Web-Based Management of IP Networks and Systems

Bell Laboratories, Lucent Technologies, Murray Hill, NJ, USA

March 15, 2000

Jean-Philippe Martin-Flatin
Swiss Federal Institute of Technology, Lausanne (EPFL)
Institute for computer Communications and Applications (ICA)
jp.martin-flatin@ieee.org
http://icawww.epfl.ch/~jpmf/

Outline

- Problems with SNMP-based mgmt
- Web-based mgmt
- Push model
- New communication model
- XML
- JAMAP: research prototype
- Conclusion

IP Management Platforms: Mandatory Tasks

- Monitoring:
  - detect faults in network devices, network links, and systems:
    - ➠ reactive w.r.t. faults
    - ➠ proactive w.r.t. short-term complaints from users
- Data collection:
  - gather data to build daily, weekly, and monthly reports:
    - ➠ proactive w.r.t. long-term complaints from users
- Notification handling:
  - pseudo real-time
  - react to events generated by agents (SNMP notifications)
  - react to events generated by the manager (rule-based data interpreter)
- Configuration mgmt: (simple and ignored)

Regular Management

- Ongoing monitoring and data collection
- Automated
- 2 modes:
  - attended mode: operators gazing at GUIs (red-icon angst)
  - unattended mode:
    - automated correlation
    - alarms trigger pager, email, telephone, siren, etc.
- Midsize and large networks

Ad Hoc Management

- Troubleshooting, configuration mgmt, and temporary monitoring
- Not automated
- Single mode: attended (administrators or operators)
- All networks
- Replaces regular mgmt in small networks

Problems with SNMP-Based Mgmt Platforms (1/2)

- For customers:
  - too expensive (hardware and software):
    - ➠ dedicated hardware for network mgmt
  - limited support for third-party RDBMSs
  - insufficient integration
- For equipment vendors:
  - the support for device-specific mgmt GUIs is too expensive:
    - ➠ many mgmt platforms
    - ➠ many operating systems
    - ➠ many GUIs

Problems with SNMP-Based Mgmt Platforms (2/2)

- For customers and equipment vendors:
  - poor time-to-market for mgmt GUIs:
    - large vendors: several months after hardware release
    - startups: never --> need separate mgmt platform --> no integration
  - MIB versioning:
    - MIB upgrade in the network causes version mismatch between manager and agents:
      - manually configure the manager for each agent (no MIB-discovery protocol)
      - do not use new features of a MIB until all agents are upgraded
  - investment bound to a specific operating system

Problems with SNMP (1/2)

- SNMP expertise is domain specific --> rare and expensive
- Scalability, network overhead, and latency are adversely affected by old protocol design decisions:
  - BER encoding [Mitra 1994]
  - SNMP table retrieval mechanism (“holes”, many messages)
  - OIDs take much more space than values
  - no compression
- Low-level semantics:
  - only instrumentation MIBs
  - no standard high-level APIs
  - site-specific network applications developed from scratch:
    - ➠ bound to the API of a specific mgmt platform, not to a standard technology

Problems with SNMP (2/2)

- Security:
  - SNMPv1 and SNMPv2c: community string (simplistic)
  - SNMPv3: better, still simple, but not used
  - Next step: expensive encryption hardware (e.g., VPNs)
  - firewalls: complex and costly UDP relays [Chapman & Zwicky 1995]
- Unreliable transport protocol:
  - important SNMP notifications (unacknowledged) are lost for silly reasons (e.g., buffer overflow)
  - SNMPv3 informs (acknowledged) are not used yet
  - important mgmt data requires retransmissions at the application level
- Evolution of SNMP hampered by legacy systems:
  - “better replace than repair”

