Grapevine

Robert Grimm
New York University

The Three Questions

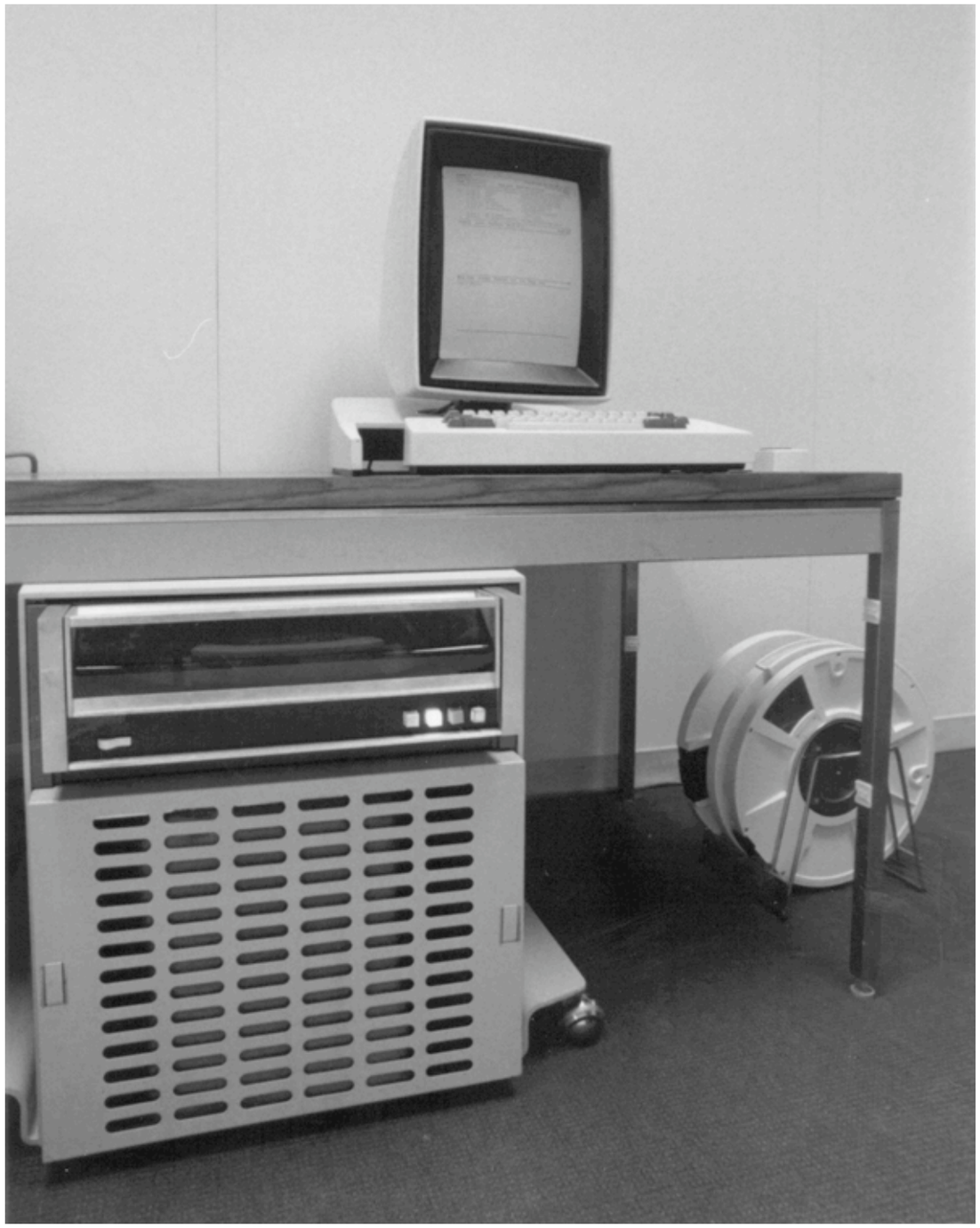
- What is the problem?
- What is new or different?
- What are the contributions and limitations?

