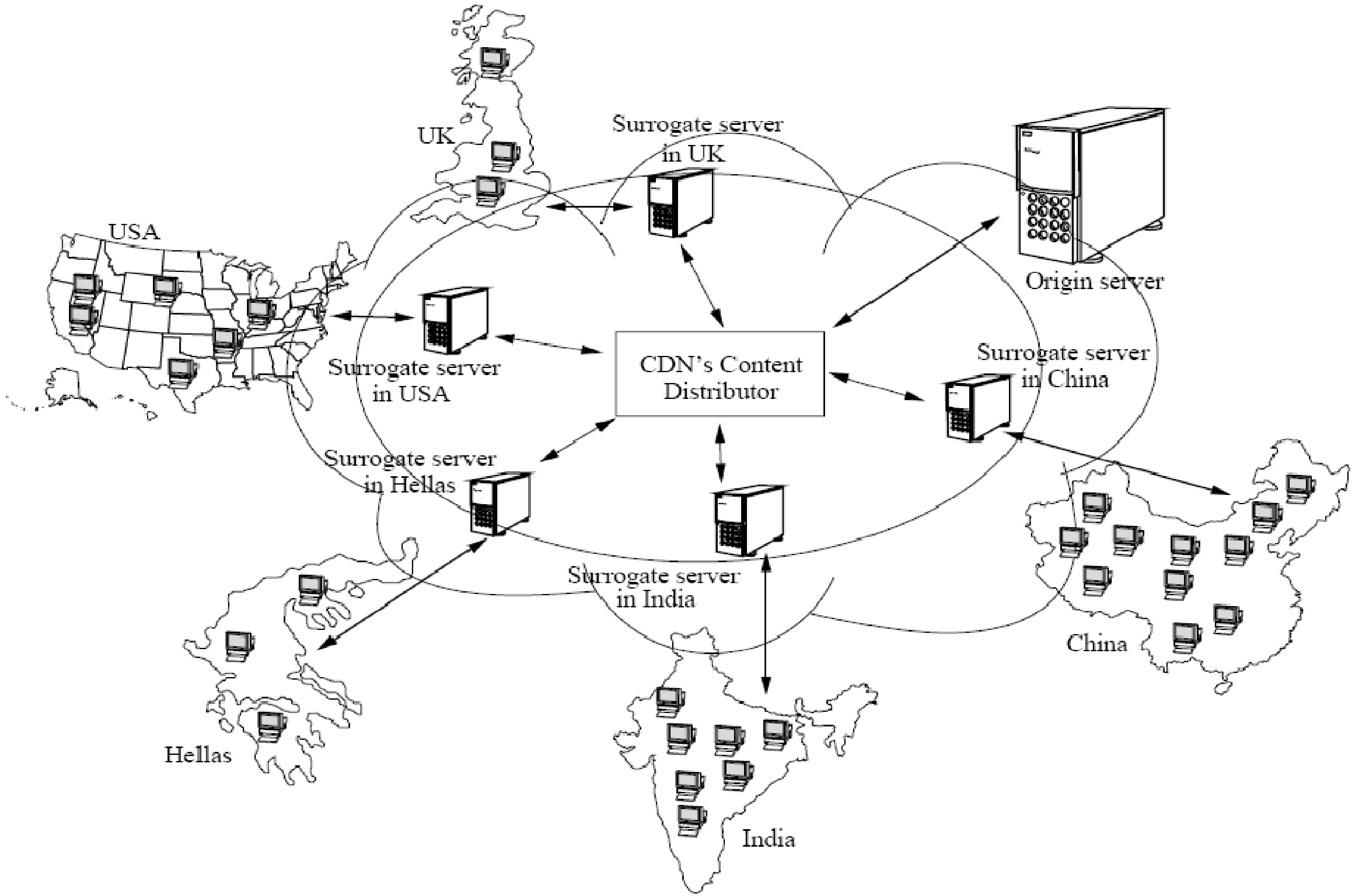
Web page is a composite object:

- HTML file is delivered first
- Client browser parses it for embedded objects
- Send a set of requests for this embedded objects
- Typically, 80% or more of a web page are images
- 80% of the page can be served by CDN.

CDN's Design

- Two main mechanisms
  - URL rewriting
    - <img src =http://www.xyz.com/images/foo.jpg>
    - <img src =http://akamai.xyz.com/images/foo.jpg>
  - DNS redirection
    - Transparent, does not require content modification
    - Typically employs two-level DNS system to choose most appropriate edge server