Part 1: Web-Based Management

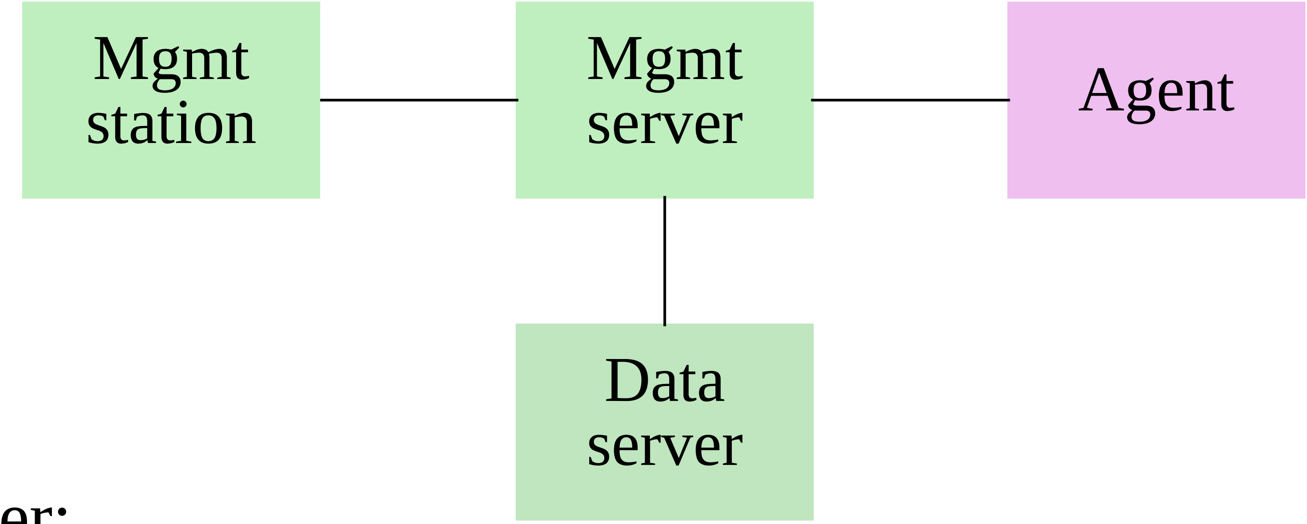
- Definition: integrated mgmt (= network, systems, application, service, and policy mgmt) based on Web technologies
- Large choice:
  - HTML forms
  - CGI (Perl scripts, Tcl/Tk scripts, shell scripts, binaries)
  - Java applets, servlets, and applications
  - Java Object Serialization
  - Java RMI (distributed objects)
  - Java IDL (CORBA)
  - JDBC (databases)
  - XML
  - ...

Why Use Web Technologies?

- Reduce development costs of mgmt GUIs (applets):
    - less expensive for customers
- Zero the time-to-market of mgmt GUIs (embedded)
- Suppress the need for separate mgmt platforms:
    - integrated mgmt
    - put small and large equipment vendors in fair competition
- Simplify mgmt of remote subsidiaries across firewalls
- Reduce network overhead (compressed mgmt data)
- Make mgmt platforms more open, more modular, and less costly
- Improve the support for 3rd-party databases

Better Design of Mgmt Platform (1/2)

- Split manager:



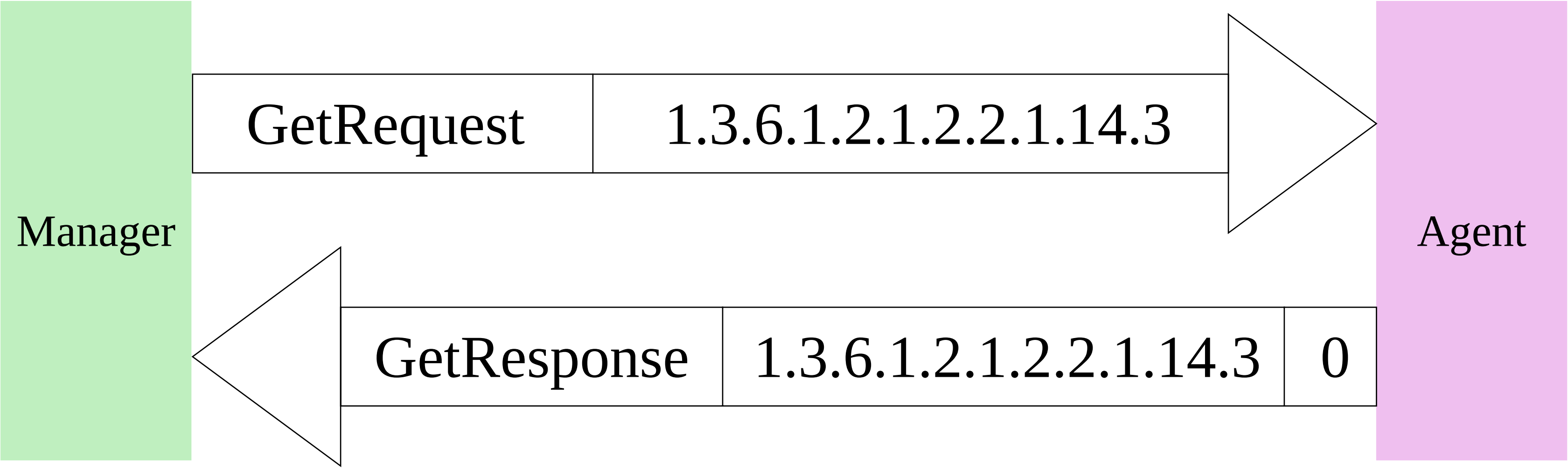
- Split mgmt server:
  - was: big, monolithic, opaque, and proprietary code
  - now:
    - integration of COTS components and OO frameworks
    - fine-grained competition between vendors (e.g., buy an event correlator):
      - less expensive
      - manager to manager: more interoperable
      - no longer enchained by big investment

