Software Systems Research – Portfolio Review

Dr. Nelson R. Manohar Alers

nelsonmanohar@yahoo.com
nelsonmanohar.sphosting.com/research

Outline of the Talk – Update

- **Background [10%]**
- **Computer-Supported Collaboration [25%]**
- **Dynamically Customized Web Touring [25%]**
- **Multimedia Computing Networking [35%]**
  - Building Robust Network Performance Indicator
  - **Distributed Resource Management for Multimedia**
- **Wrap-Up [5%]**

Distributed Resource Management

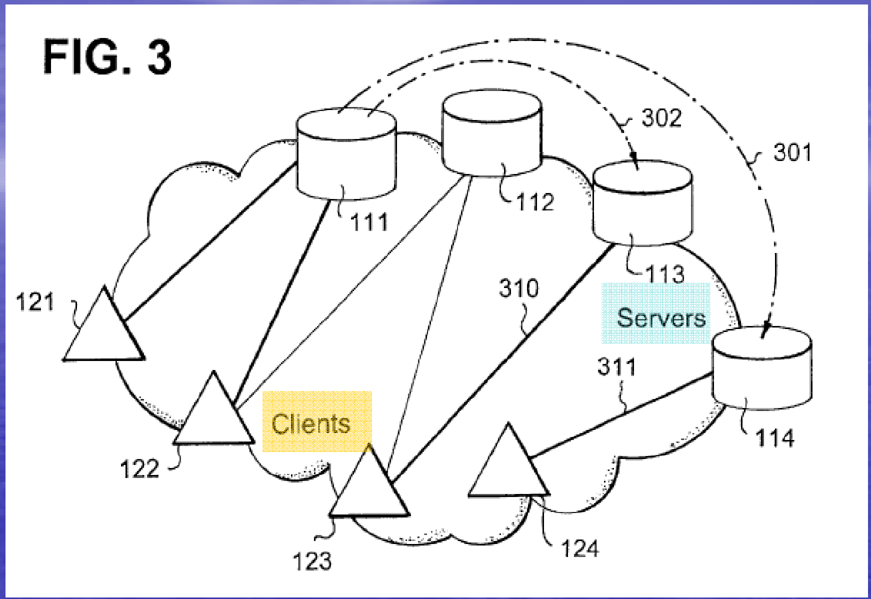
Work at IBM T. J. Watson Research Center

By N. R. Manohar, L. Lumelsky, and S. Wood

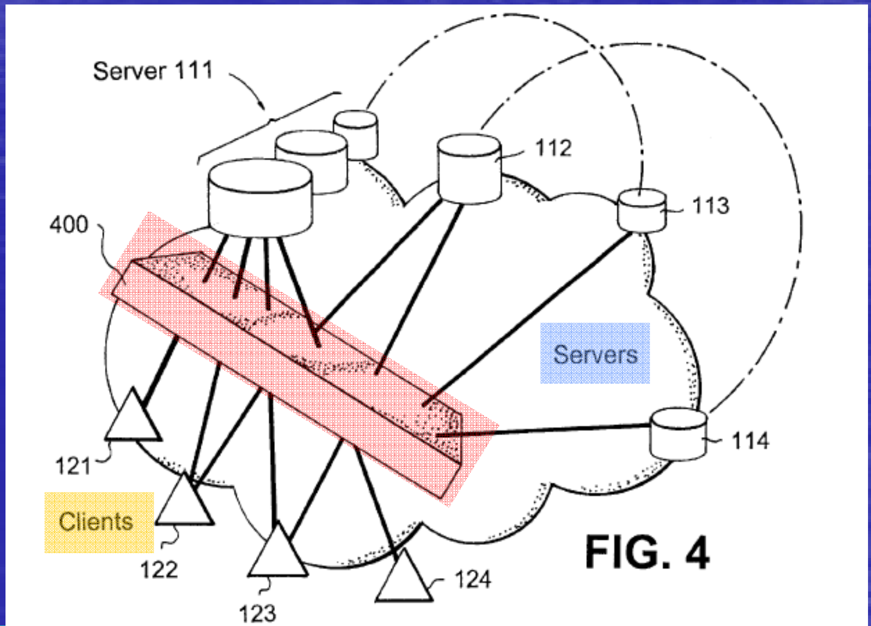
U.S. Patents: *6,516,350; 6,463,454; 6,529,950; 6,377,996; 6,460,082; 6,466,980;*

Distributed Resource Management Plane

- **<u>From</u>: Ad-Hoc Resource Management**
  - ad-hoc client-services
    - streaming, metering, etc.
  - ad-hoc server-services
    - caching, load-balancing, etc.