Historical Context

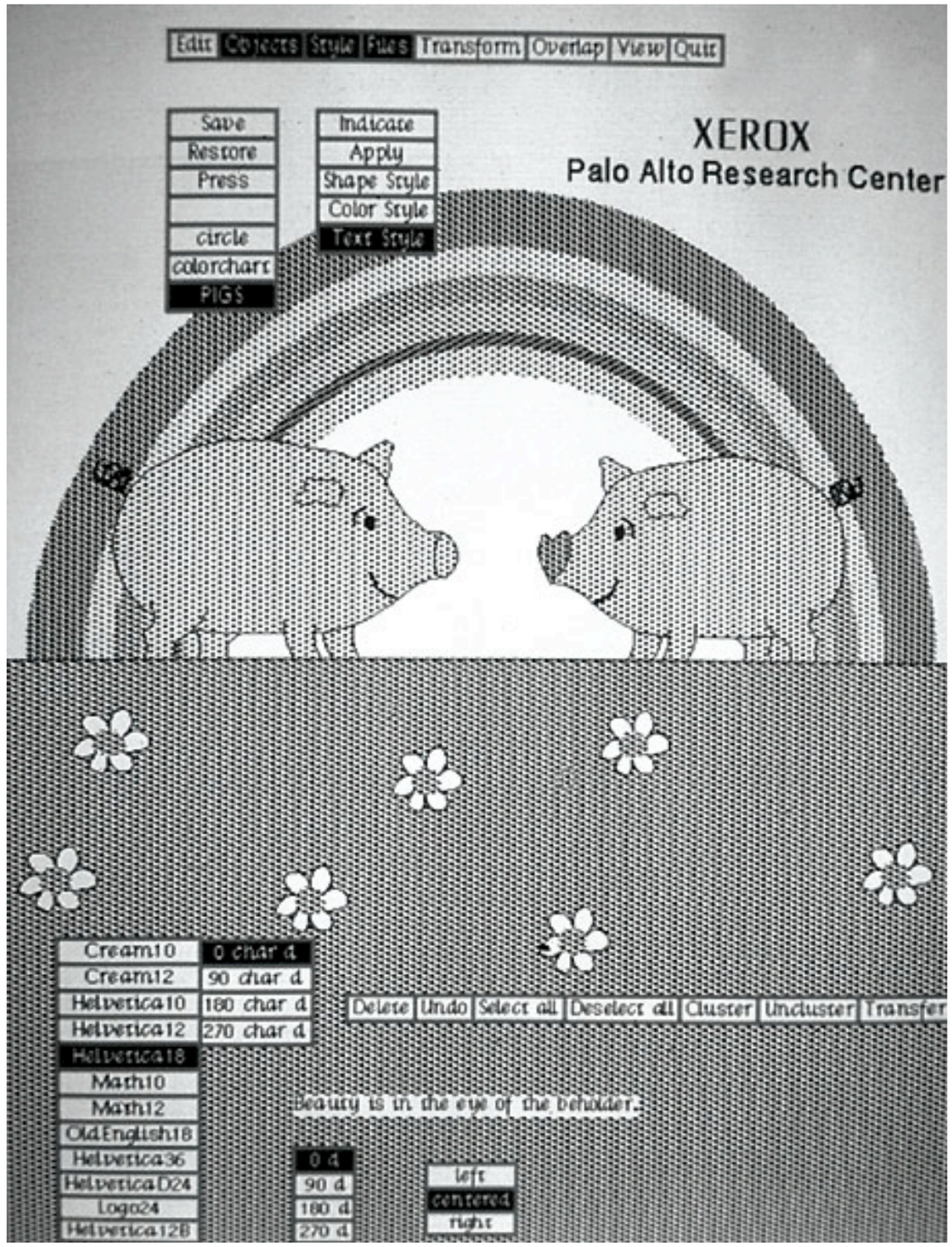
- Grapevine in everyday use: Fall 1981
- SMTP RFC: August 1982
- POP2 RFCs: October 1984, February 1985
- POP3 RFCs: November 1988, May 1991, June 1993
- IMAP
  - Conceived in 1986
  - First RFC in 1988
  - First and second meetings at UW in 1996
    - Sun, Netscape announce support for IMAP4
- DNS RFCs: November 1983