Better Design of Mgmt Platform (2/2)

- Generic hooks for accessing the data server:
  - virtually all databases support JDBC or XML
  - customers are no longer dependent on peer-to-peer agreements between mgmt-platform and database vendors
  - customers need not buy a new database for integrated mgmt

Part 2: The Push Model

- Why use the push model?
  - reduce network overhead of mgmt data --> save network bandwidth
  - move some workload from the manager to the agents
  - e.g., error rate for inbound traffic through interface #3:



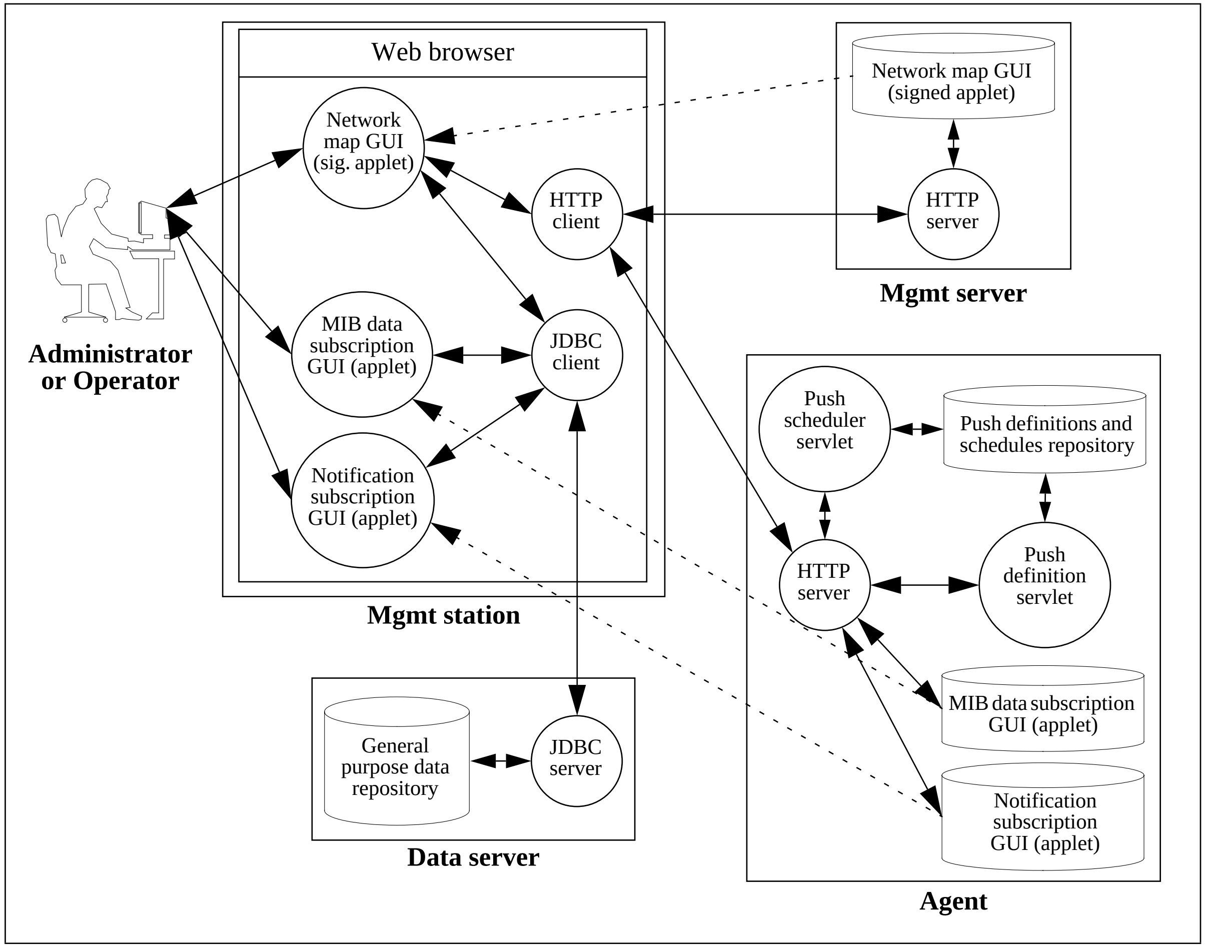
`get`: (2xOID) + value