- **<u>To</u>: Resource Management Plane**
  - distributed management plane
  - brokering of clients to servers
  - standard plane server-services
    - caching, replication, load-balancing, dynamic hosting, etc.
  - compliant plug-in servers



Motivation and Goals

- **Evaluation**
  - Multimedia Computing Networking at Internet2-like networking level
    - very high performance Backbone (vBNS 2.4Gbps OC-12)
    - differentiated class services, RSVP, RTCP, RTP, etc.
- **What could benefit from this?**
  - large multimedia objects under a value-asset model
  - e.g., movies
- **What could we do now?**
  - distributed resource management

Very Large Multimedia Objects?

- **internet-perspective**
  - flow-oriented, end-to-end, distributed QoS/reservation problem
  - non-negligible, considerable, replication cost
- **web-perspective**
  - extremely large in comparison to dominant web objects
  - request characterization of web-servers [MSNBC00]
    - some objects significantly more requested than others
    - small set of objects accounts for majority of requests (Zipf relative frequencies)
  - spatial and temporal locality on requests [IBM97]
    - movie is blockbuster, demographic and temporal consequences
  - and then, some are exceptions, i.e., statistical outliers
    - for example, fads (30s movies, some-actor movies, etc.)
- **provisioning-perspective**
  - value-asset model, digital rights model
  - non-cacheable

Distributed Resource Management?

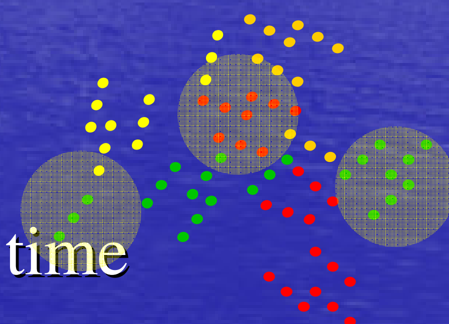
- **What else can we do there? – again, viewpoint matters**

- **adapting demand to the network (variable client demand, fixed server capacities)**
  - balance client demand presented to the network *wrt* fixed server capacities found across the network
  - traditional load balancing, i.e., demand shaping
  - that is, “demand-follows-capacity” policy

- **adapting the network to demand (variable client demand, variable server capacity)**
  - regulate the allocation of server capacities across the network *wrt* client demand presented to the network
  - the other side of the coin, i.e., distributed capacity shaping
  - that is, “capacity-follows-demand” policy