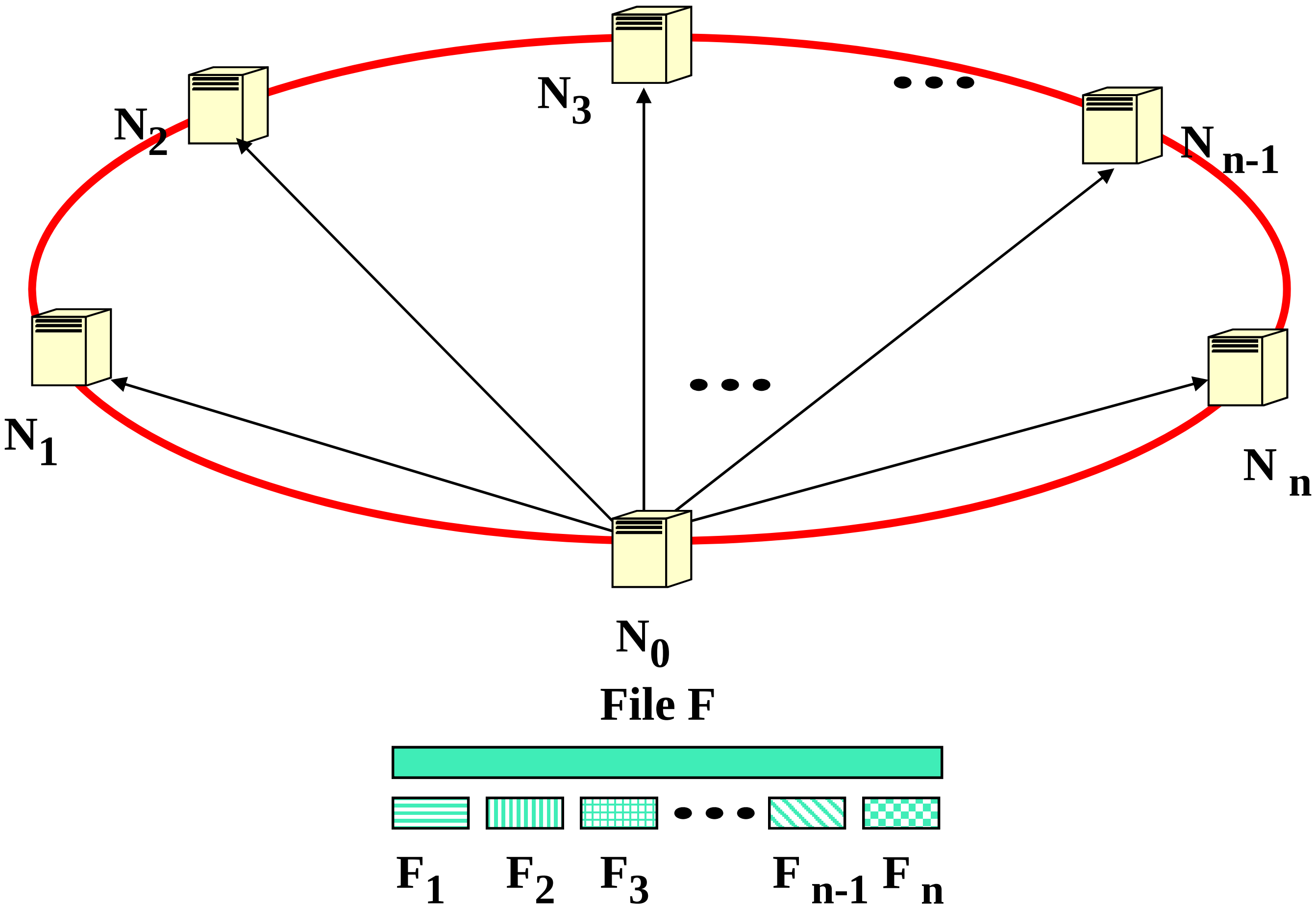
CDN Architecture



Research Problems

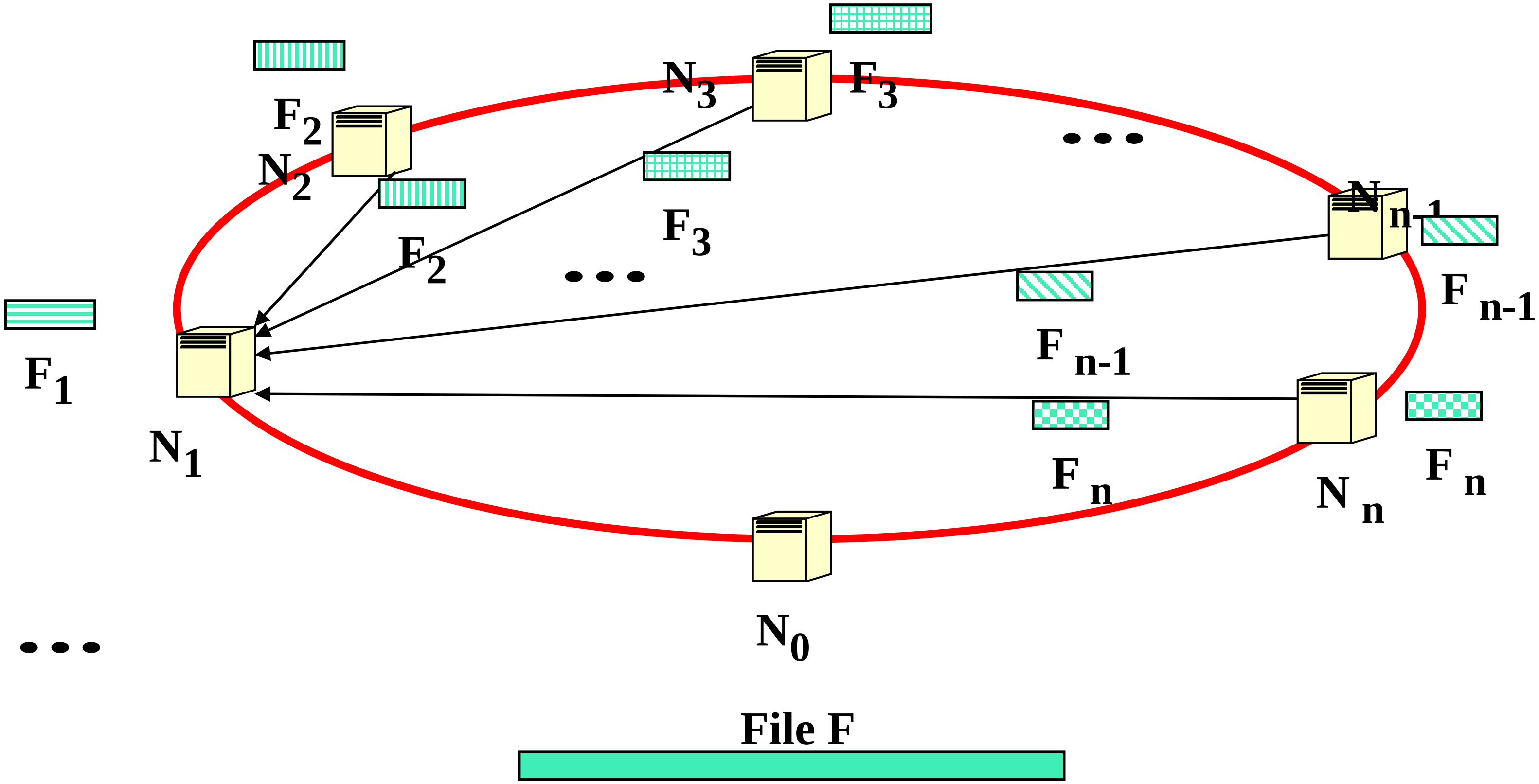
- Efficient large-scale content distribution
  - large files, video on demand, streaming media
    - FastReplica for CDNs
    - BitTorrent (general purpose)
    - SplitStream (multicast, video streaming)

FastReplica: Distribution Step



L. Cherkasova, J. Lee. *FastReplica: Efficient Large File Distribution within Content Delivery Networks*
Proc. of the 4th USENIX Symp. on Internet Technologies and Systems (USITS'2003).

FastReplica: *Collection Step*

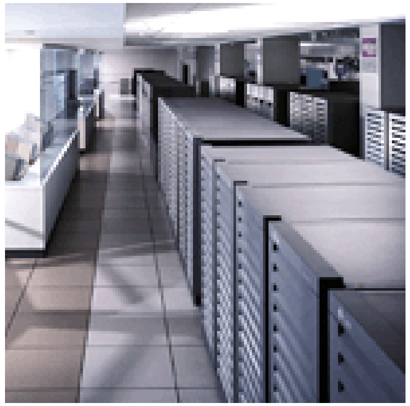


Research Problems

Some (still) open questions:

- Optimal number of edge servers and their placement
  - Two different approaches:
    - *Co-location*: placing servers closer to the edge (Akamai)
    - *Network core*: server clusters in large data centers near the main network backbones (Limelight and AT&T)
- Content placement
- Large-scale system monitoring and management

Data Center Evolution



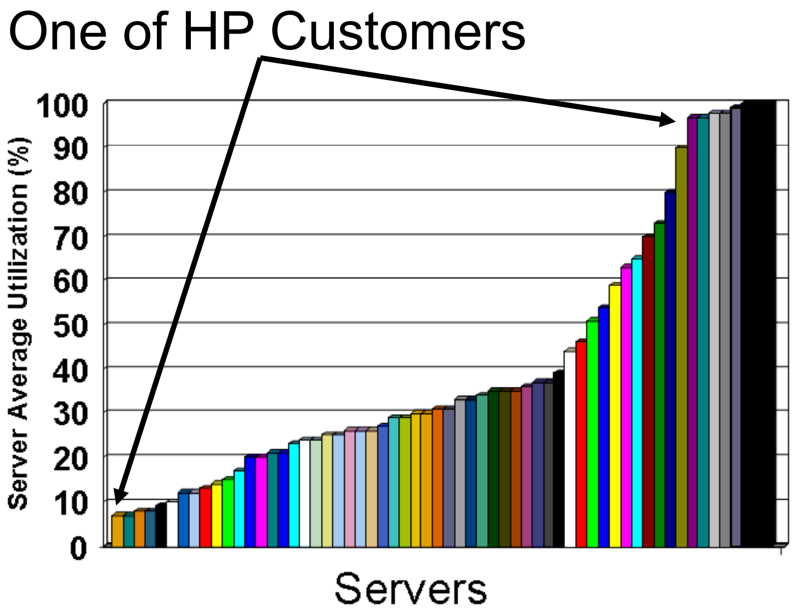
- Enterprise Data Centers
  - New application design: multi-tier applications
  - Many traditional applications, e.g. HR, payroll, financial, supply-chain, call-desk, etc, are re-written using this paradigm.
  - Many different and complex applications
  - *Trend: Everything as a Service*
    - Service oriented Architecture (SOA)
  - Dynamic resource provisioning
  - Virtualization (datacenter middleware)
  - Dream of Utility Computing:
    - Computing-on-demand (IBM)
    - Adaptive Enterprise (HP)