`get-next`: (3xOID) + value

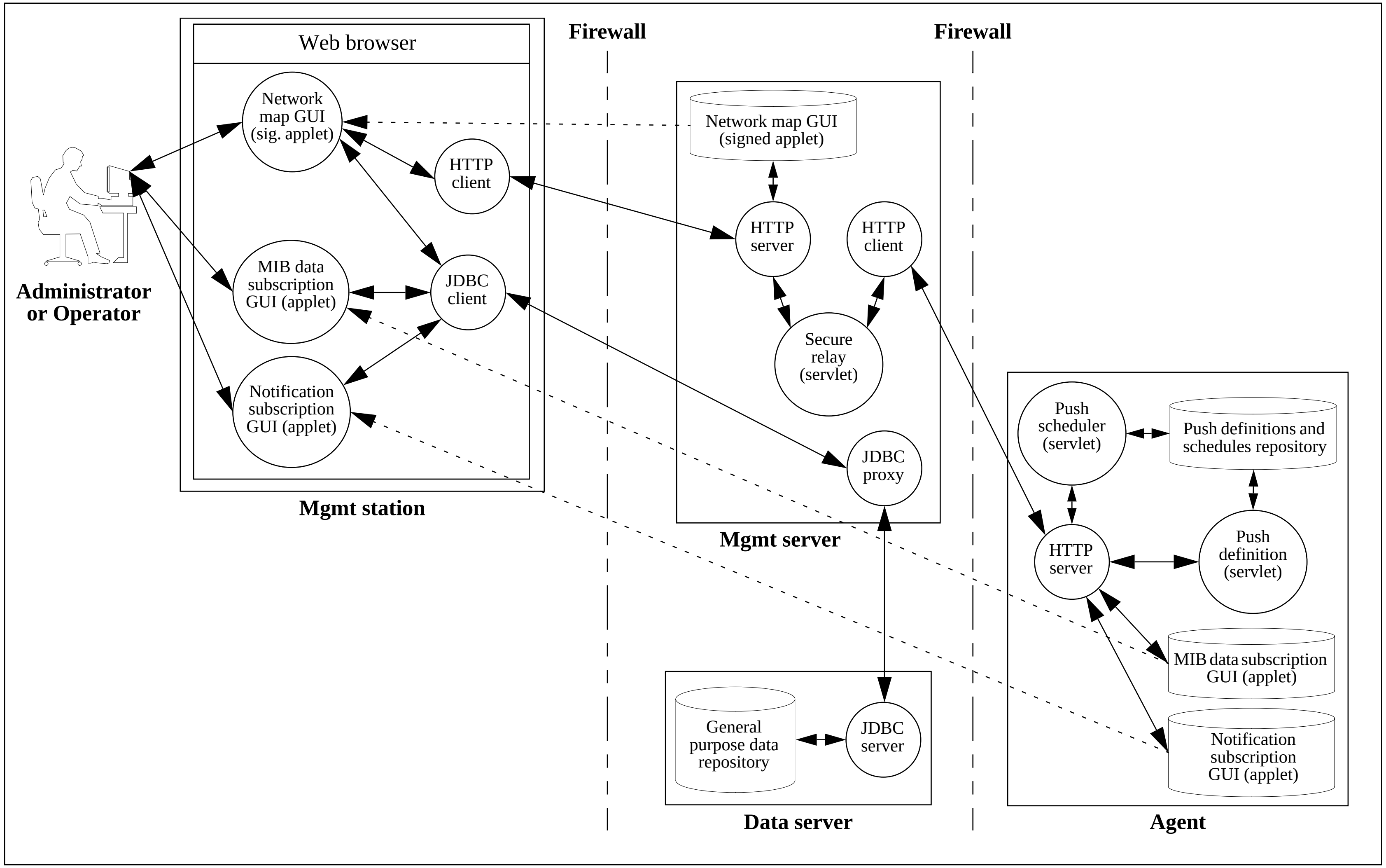
Characterization of the Push Model

- Variant of the Publish-Subscribe design pattern (Observer in [Gamma *et al*. 1995]):
  - one subscriber (manager), many publishers (agents)
  - 3 phases: publication, subscription, and distribution
- Pseudo client-server communication model:
  - client sends data to server
  - server may acknowledge (e.g., SNMPv3 informs) or not acknowledge (e.g., SNMPv1 traps and SNMPv2 notifications) receipt of this data
- Client = agent
- Server = manager
- Parallel and independent data transfers initiated by the clients