Capacity Shaping – Basic Idea

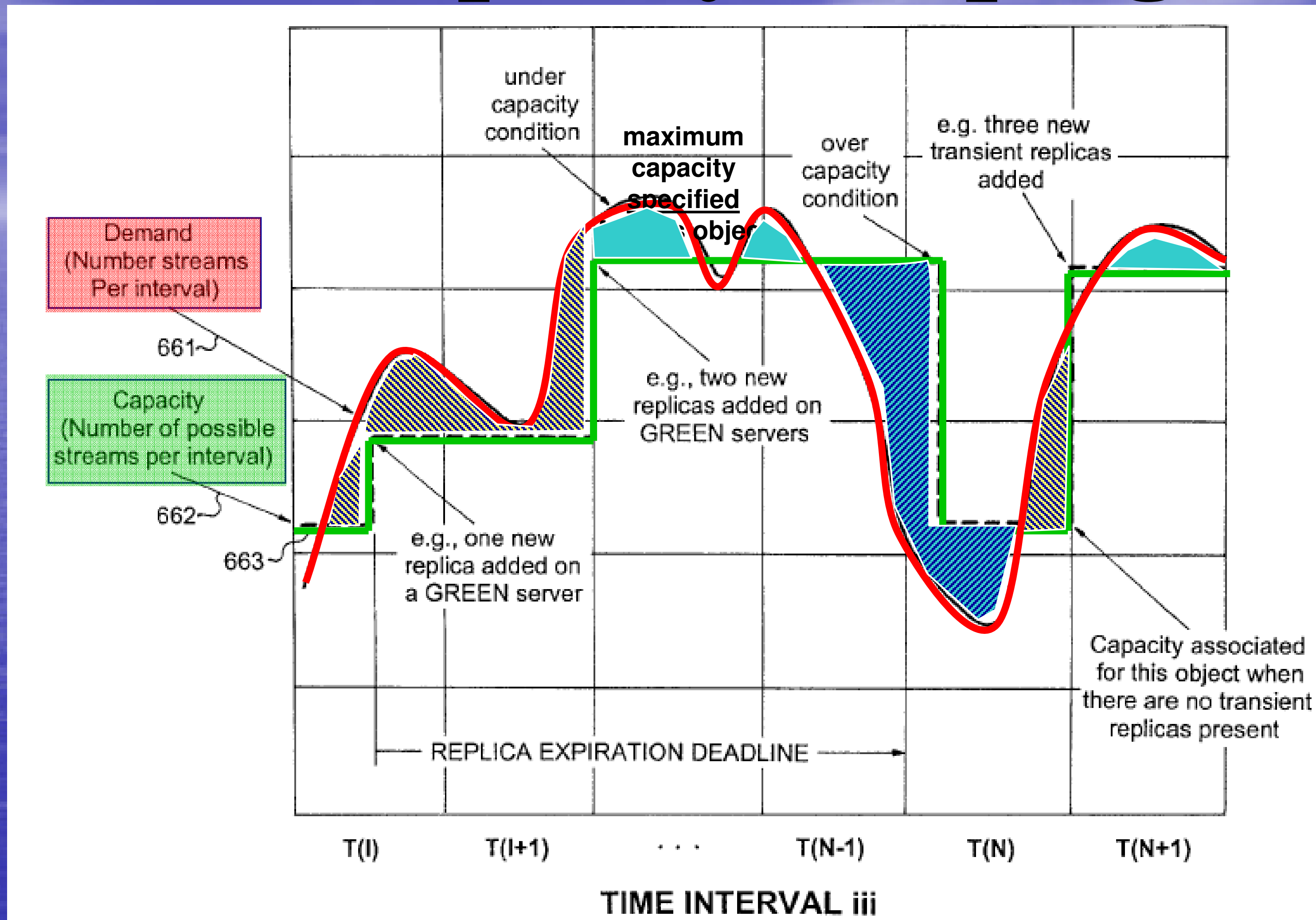
- **capacity shaping mechanism**
  - object is associated with object replicas
  - object replica specifies (serving) resource requirements (i.e., normalized capacity)
  - allocation of an object replica commits the replica's resource allocation (i.e., normalized capacity allocation)
- **distributed dynamic capacity shaping**
  - expiration time associated with object replica
  - server placement of object replicas varies over time
  - total number of object replicas varies over time



- **thus, allocated object-serving capacity placed over the network shaped over time**
  - done for selected objects – referred to as hot-objects