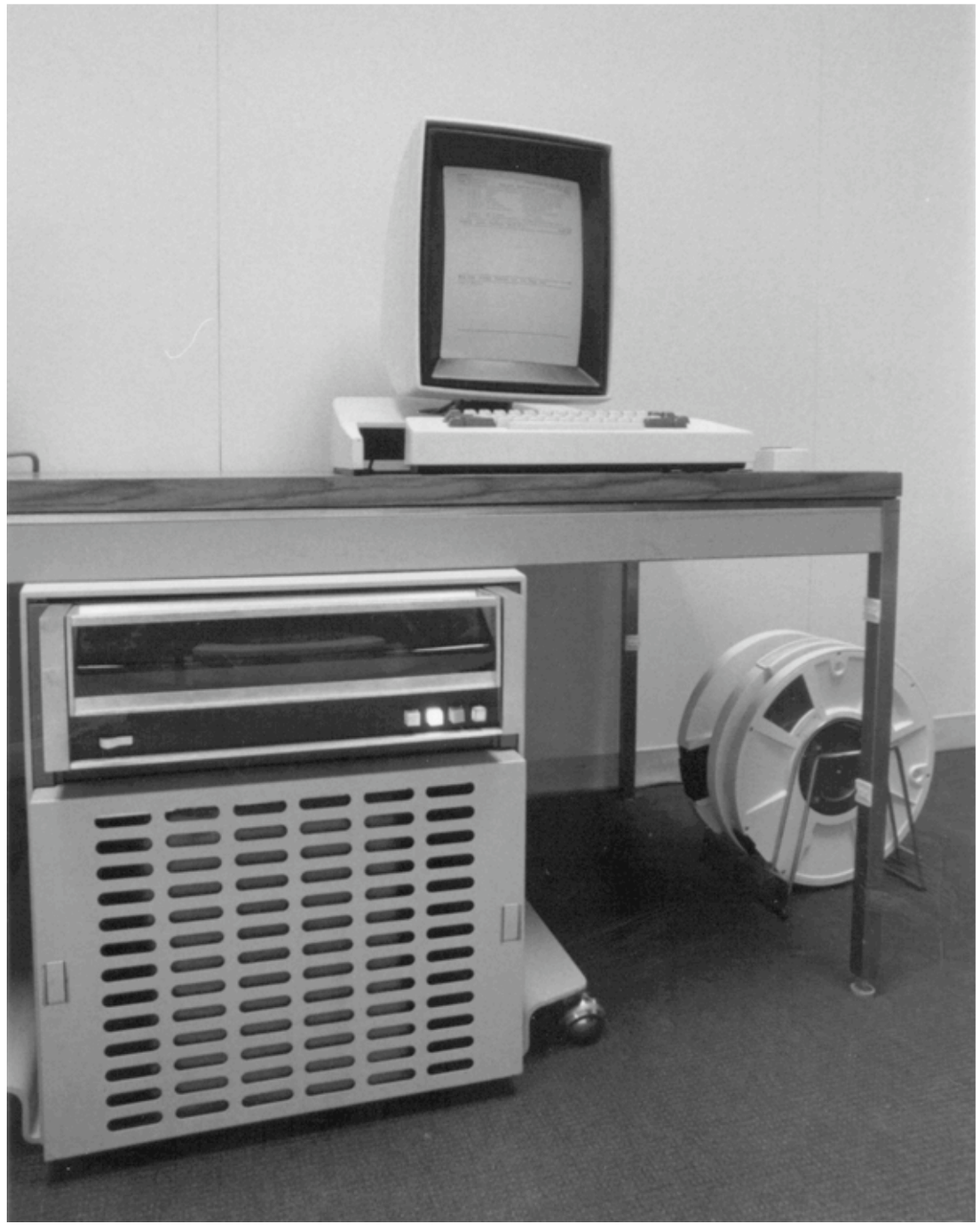
Grapevine Environment

- Xerox research internet
  - Local ethernet networks
  - Gateways
  - Long distance links
- Dedicated servers
  - Alto with 128 KB RAM and 5 MB disk
  - Programmed in Mesa
    - Loosely based on Pascal
    - Support for interfaces, coroutines, exceptions, incremental compilation



Grapevine Environment





Message Service

- Provides distributed messages
  - Sent to individuals and distribution lists
  - Buffered in inboxes
    - On two different servers per user for fault tolerance
    - Though, messages are *not* replicated, only the delivery path
  - Treated as opaque objects
    - Message content not interpreted by service
  - Retrieved onto client machines
- Relies on registration service...

Registration Service

- Provides naming, authentication, access control, and location function
- Based on database of *RNames*
  - Group entries contain RNames of members
  - User entries specify password, ordered list of inbox sites, connect site (for servers), and other information
- Bootstrapped on itself
  - Configuration information stored in Grapevine as well
    - GV registry (*.gv) replicated across all Grapevine servers
    - Each server represented as individual (connect site is location)
    - Each registry represented as group (members are servers)
      - gv.gv lists all servers

Registration Service (cont.)