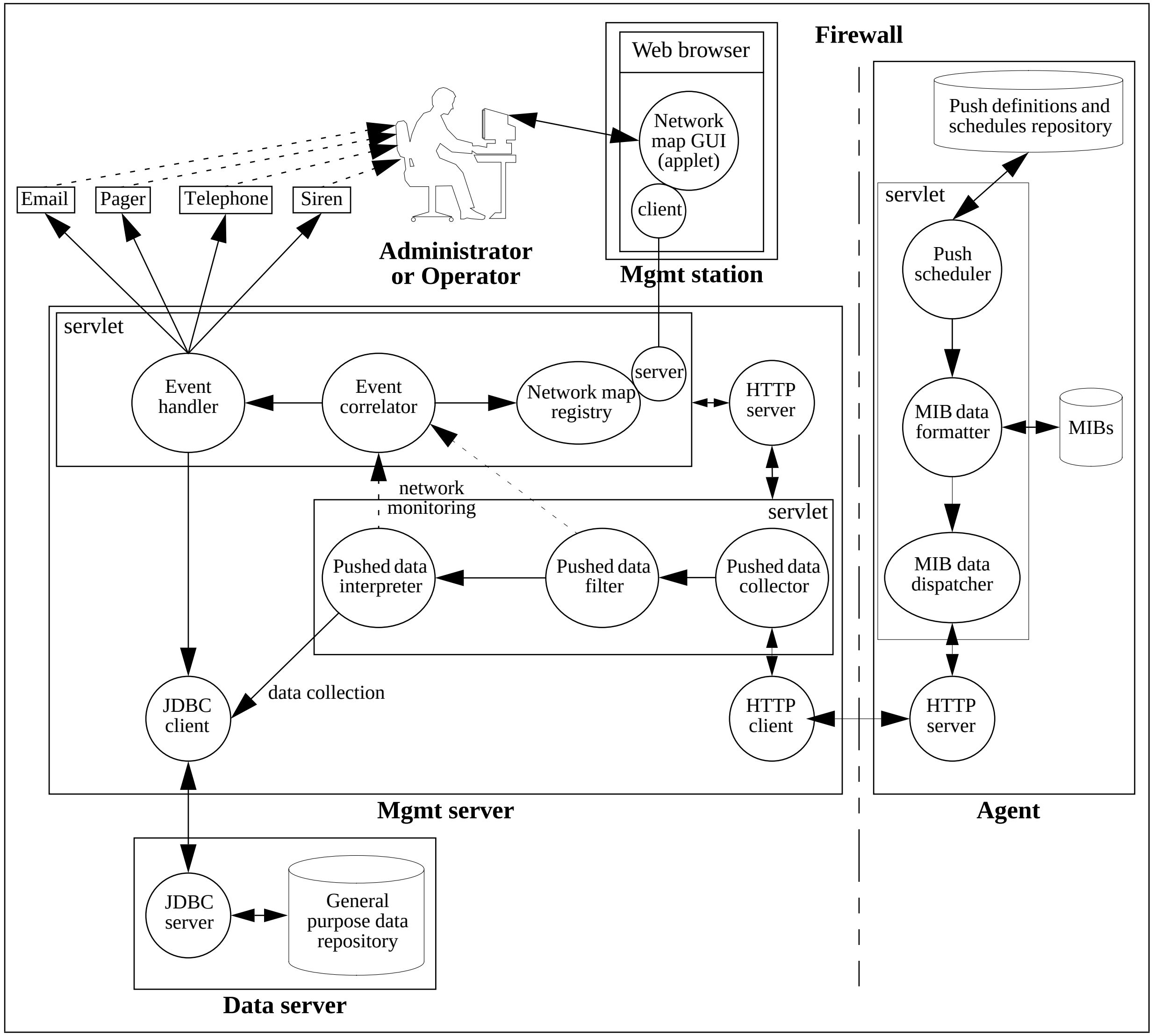
Publication and Subscription Phases


Web browser
Network map GUI (sig. applet)
HTTP client
MIB data subscription GUI (applet)
JDBC client
Notification subscription GUI (applet)
Mgmt station
Administrator or Operator
Network map GUI (signed applet)
HTTP server
Mgmt server
Push scheduler servlet
Push definitions and schedules repository
HTTP server
Push definition servlet
MIB data subscription GUI (applet)
Notification subscription GUI (applet)
Agent
General purpose data repository
JDBC server
Data server