Self-Regulated Capacity Shaping

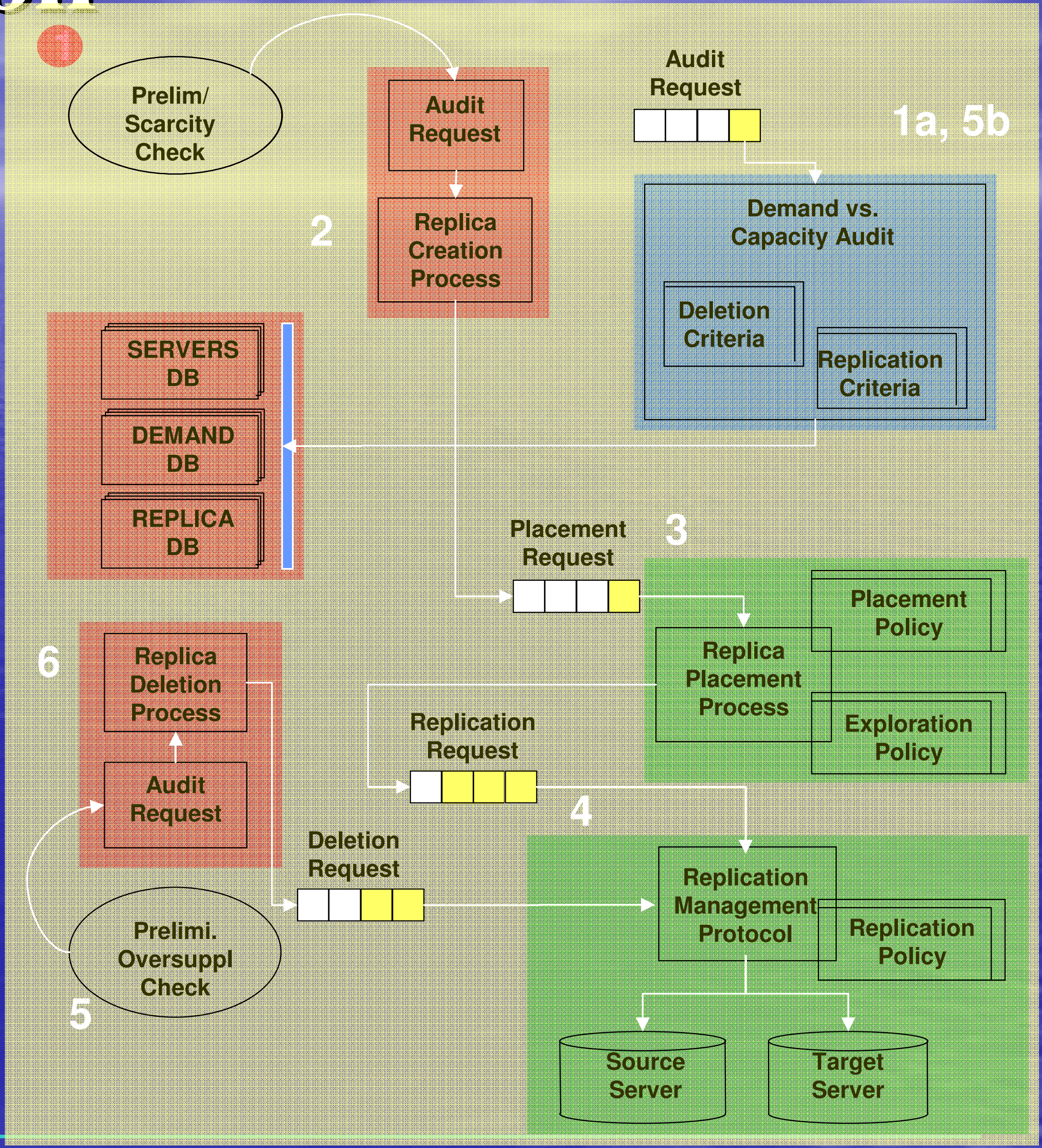
- load balancing and capacity shaping are complementary
- traditionally, throttle control (against fixed capacity) for demand-shaping
- now, self-regulation (against variable capacity) for capacity shaping
- building block: robust state signaling



- dynamic capacity shaping for which objects?
  - Zipf relative frequency distribution – hot objects
- then, how is self-regulated capacity shaping implemented?