- Bootstrapped on itself...
  - Configuration information stored in Grapevine as well
    - GV registry (*.gv) replicated across all Grapevine servers
      - Each server represented as individual (connect site is location)
      - Each registry represented as group (members are servers)
        - gv.gv lists all servers
- ...and on the messaging service
  - Updates propagated through messages

More Details

- Replication of registries
  - Unit of replication is a registry
  - No server hosts all registries
  - Any server hosting a registry accepts operations
- Applications rely on user-level library
  - Makes multiple *servers* look like a single *service*
- Overall system growth
  - 1981: 5 servers, 1,500 individuals, 500 groups, 2,500 messages/day (1.7 messages/day/user)
  - 1983: 17 servers, 4,400 individuals, 1,500 groups, 8,500 messages/day (1.9 messages/day/user)

A Compendium of Techniques

- Divide and conquer
  - Partition state into distinct registries
    - What limits each registry's size?
- Replicate delivery path
  - But do not replicate actual emails — why?
- Replicate registries
  - But provide only *eventual consistency* — why?
- Provide functional homogeneity
  - Every server runs the message and registry services
    - Though not all state is available on all servers

Experiences

- Effects of scale
- Configuration decisions
- Transparency
- Adjusting to load
- Operational concerns
- Reliability

Effects of Scale

- Design target: 30 servers managing 10,000 users
  - But what happens if the system grows larger?
- Global state might become limiting factor
  - Space: 15 KB for the GV registry, <1% of disk space
  - Time: Locating closest server out of 30 is acceptable
- Manual partitioning has only been partially effective
  - Distribution lists grow with population
    - E.g., Tax^.pa has 500 members, which need to be resolved by accepting server (including locating inbox for each user)
    - Suggested solution: Distribute load through layer of indirection
      - Tax^.all, which breaks down into per-registry lists

Effects of Scale (cont.)

- Overall message volume is a concern
  - Physical world: there's only so much paper that can be pushed around
  - Electronic world: need a better filtering mechanism
- Large number of unreliable links is a concern
  - Need store & forward architecture instead of direct delivery
- How have these concerns played out for email?
  - SMTP? Projected 2.7 trillion messages for U.S. in 2007
  - Spam? 58% of all email in May 2006

Configuration Decisions

- Organizational structure for email inboxes
  - Encourages sharing of data
    - Same email stored only once
      - In 4.7 inboxes on average, maximally 300
    - Has natural scalability limits
      - There's only so many people that can collaborate effectively
- Geographical structure for registries
  - Relatively stable within commercial organization
    - Unlike the organizational hierarchy...
  - But can be an arbitrary criterion in the real world
    - One group split evenly between El Segundo and Palo Alto

More Configuration Decisions

- Location of registry replicas
  - Close to inboxes for that registry
  - Close to servers accepting messages to distribution lists
  - On both sides of unreliable links
  - On enough machines to avoid catastrophic losses
    - Constraints 2-4 lead to three replicas
  - Not on busy servers

- Ph, that's an awful lot to consider for an administrator

Transparency

- Works well in common case but leads to surprising results in some uncommon cases
  - Propagation delay in registry updates
    - Client library may pick different server for next operation
  - Expensive consequences of simple operations
    - Changing inbox list originally caused remailing of entire inbox
  - No notion of distance of server
    - Long delays when using nonlocal server
  - Little information on overall state of the system
    - Unused distribution lists, inaccessible servers, duplicate msg's

Adjusting to Load

- Naïve algorithm kills your performance
  - Sending whole object updates instead of deltas/operations
    - Adding/removing *one* member to/from list is frequent
  - Not distinguishing between users and groups
    - Need to look up every single name for access control
    - What about flattened groups?
- Assuming that all mailbox access is sequential
  - Users move about and leave their mail on the server

Operational Concerns

- GV experts are few; operators need not be qualified
  - Operators reboot servers, load programs, fix hardware
- SSH is your friend
  - Well, at the time: remote disk editor, viticulturist's entrance
    - "a cultivator of grape vine"
- Logs are your friend
  - Provide more than a week of history
  - But need to be combined from different servers
    - Unique identifiers help
  - Can be viewed dynamically
    - Check fixes, notice oddities
- The "dead letter facility" is a good idea?

Reliability

- System requires spare capacity to work
  - Detecting resource depletion early is crucial
  - But system still needs to work without resources
- Servers being treated as equals can be a burden
  - Circuit manufacturing facility builds on Grapevine
  - But can become backup when rest of system overloaded
- Message server depends on email archive
  - Helps reduce storage space, but?
  - Also, archiving is triggered by what?

One Final Note

- "This reluctance is partly due to the potential disruption that introduced bugs would have on the large user community that depends on Grapevine services to get its work done."
- What happened here?

What Do You Think?