hp invent

Enterprise computing workloads

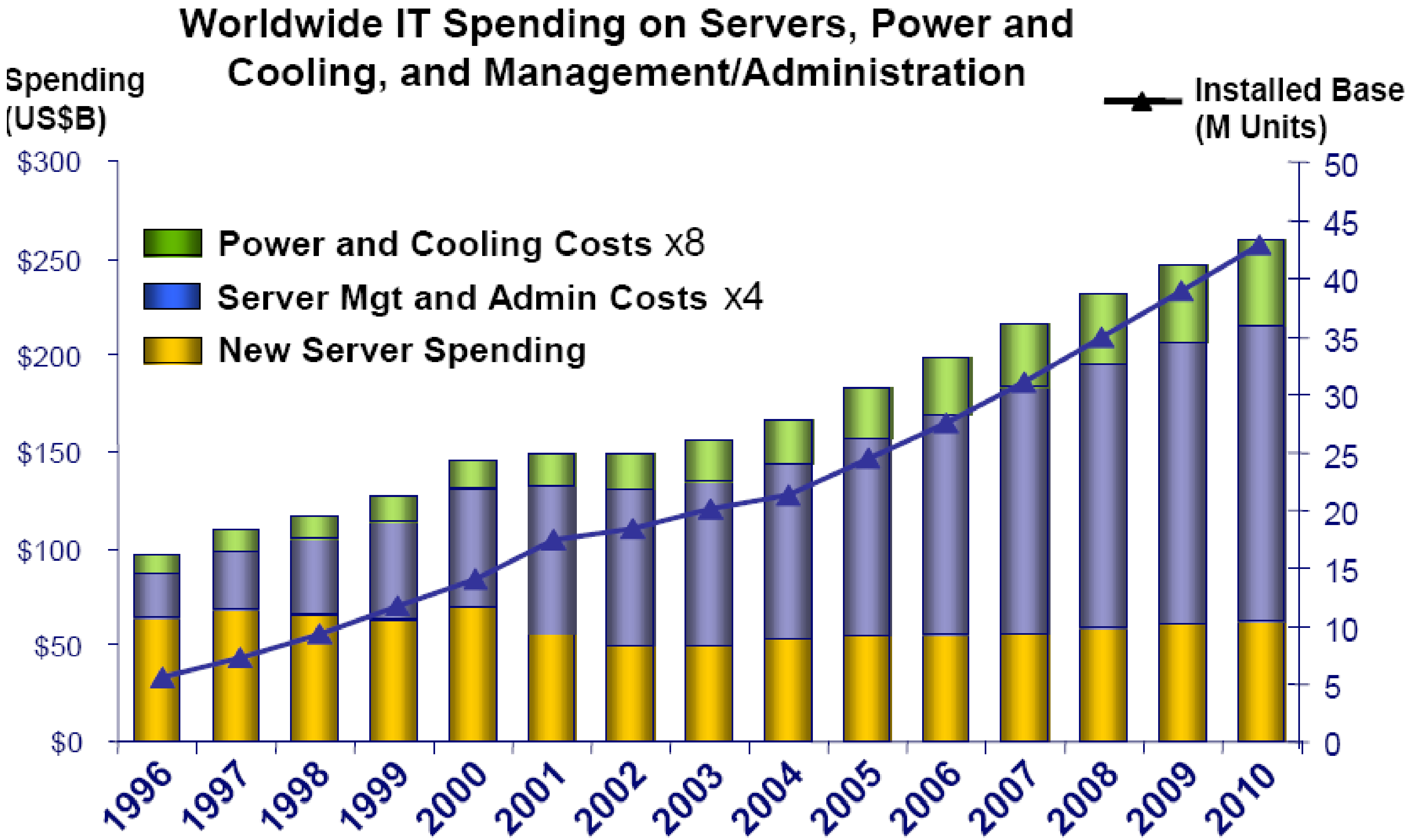
- Applications often assigned *dedicated* resources

- Issues
  - Low utilizations
  - Inflexible
    - takes time to acquire/deploy new resources
  - High management costs
    - Increased space, power, and maintenance effort



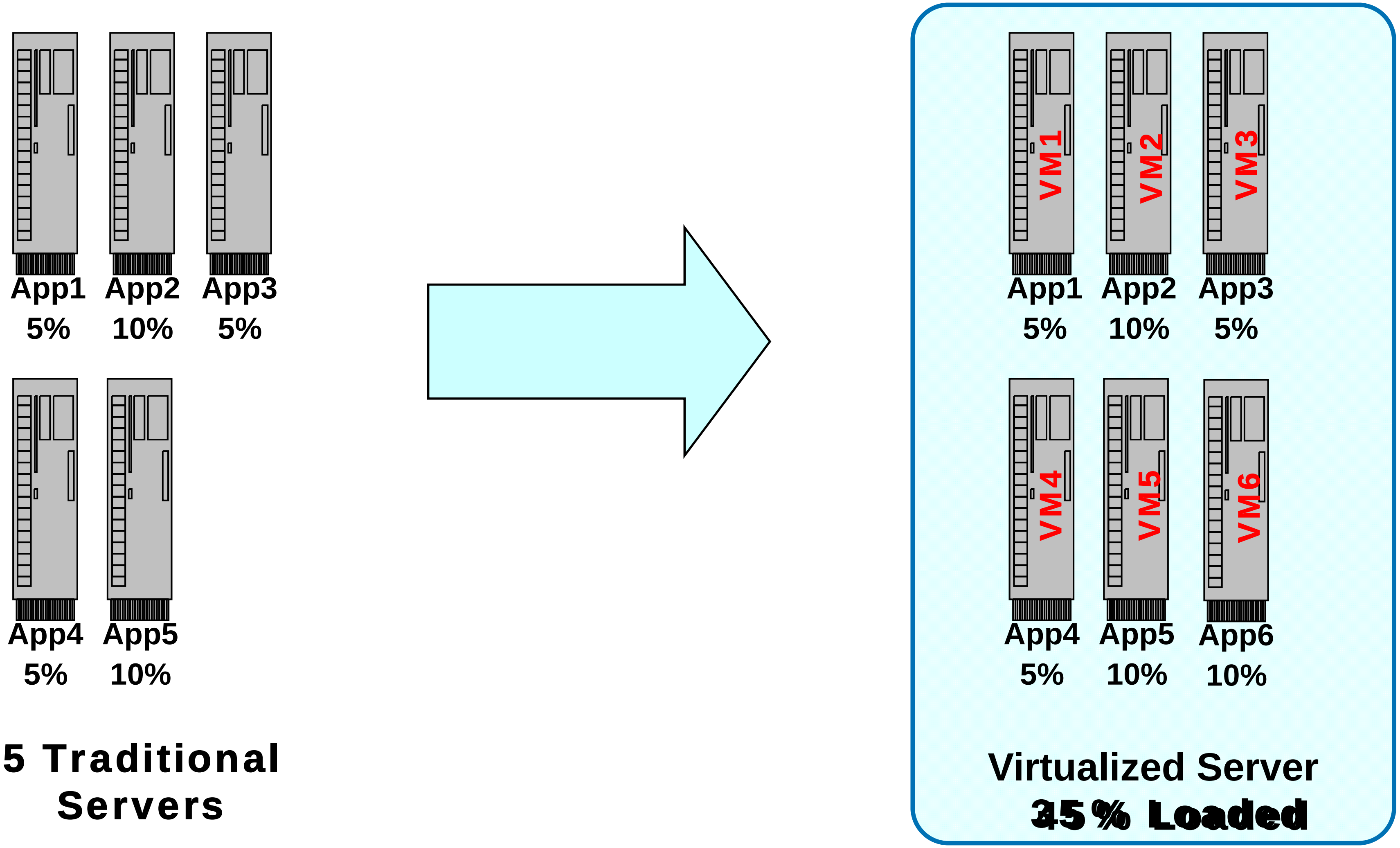
Worldwide Server Market:

Cost of Management and Power Ramps Dramatically



Source: IDC, 2008

Server Consolidation via Virtualization



Shared virtualized server pool:
utilization and power optimization

Evolution of the HP IT Environment

Adaptive (Business Processes)

Efficient (Applications)

Stable (Infrastructure)

Pre-merger (2001)	2005	2009
7,000+ applications	4,000 applications	1,500 applications
25,000 servers	19,000 servers	10,000 servers
300 Data Centers	85 Data Centers	6 Data Centers
IT cost = 4.6% of revenue	IT cost = 4% of revenue	IT cost = 2.0% of revenue

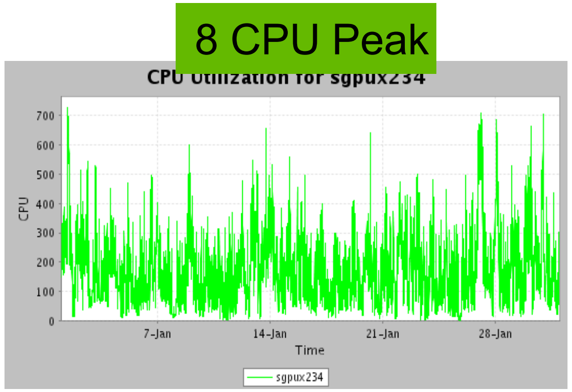
Virtualization and Automation are the key capabilities in NGDC

Virtualized Data Centers

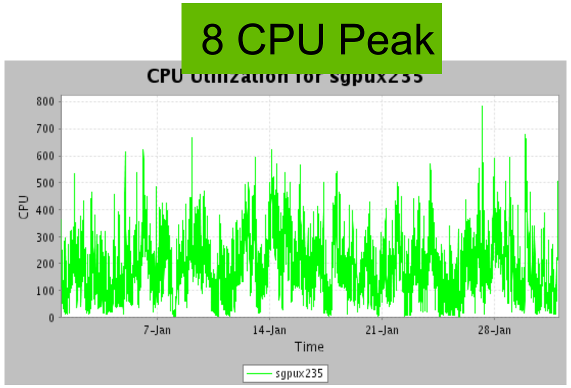
- Benefits
  - Fault and performance isolation
  - Optimized utilization and power
  - Live VM migration for management
- Challenges
  - Efficient capacity planning and management for server consolidation
    - Apps are characterized by a collection of resource usage traces in *native environment*
    - *Effects of consolidating* multiple VMs to one host
    - *Virtualization overheads*

Capacity Planning and Management

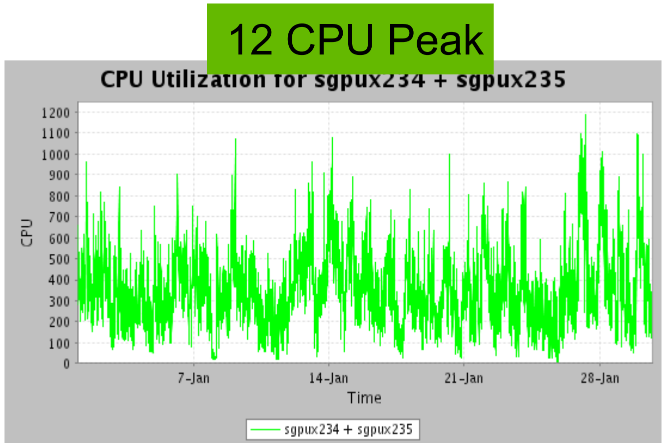
Trace-based approach



+



=



- Peaks for different workloads do not all happen at the same time.
- Two workloads each have an 8 CPU peak demand but the peak of their sum is 12 CPUs.

The new math: 8+8 = 12

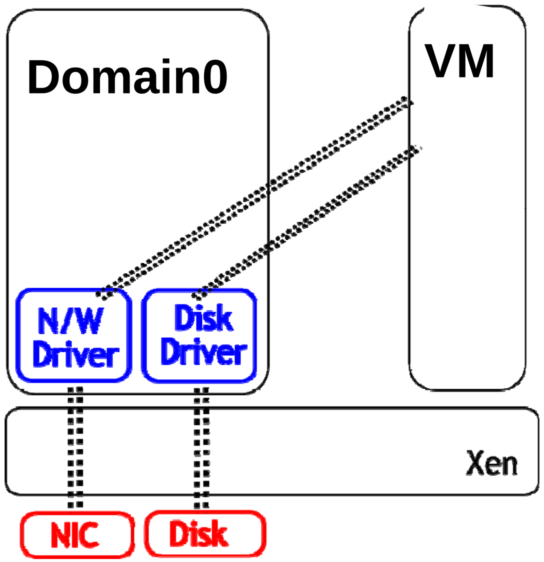
Application Virtualization Overhead

- Many research papers measure virtualization overhead but ***do not predict*** it in a general way:
  - A particular hardware platform
  - A particular app/benchmark, e.g., netperf, Spec or SpecWeb, disk benchmarks
  - Max throughput/latency/performance is X% worse
  - Showing Y% increase in CPU resources
- How do we translate these measurements in "**what is a virtualization overhead for a given application**"?