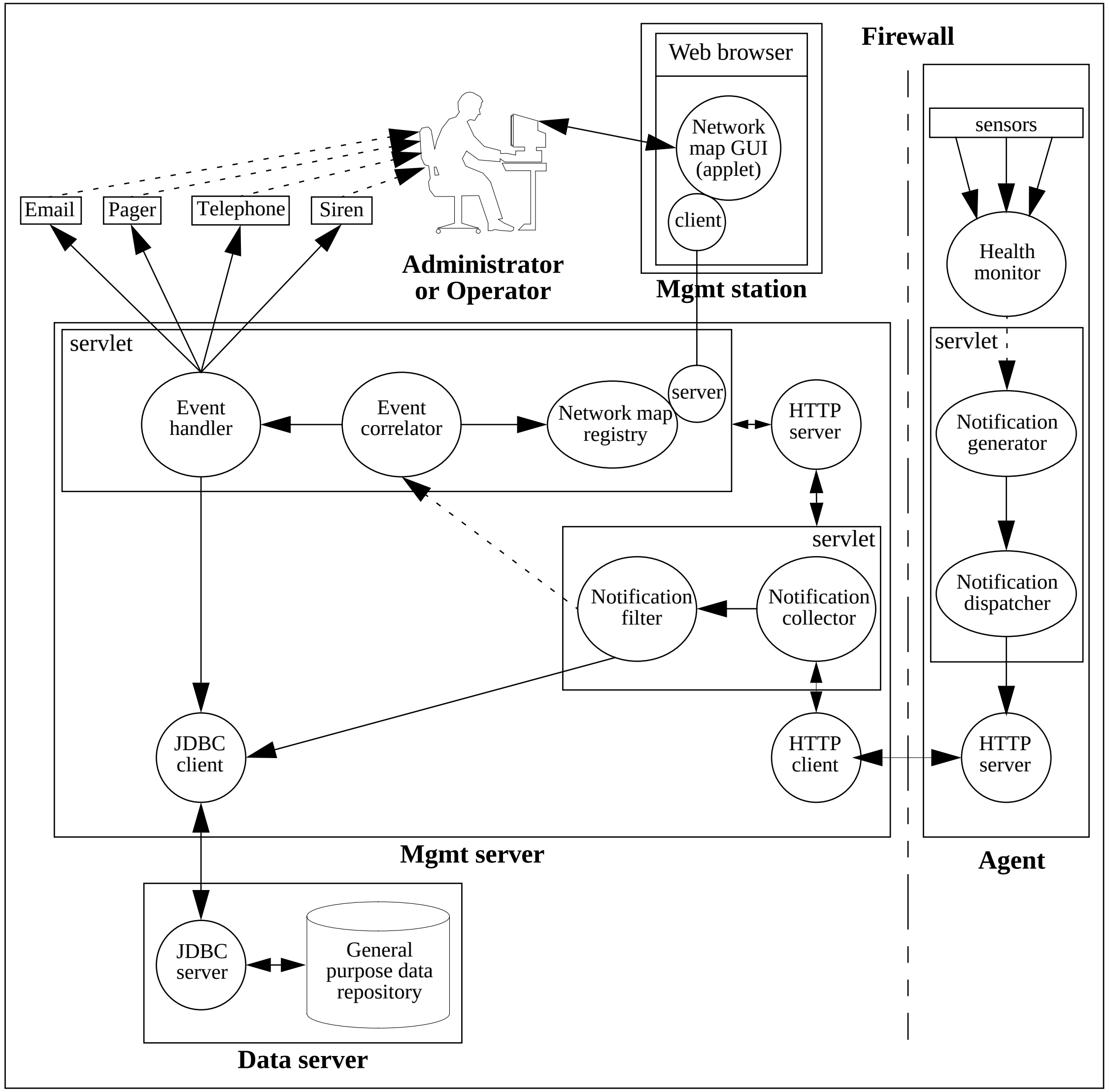
Publication and Subscription Phases (Firewall)


Firewall
Firewall
Web browser
Network map GUI (sig. applet)
HTTP client
MIB data subscription GUI (applet)
JDBC client
Notification subscription GUI (applet)
Administrator or Operator
Mgmt station
Network map GUI (signed applet)
HTTP server
HTTP client
Secure relay (servlet)
JDBC proxy
Mgmt server
General purpose data repository
JDBC server
Data server
Push scheduler (servlet)
Push definitions and schedules repository
HTTP server
Push definition (servlet)
MIB data subscription GUI (applet)
Notification subscription GUI (applet)
Agent

Distribution Phase for Monitoring and Data Collection



Distribution Phase for Notifications



Part 3: New Communication Model

- HTTP
- UDP --> TCP
- Persistent TCP connections
- Two connections per agent
- Firewalls
- Persistent HTTP connections with MIME multipart
- Timeouts and reconnections

Communication based on HTTP (1/2)

- Four techniques to communicate between agents and managers:
  - HTTP
  - sockets
  - Java RMI
  - Java IDL (CORBA)
- Distributed objects (Java RMI or CORBA):
  - telecoms world = yes
  - IP world = no
  - the *my-middleware-is-better-than-yours* syndrome