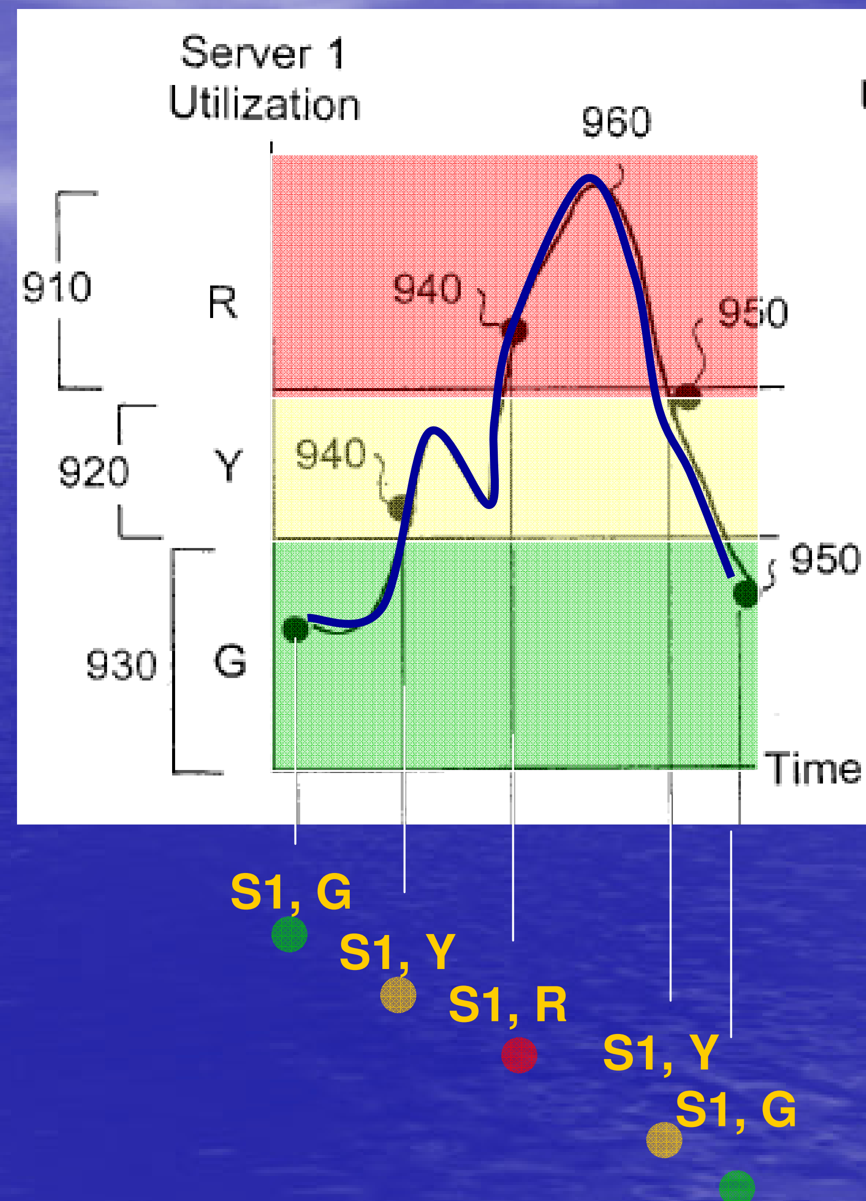
Capacity Shaping Implementation

1. scarcity check
   - **demand/capacity analysis (a)**
2. replica creation
3. replica placement
4. replication
5. oversupply check
   - **demand/capacity analysis (b)**
6. replica deletion