New performance models are needed

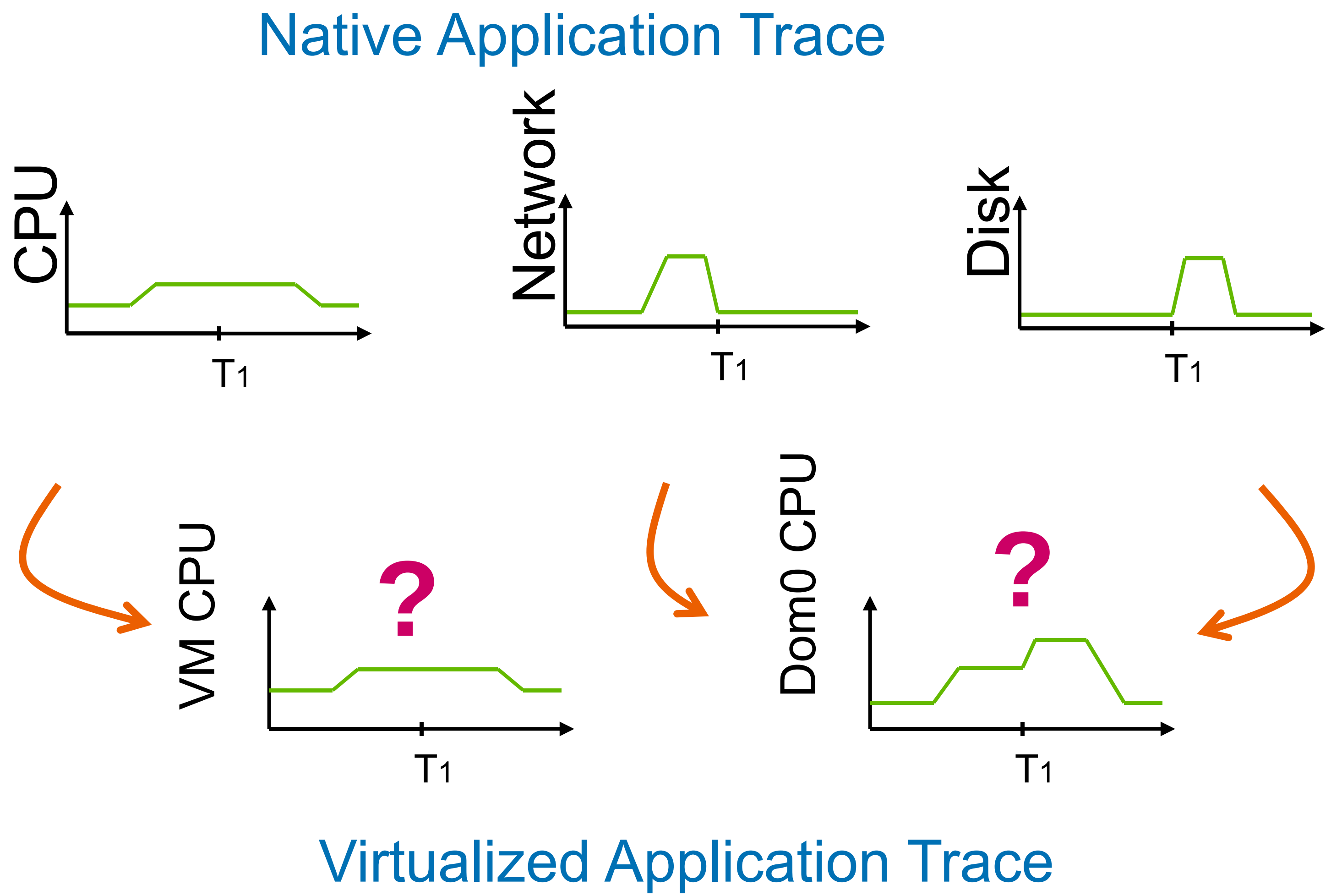
Predicting Resource Requirements

- Most overhead caused by I/O
  - Network and Disk activity
- Xen I/O Model
- 2 components
  - Dom0 handles I/O
- Must predict CPU needs of:
  1. Virtual machine running the application
  2. Domain 0 performing I/O on behalf of the app



Requires several prediction models based on multiple resources

Problem Definition



T. Wood, L. Cherkasova, K. Ozonat, P. Shenoy: *Profiling and Modeling Resource Usage of Virtualized Applications.* Middleware'2008.

Relative Fitness Model

- Automated robust model generation
- Run benchmark set on native and virtual platforms
  - Performs a range of I/O and CPU intensive tasks
  - Gather resource traces

- Build model of Native --> Virtual relationship
  - Use linear regression techniques
  - Model is specific to platform, but not applications
- Black box approach

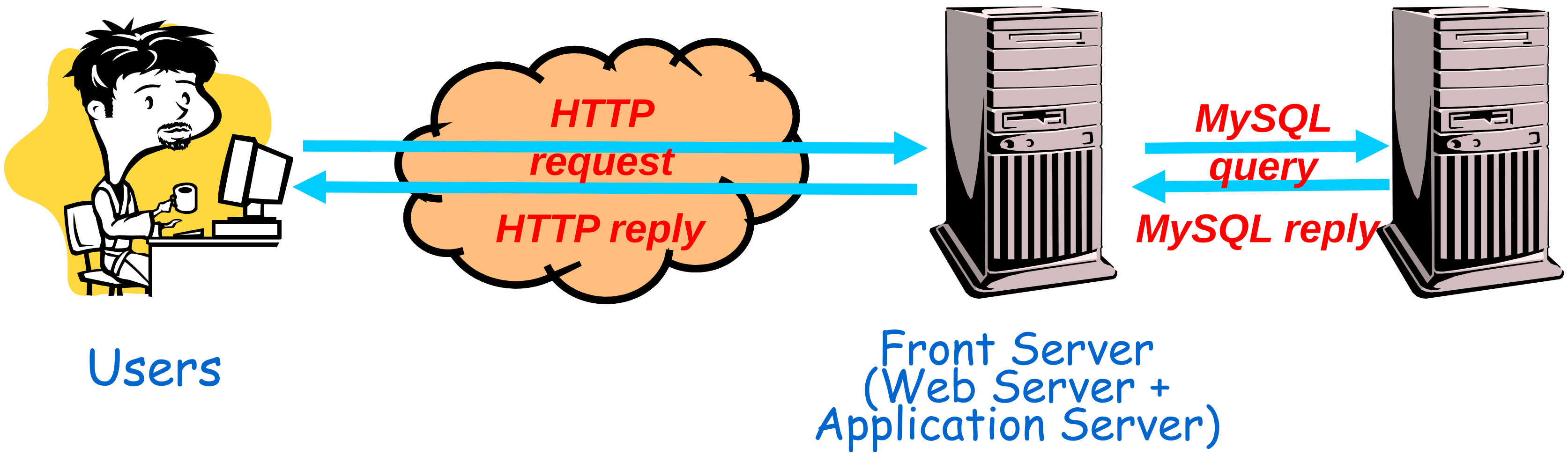
Can apply this general model to any application's traces to predict its requirements

Multi-tier Applications: Motivation

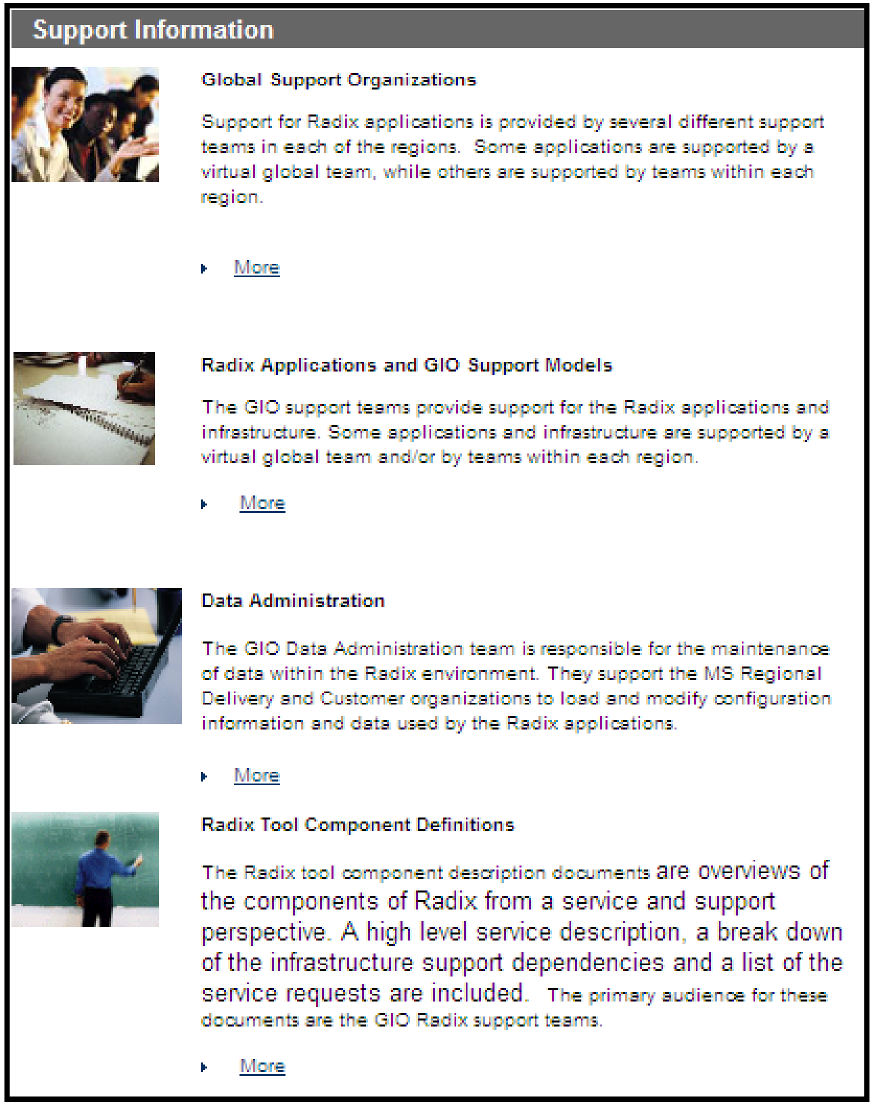
- Wayne Greene's story:
  - Large-scale systems: 400 servers, 36 applications
  - Rapidly evolving system over time
- Questions from service provider on current system:
  - How many additional clients can we support?
  - Anomaly detection or cause of performance problems: workload or software "bugs" ?
- Traditional capacity planning (pre-sizing):
  - Benchmarks
  - Synthetic workloads based on *typical* client behavior
- New models are needed

Multi-tier Applications

- Enterprise applications:
  - Multi-tier architecture is a standard building block



Units of Client/Server Activities: Transactions



Support Information

Global Support Organizations

Support for Radix applications is provided by several different support teams in each of the regions. Some applications are supported by a virtual global team, while others are supported by teams within each region.