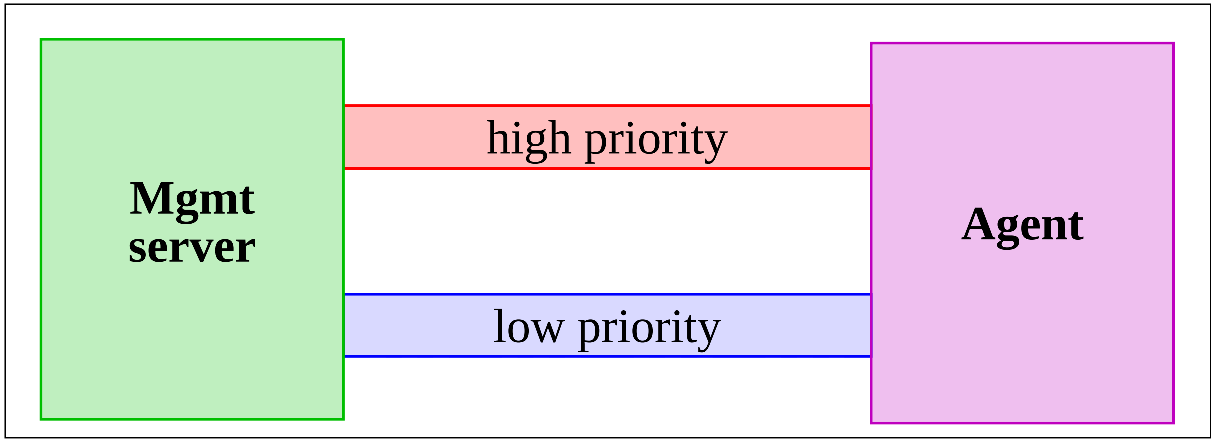
Communication based on HTTP (2/2)

- HTTP > sockets:
  - avoid a domain-specific transfer protocol
  - firewall setup easier for nonexperts:
    - important for small and midsize companies
  - manager: natural communication between servlets
  - same technology:
    - between agents and manager
    - within the manager

Persistent TCP Connections

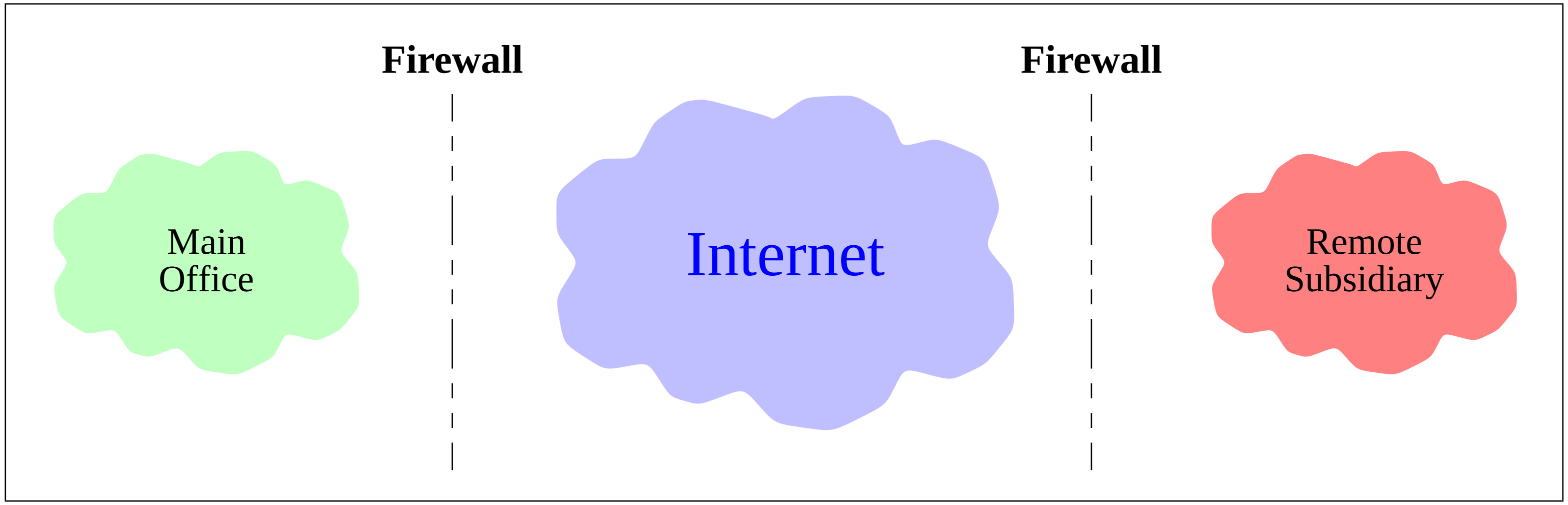
- TCP vs. UDP:
  - decrease losses of mgmt data:
    - ➠ still no guarantee of delivery
  - retransmissions and ack’s need not be performed at the app. level:
    - ➠ better interoperability
    - ➠ simpler application
- Persistent TCP connections:
  - avoid overhead of frequently setting up and tearing down connections
  - necessary for security reasons: the agent pushes mgmt data in a pre-existing connection