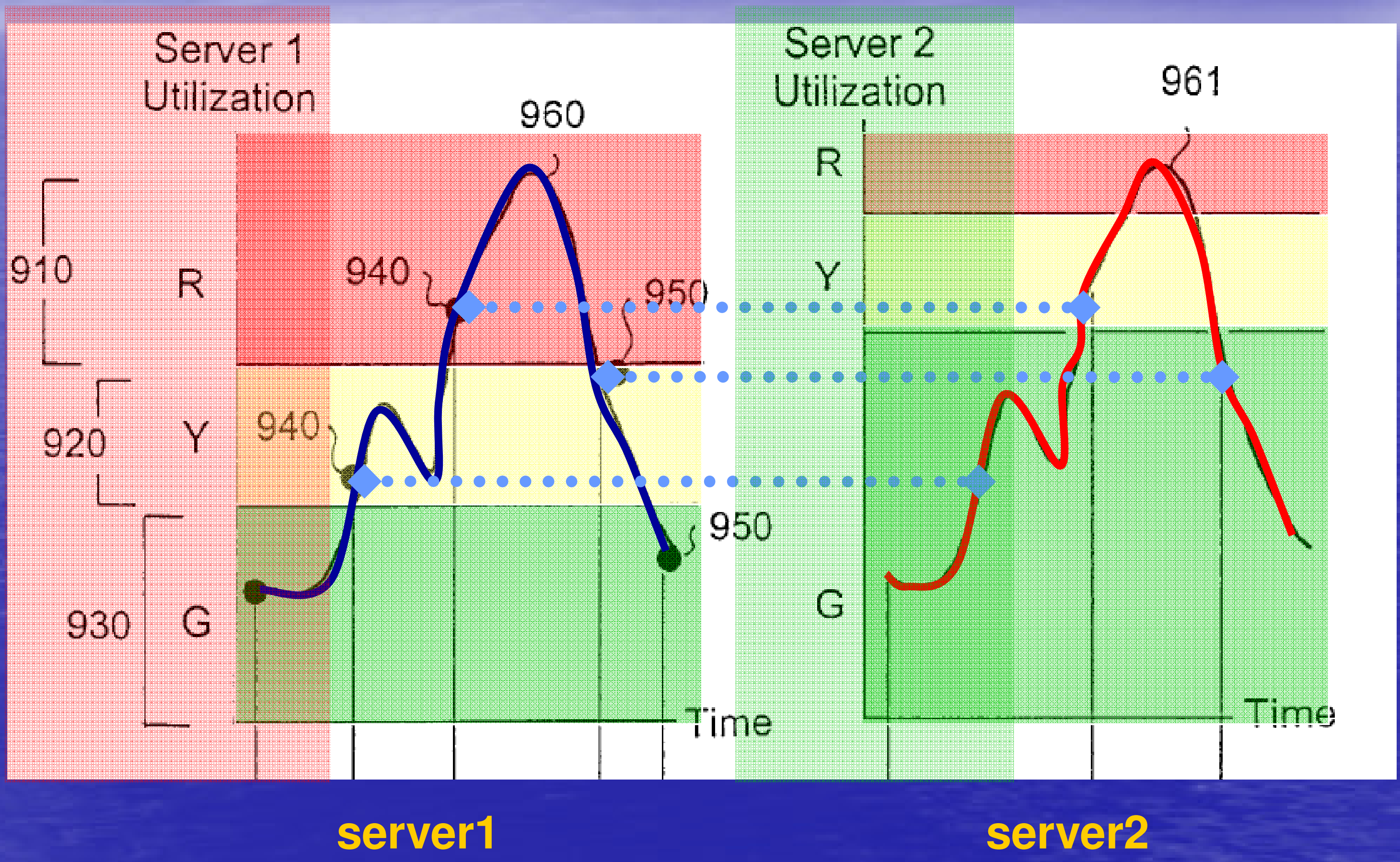
Management of Remote Capacities

- **low overhead on tracking of distributed state**
  - server capacity is normalized
  - semaphore-like
  - green, red, yellow regions
  - trigger-management problem
- **then, to handle transient changes**
  - regions act as buffering regions
  - green
    - safe operating region
    - low management overhead region
  - red
    - critical spare (X%) region
    - non-allocable (safe operating margin)
  - yellow
    - trigger region <u>and</u> buffering region
    - buffers transient state changes and messaging delays



Management of Heterogeneous Servers

- to handle server heterogeneity
  - regions are server specified
- management goal becomes about
  - handling of red servers
- where
  - forewarning through state signaling
  - capacity spare used to manage signaling



Resource Management at Plane