More

Radix Applications and GIO Support Models

The GIO support teams provide support for the Radix applications and infrastructure. Some applications and infrastructure are supported by a virtual global team and/or by teams within each region.

More

Data Administration

The GIO Data Administration team is responsible for the maintenance of data within the Radix environment. They support the MS Regional Delivery and Customer organizations to load and modify configuration information and data used by the Radix applications.

More

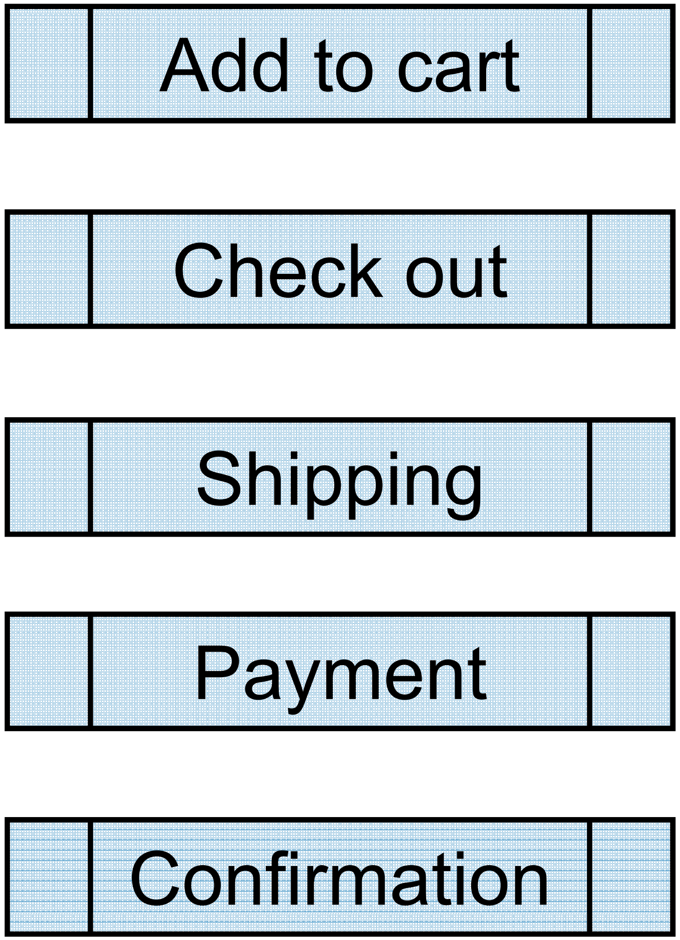
Radix Tool Component Definitions

The Radix tool component description documents are overviews of the components of Radix from a service and support perspective. A high level service description, a break down of the infrastructure support dependencies and a list of the service requests are included. The primary audience for these documents are the GIO Radix support teams.

More

- Web page:
  - An HTML file and several embedded objects (images)
- Transaction = Web page view
- Often, application server is responsible for sending the web page and its embedded objects
- Our task:
  - Evaluate CPU service time for each transaction

Units of Client/Server Activities: Sessions



- Session:

  A sequence of individual transactions issued by the same client

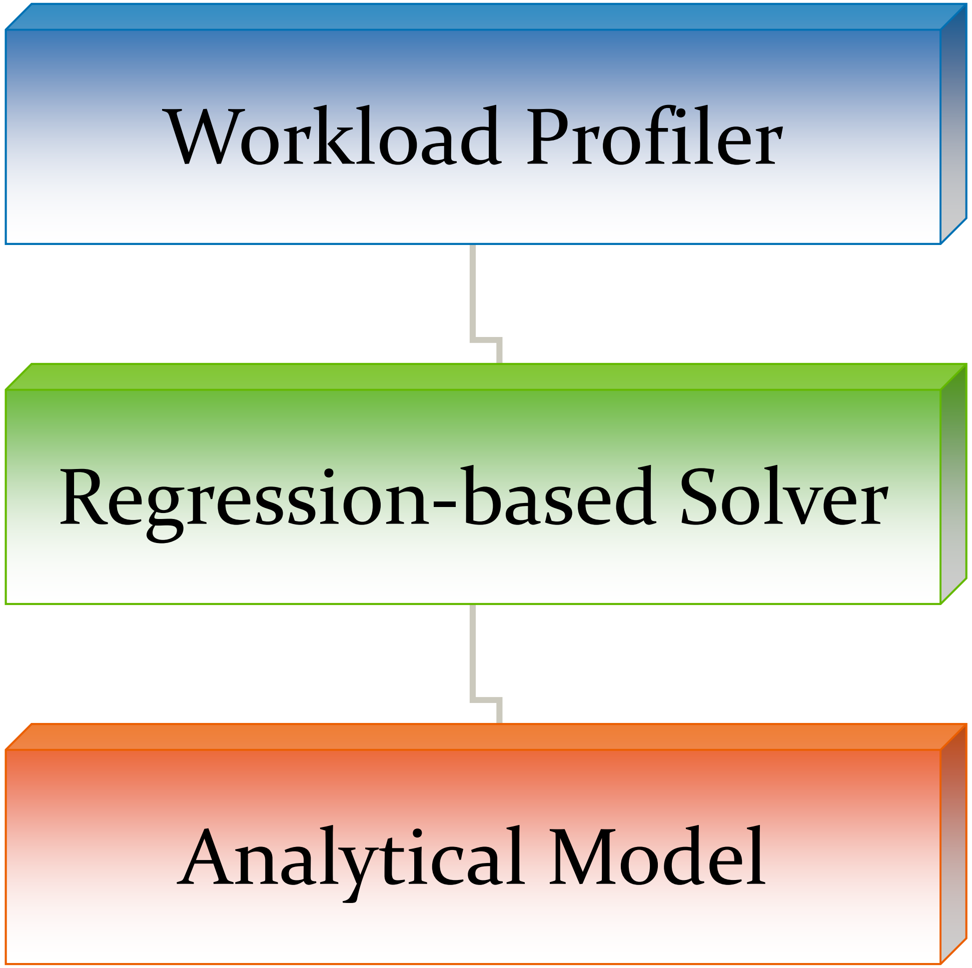
- Concurrent Sessions = Concurrent Clients

- Think time:

  The interval from a client receiving a response to the client sending the next transaction

invent

Automated Capacity Planning Framework



- Extract the profile of the transactions
- Approximate the resource cost of each transaction type
- Solve the system by the analytical model parameterized by resource costs

L. Cherkasova, K. Ozonat, N. Mi, J. Symons, and E. Smirni: *Automated Anomaly Detection and Performance Modeling of Enterprise Applications.* ACM Transactions on Computer Systems, (TOCS), 2009.

