Announcements

Bayou Write

3-tuple: <update><dependency check><mergeproc>

For example,

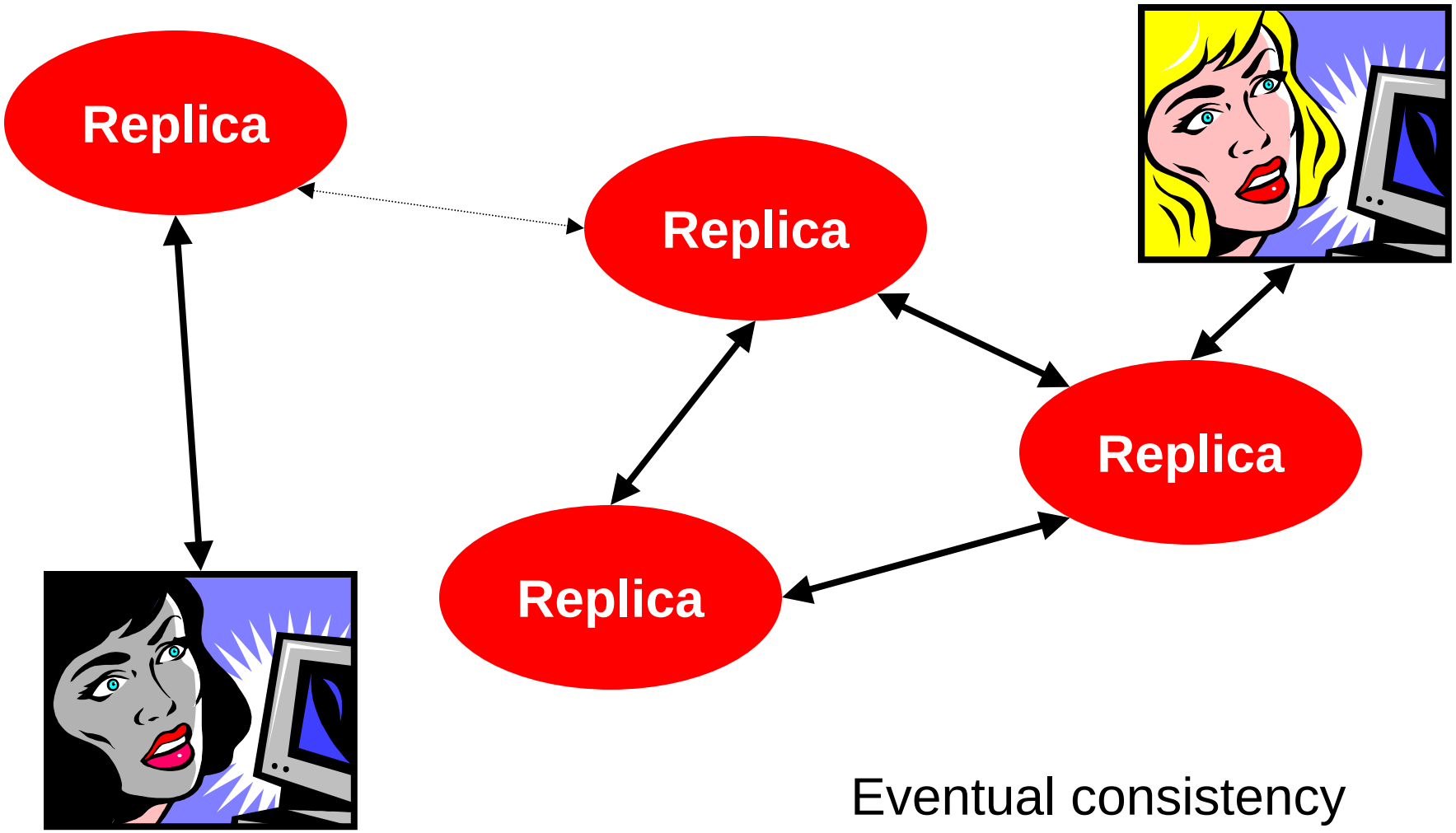
Update: <insert, Chem453, 2/7/2001, 11:15am, 50min, “Ubiquitous Computing”>

Dependency check: <Is there an entry in Chem453 database on 2/7/2001 at 11:15am for 50 min? expected answer is empty>

Mergeproc: <Try 2/7/2001 at 12:15pm for 50min; if successful write that tuple>

- sometimes users like conflicts

Pair-wire reconciliation



Eventual consistency

Global commit order assigned by Primary server

Replica management

- New replicas are cloned from existing replicas
- Information about new replicas propagate with normal writes (“creation write”)
- Information about retired replicas also propagate with normal writes (“retirement write”)
- Replicas are created by users or sys. Admins.
- Replication schedule is controlled by users
- Sometimes users want to choose specific replicas
- Tentative and committed data
  - Applications can choose to read tentative or committed data

Technology impact

- TrueSync - end-to-end synchronization software and infrastructure solutions for the wireless Internet
  - http://www.starfish.com/

- SyncML - SyncML is the common language for synchronizing all devices and applications over any network.
  - Ericsson, IBM, Lotus, Motorola, Nokia, Palm Inc., Psion, Starfish Software etc. (614 companies)
  - http://www.syncml.org/

Discussion

Outline

- *Epidemic Algorithms for replicated database maintenance* Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. In Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing

*Involved in bayou

Epidemic algorithms

- Randomized algorithms for maintaining consistency for updates to replicas
- Precursor to Bayou and other systems
- Algorithms modeled after epidemics (diseases are spread by infecting the next victim)
- One algorithm implemented in Xerox clearing house system

Xerox Clearinghouse servers

- The Clearinghouse is a decentralized set of processes that provides an efficient but not terribly robust method for a distributed name service (late ’70s)
- They used mail and anti-entropy as the mechanism to distribute updates between replicas
- Within domain - Anti-entropy. On failure direct mail
- In related work they mention this DARPA domain system (we call it DNS now)

Epidemic terminology

- Site holding an update it is willing to share “infective”
- Site that has not received an update “susceptible”
- Site that has received an update but not willing to share it “removed”
  - Anti-entropy: sites are always susceptible or infective

Considerations

- Time required for an update to propagate to all sites

- Network traffic generated in propagating a single update. Ideally traffic is proportional to the number of nodes, but some algorithms create much more traffic

Direct Mail

- Each update is immediately sent from its entry site to all other sites.
- When a node receives an update, it checks the timestamp of update with local timestamp. Newer updates win
  - Timely – updates are sent immediately
  - Efficiency – reasonable. Number of messages proportional to number of updates and average hop count
  - Problems:
    - Nodes do not know about all replicas
    - Mail is not reliable delivery mechanism

Anti-entropy

- Entropy - amount of entropy is a measure of the disorder, or randomness, of a system. (from thermodynamics – Encyclopedia Britannica)
- Updates available in few sites – high entropy. Anti-entropy tries to restore order back into the system
- Every site regularly chooses another side at random and exchanges database contents with it and resolves any different between the two

Anti-entropy

- Differences are resolved using:
  - Push: infective -> susceptible
  - Pull: susceptible -> infective
  - Push-Pull: depending on the time stamps, updates are either pushed or pulled
- Common case: Pull or push-pull preferred
- Reliable, but high overhead because have to “diff” the databases

Rumor mongering

- Sites are initially “ignorant”
- When site receives new information, it becomes a “hot rumor”
  - Periodically chooses another site at random and ensures that the other site has seen the update
  - When a site has tried to share a hot rumor with too many sites that have already seen it, the site stops treating the rumor as hot and retains the update without propagating it further
  - 1/k probability : k=1, 20% and k=2, 6% will miss updates
  - There is a chance that an update will not reach all sites (backup anti-entropy process)

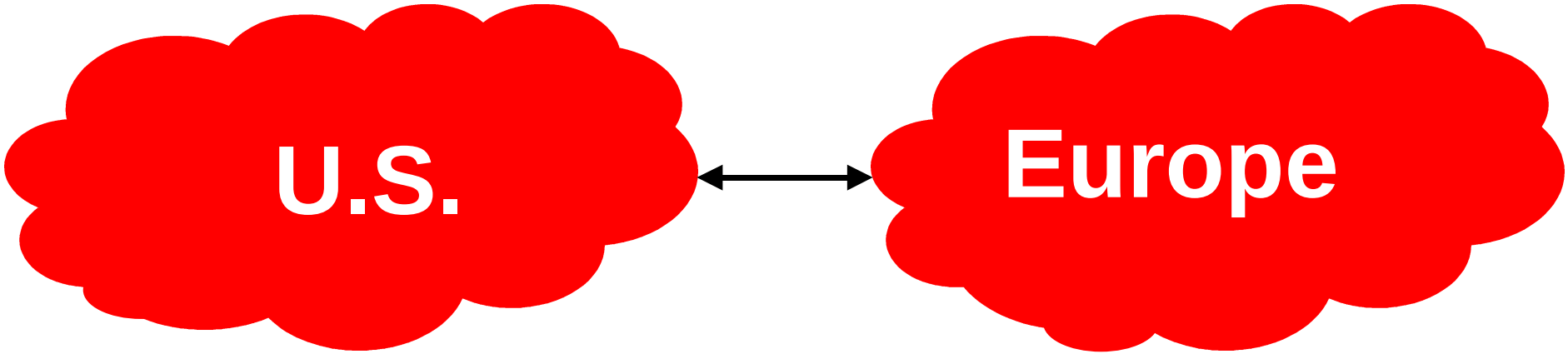
Complex epidemic variations

- Blind – 1/k probability of losing interest regardless if recipient is susceptible
- Feedback – 1/k probability only if recipient is infective
- Counter – lose interest after k unnecessary contacts
- Coin – k cycles regardless if susceptible
- Push and Pull
- Minimization – counters on both ends
- Connection limit – limits the number of connections
- Hunting – if a connection is rejected, choosing site can hunt for alternate sites

Deletion and death certificates

- When we delete an item, we insert a death certificate so that the data is deleted in other replicas (rather than filled with older data values)
- How do we make sure that these death certificates are deleted?
  - Make sure that all nodes have seen the death certificates
    - What is a node crashes in the middle. Have to make sure that node deletions propagate before death certificates
  - Fixed time interval
- Dormant Death Certificates
- Anti-entropy with dormant death certificates
  - Activation timestamp
- Rumor mongering with dormant death certificates

Spatial distribution and anti-entropy



- The critical link can become the hot-spot for anti-entropy and rumor mongering algorithms

Discussion