- Low-overhead State Tracking
  - quantized demand
  - normalized state & capacity
- Object to Demand
  - demand volume
  - object-demand rating
  - interval-tracking
- Object to Replica
  - server placement
  - server capacity
  - expiration time
- Server to Capacity
  - capacity rating
  - utilization state
  - globality

ObjectID	Demand rate req/s	Volume $t_{(i-1)}$ req	Volume $t_{(i-2)}$ req	Hot Object	Time Stamp
420	10	120	60	yes	t_i
425	5	60	55	no	t_i
428	5	30	62	no	t_0

object 420: hot, large demand

Object_ID	Replica	Server	Transient Replica	Time-to-Live
420	421	1211	NO	
	422	1221	YES	060599-133000
440	441	1211	NO	

transient replica for 420 at server 1221

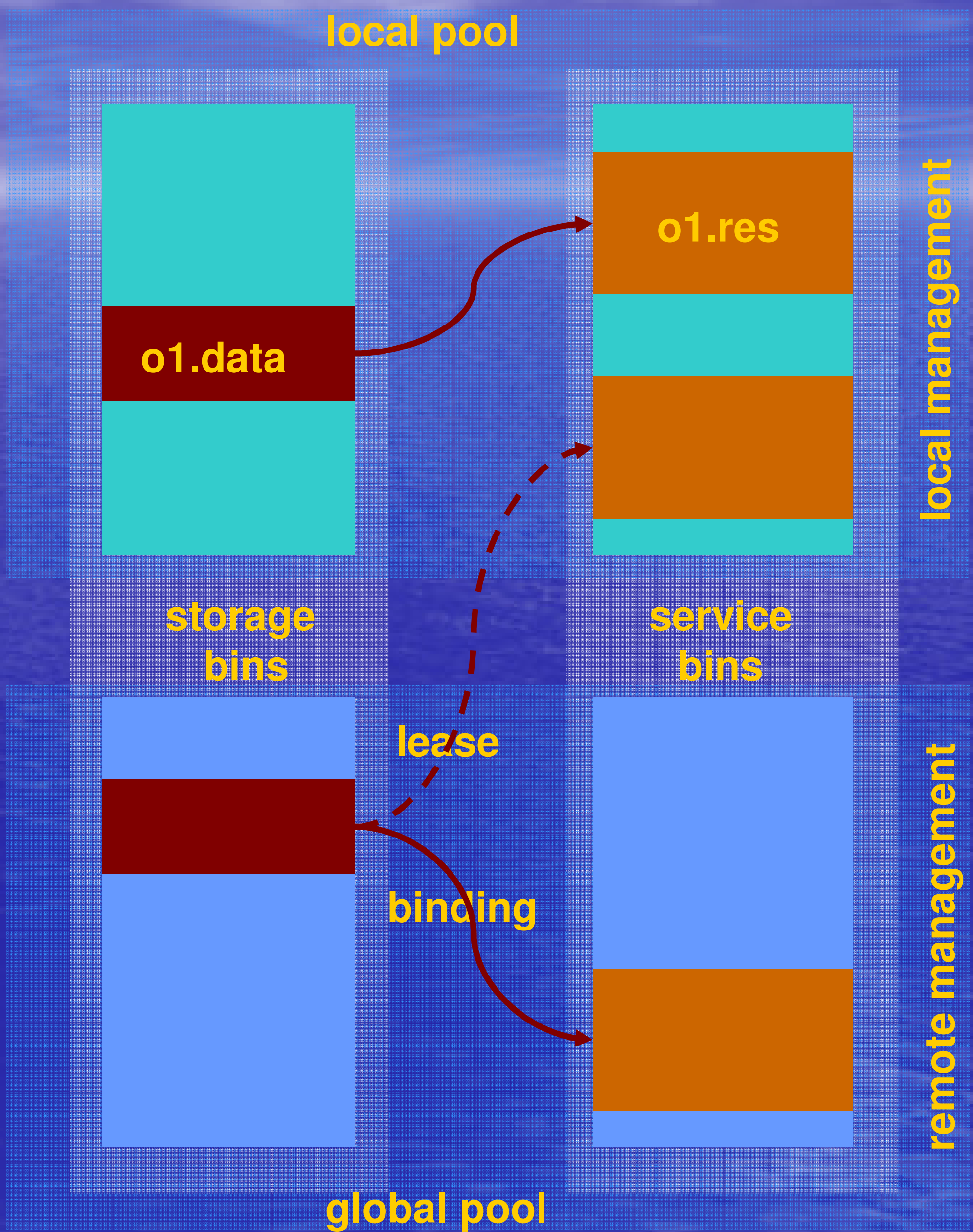
Server	IP Address	Capacity Rating	Utilization State	Timestamp	Globality
1211	209.09.9.127	Low	Red	t1	local
1221	128.0.0.1	High	Green	t2	global