Workload Profiler

Time	N_1	N_2	N_3	N_4	…	N_n	U_{CPU}(%)	Think (sec)
1	21	15	21	16	…	0	13.32	72.58
2	24	6	8	5	…	0	8.43	107.06
3	18	2	5	4	…	1	7.41	160.21
4	22	2	4	7	…	0	6.42	173.64
5	38	5	6	7	…	0	7.54	144.85
				…				

Regression

- *Non-negative LSQ Regression* to get cost C_i

$$\sum_i N_i \cdot C_i = U_{CPU} \cdot T$$

$N_1 \quad N_2 \quad \dots \quad N_n \quad U^{front}_{cpu}$

$$\begin{cases} 21C_1 + 15C_2 + \cdots + 20C_{14} = 0.1332*60 \\ 24C_1 + 6C_2 + \cdots + 30C_{14} = 0.0843*60 \\ 18C_1 + 2C_2 + \cdots + 5C_{14} = 0.0741*60 \\ 22C_1 + 2C_2 + \cdots + 12C_{14} = 0.0643*60 \\ 38C_1 + 5C_2 + \cdots + 8C_{14} = 0.0755*60 \\ \cdots \end{cases}$$

Front Server

Model: (C^f_0, C^f_1, … , C^f_n)

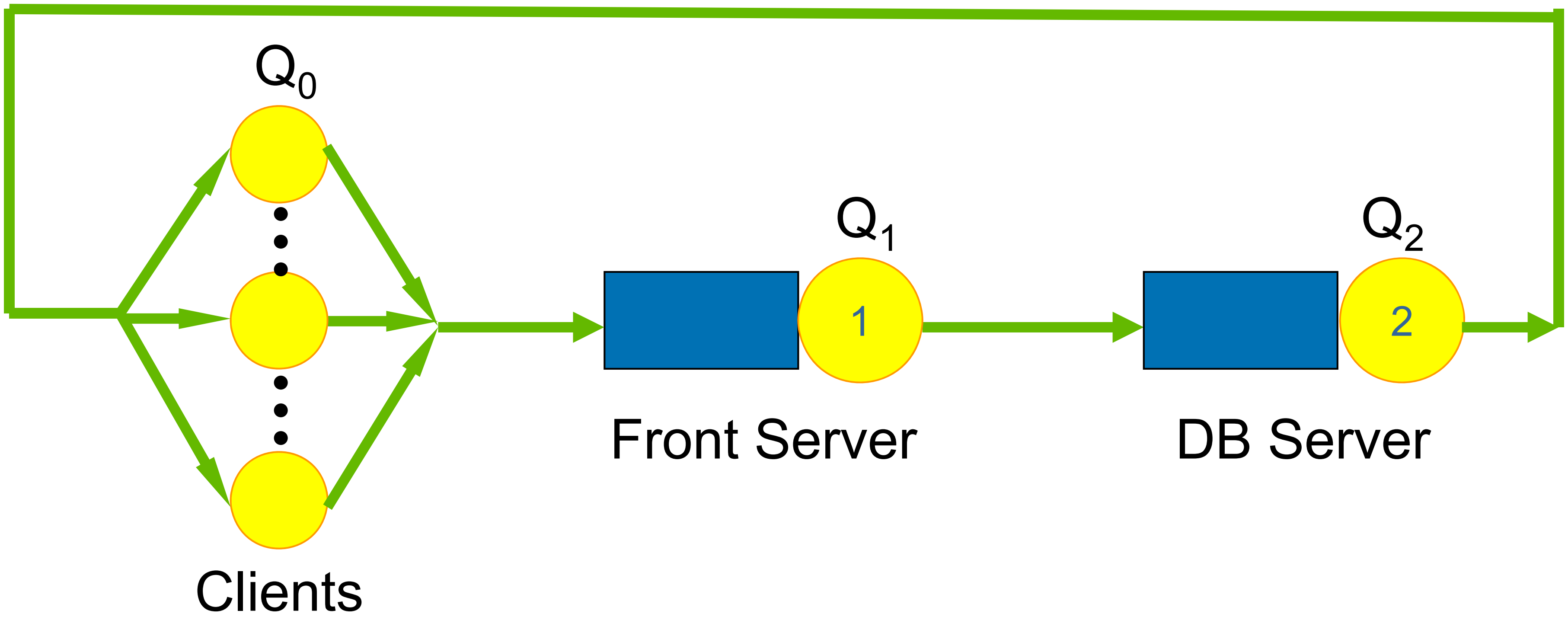
$N_1 \quad N_2 \quad \dots \quad N_n \quad U^{db}_{cpu}$

$$\begin{cases} 21C_1 + 15C_2 + \cdots + 20C_{14} = 0.2662*60 \\ 24C_1 + 6C_2 + \cdots + 30C_{14} = 0.1590*60 \\ 18C_1 + 2C_2 + \cdots + 5C_{14} = 0.2040*60 \\ 22C_1 + 2C_2 + \cdots + 12C_{14} = 0.1589*60 \\ 38C_1 + 5C_2 + \cdots + 8C_{14} = 0.2901*60 \\ \cdots \end{cases}$$

Database Server

Model: (C^{db}_0, C^{db}_1, … , C^{db}_n)

Analytical Model



- A network of queues, each representing a machine
- Model is solved by MVA
- Service time at each tier is parameterized by regression results

Scaling Performance with memcached

- memcahed – distributed memory object caching system for speeding up dynamic web applications by alleviating database load
- Cache the results of popular (or expensive) database queries
- memcahed is an in-memory key-value store for small chunks of arbitrary data (strings, objects) where key is 250 bytes, value is up to 1 MB.
- Used by Facebook, YouTube, LiveJournal, Wikipedia, Amazon.com, etc.
- For example, Facebook use more than 800 memcached servers supplying over 28 terabytes of memory
- *Scalability and performance* are still the most challenging issues for large-scale Internet applications.

Data Growth

- Unprecedented data growth:
  - The amount of managed data by today's Data centers quadruple every 18 months
- New York Stock Exchange generates about 1 TB of new trade data each day.
- Facebook hosts ~10 billion photos (1 PB of storage).
- The Internet Archive stores around 2PB, and it is growing at 20TB per month
- The Large Hadron Collider (CERN) will produce ~15 PB of data per year.

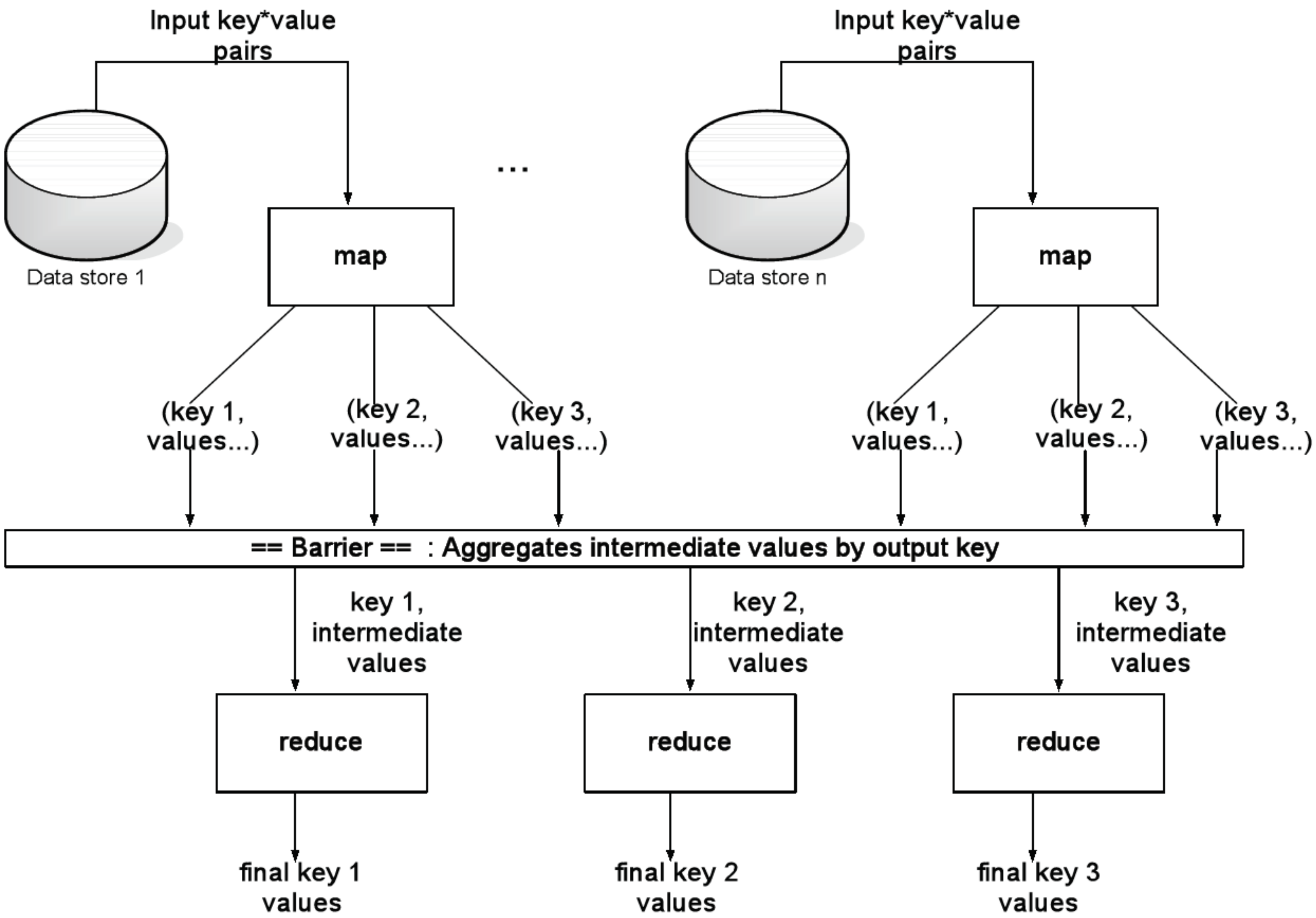
Big Data

- IDC estimate the size of “digital universe” :
  - 0.18 zettabytes in 2006;
  - 1.8 zettabytes in 2011 (10 times growth);
- A zettabyte is 10^{21} bytes, i.e.,
  - 1,000 exabytes or
  - 1,000,000 petabytes
- Big Data is here
  - Machine logs, RFID readers, sensors networks, retail and enterprise transactions
  - Rich media
  - Publicly available data from different sources
- New challenges for storing, managing, and processing large-scale data in the enterprise (information and content management)
  - Performance modeling of new applications

Data Center Evolution

- Data Center in the Cloud
  - Web 2.0 Mega-Datacenters: Google, Amazon, Yahoo
  - Amazon Elastic Compute Cloud (EC2)
  - Amazon Web Services (AWS) and Google AppEngine
  - New class of applications related to parallel processing of large data
  - Map-Reduce framework (with the open source implementation Hadoop)
    - *Mappers* do the work on data slices, *reducers* process the results
    - Handle node failures and restart failed work
  - One can rent its own Data Center in the Cloud on "pay-per-use" basis
  - Cloud Computing: Software as a Service (SaaS) + Utility Computing

MapReduce Data Flow



Slide from the Google's Tutorial on MapReduce

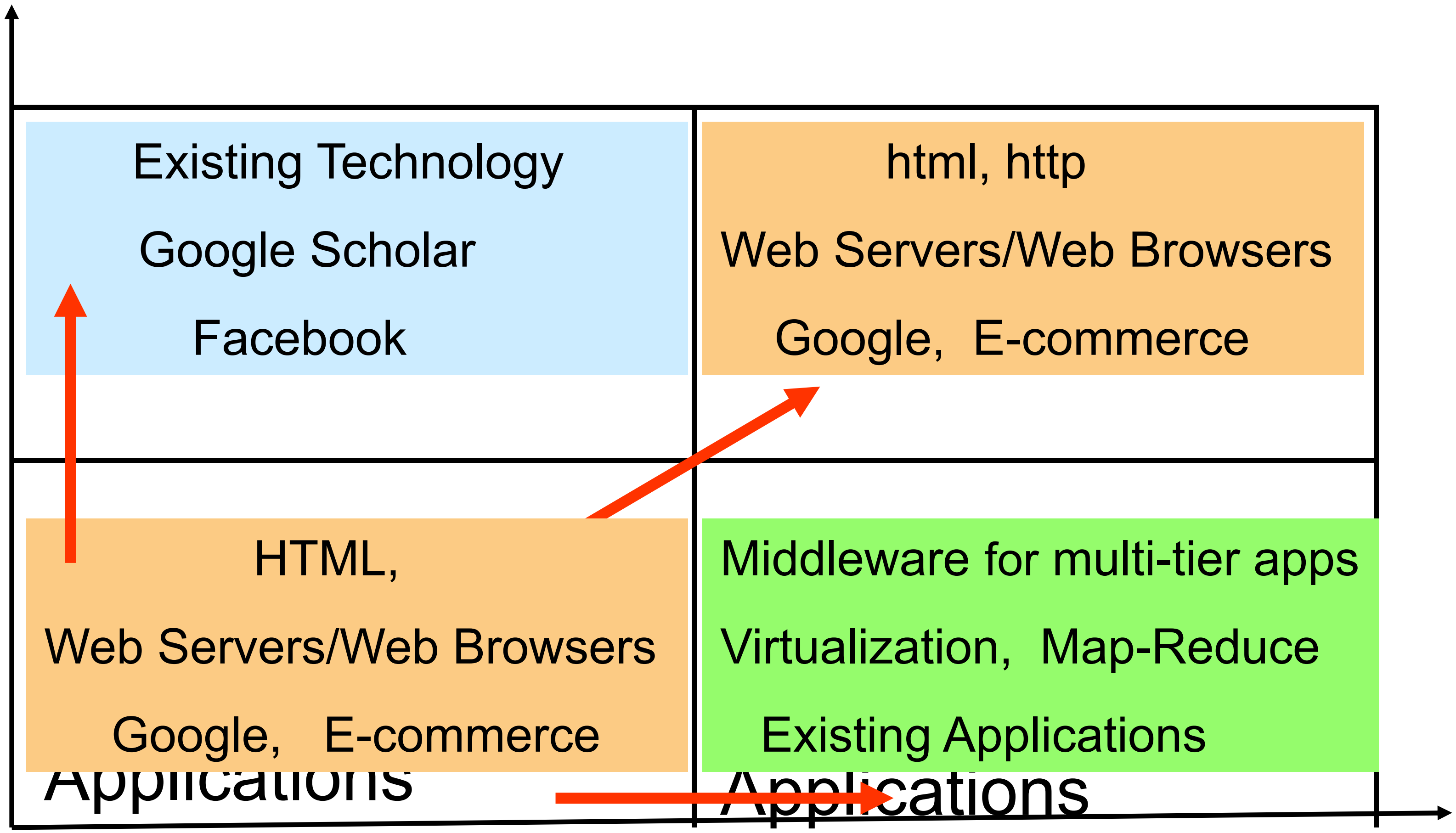
MapReduce

- A simple programming model that applies to many large-scale data/computing problems
- Automatic parallelization of computing tasks
- Load balancing
- Automated handling of machine failures
- Observation: for large enough problems, it is more about disk & network than CPU & DRAM
- Challenges:
  - Automated bottleneck analysis of parallel dataflow programs and systems
  - Where to apply optimizations efforts: network? disks per node? map function? Inter-rack data exchange?...
  - Automated model building for improving efficiency and better utilization of hardware resources

Existing and New Technologies

Existing Technology New Applications	New Technology New Applications
Existing Technology Existing Applications	New Technology Existing Applications

Existing and New Technologies



Summary and Conclusions

- Large-scale systems require new middleware support
  - memcached and MapReduce are prime examples
- Monitoring of large-scale systems is still a challenge
- Automated decision making (based on imprecise information) is an open problem
- Do not underestimate the "role of a person" in the automated solution
  - "It is impossible to make anything foolproof because fools are so ingenious" -- Arthur Bloch

Thank you!

Questions?