global server 1221: high capacity, low use

Resource Management at Server

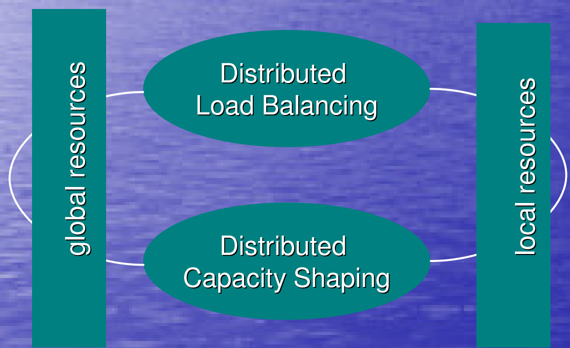
- **reservation bins abstraction**
  - normalized management of replication and provisioning problems
  - storage-bins – for replica placement problem
  - service-bin – for resource reservation problem

- **reservation pools abstraction**
  - for <u>local resource management</u>
    - local serving resources
  - for <u>global resource management</u>
    - *degree-of-freedom* resource for the distributed capacity-shaping problem

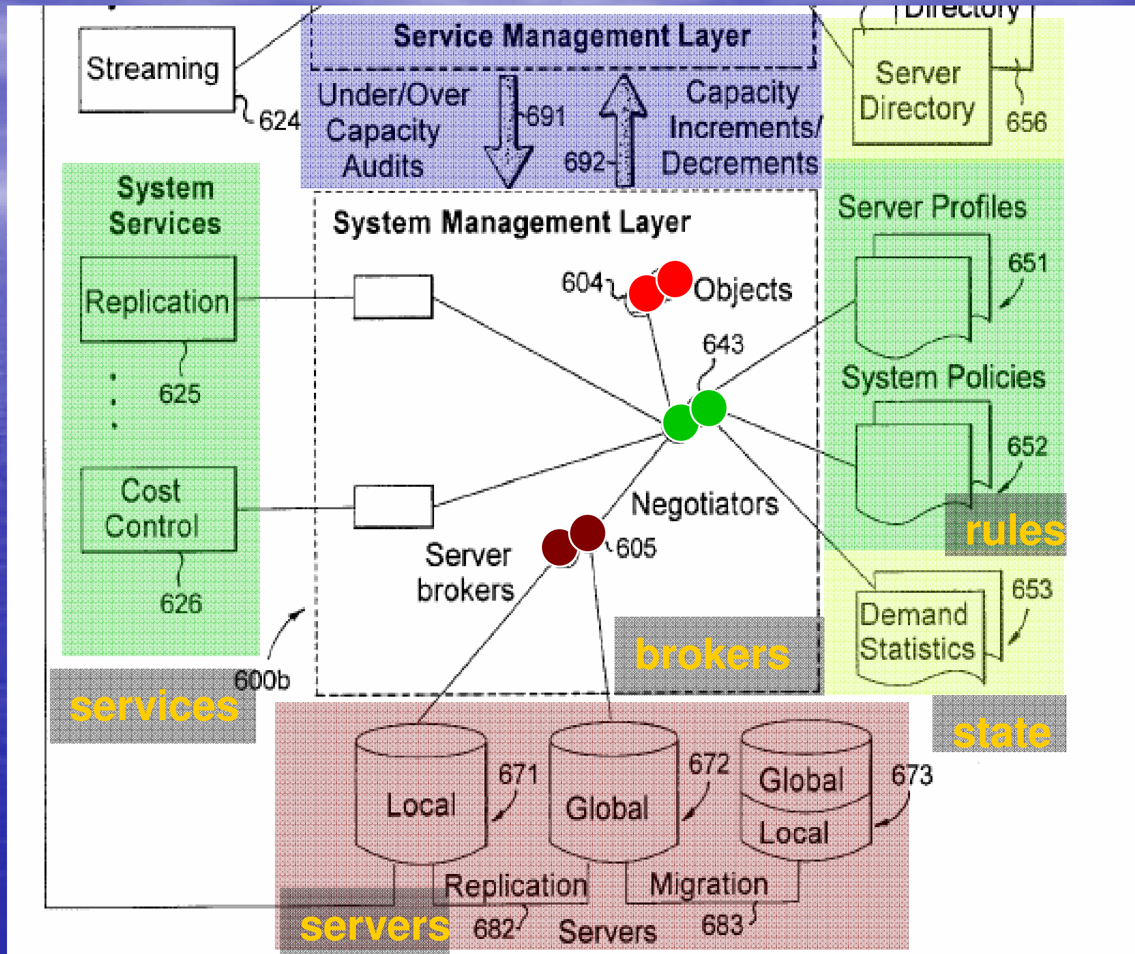


Capacity Shaping Management Layer

- capacity shaping plane brokers object replication between servers
  - dynamically creating capacity for load balancing

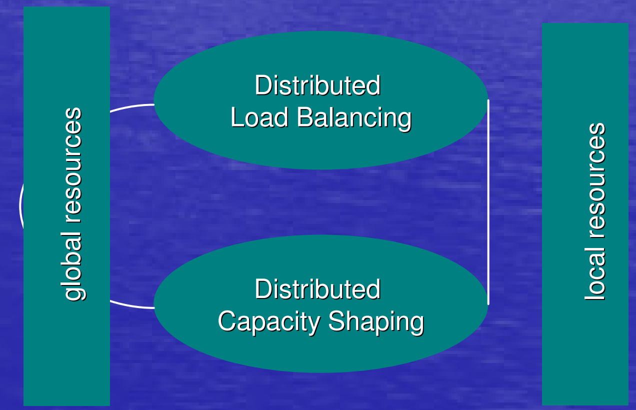


- driven by service management layer
  - by demand-capacity analysis resulting in dynamic replica hosting at brokered servers



Contributions

- **Distributed Dynamic Capacity Shaping**
  - for distributed resource management of large (multimedia) objects over high-bandwidth internet
  - self-regulated distributed capacity shaping
    - integrated load-balancing and dynamic capacity shaping
    - management of global and local replicas *wrt* server capacities
  - distributed state management
    - red-server avoidance management
    - low-overhead distributed state tracking
    - stability-oriented trigger signaling
    - tradeoff of local capacity to remote state tracking
- **complementary to existing technologies**
  - multicast, rights, RSVP, grids, etc.
  - but levering high bandwidth internet (e.g., internet2)



Related Work

- **Dynamic Capacity Virtual Hosting**
  - server farms (e.g., IBM Global Services) – demand-shaping of capacity at centralized center(s) vs. capacity-shaping of demand through placement of distributed capacities
- **Distributed Programmable Planes**
  - grid computing (e.g., Globus) – resource management for entirely different problem, small number of large (computations and datasets) objects with low viewership vs. large number of large objects with high and wide viewership
- **Distributed Replica Management**
  - edge-caching network (e.g., Akamai) – on-demand caching of relatively small objects vs. valuable asset very large object
- **Multicast Streaming**
  - multicast is complementary technology – plane represents the self-regulated placement of the sources of the multicast trees
- **Brokered Distributed QoS Architectures**
  - distributed QoS brokering is complementary technology – delivering end-to-end QoS feasibility used for brokering of clients to servers