Two Persistent Connections Per Agent



- High priority: e.g., urgent SNMP notifications
- Memory overhead for the manager:
    - several MBytes to manage 100s of agents
    - requires special tuning of the kernel:
        - ➠ drawback: we still need a dedicated mgmt platform

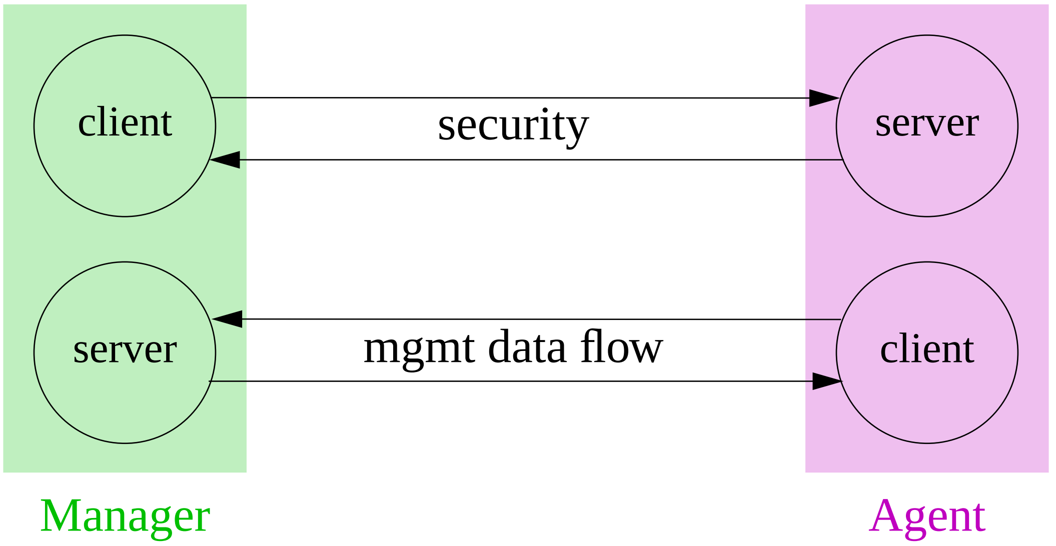
Firewalls



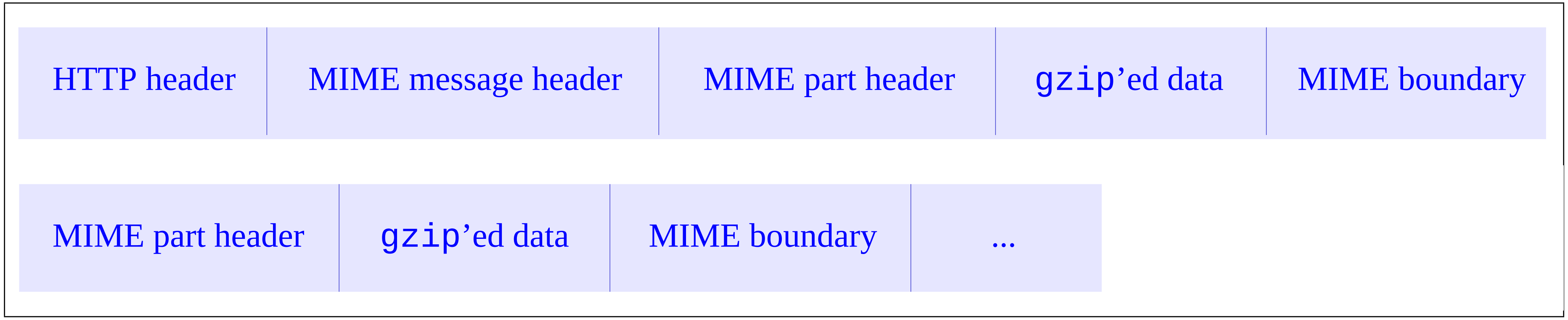
- Robustness principle: TCP connections should be created by internal trusted manager, not external untrusted agent:
  - avoid TCP ports probing by external intruders
  - avoid certain DoS attacks (e.g., TCP SYN flooding)

Reversed Client and Server

- Firewalls --> positions of client and server now reversed:
  - transfer of mgmt data initiated by the agent
  - client side of the persistent connection still on the manager side
  - we want the server to initiate a transfer in a client-server architecture!



Persistent HTTP Connections with MIME Multipart



HTTP header | MIME message header | MIME part header | `gzip`'ed data | MIME boundary

MIME part header | `gzip`'ed data | MIME boundary | ...

MIME = Multipurpose Internet Mail Extensions

- Advantages:
  - simple to implement
  - firewalls: minor change (assuming Web access already)
- Drawback:
  - how does the manager detect that a connection was broken?

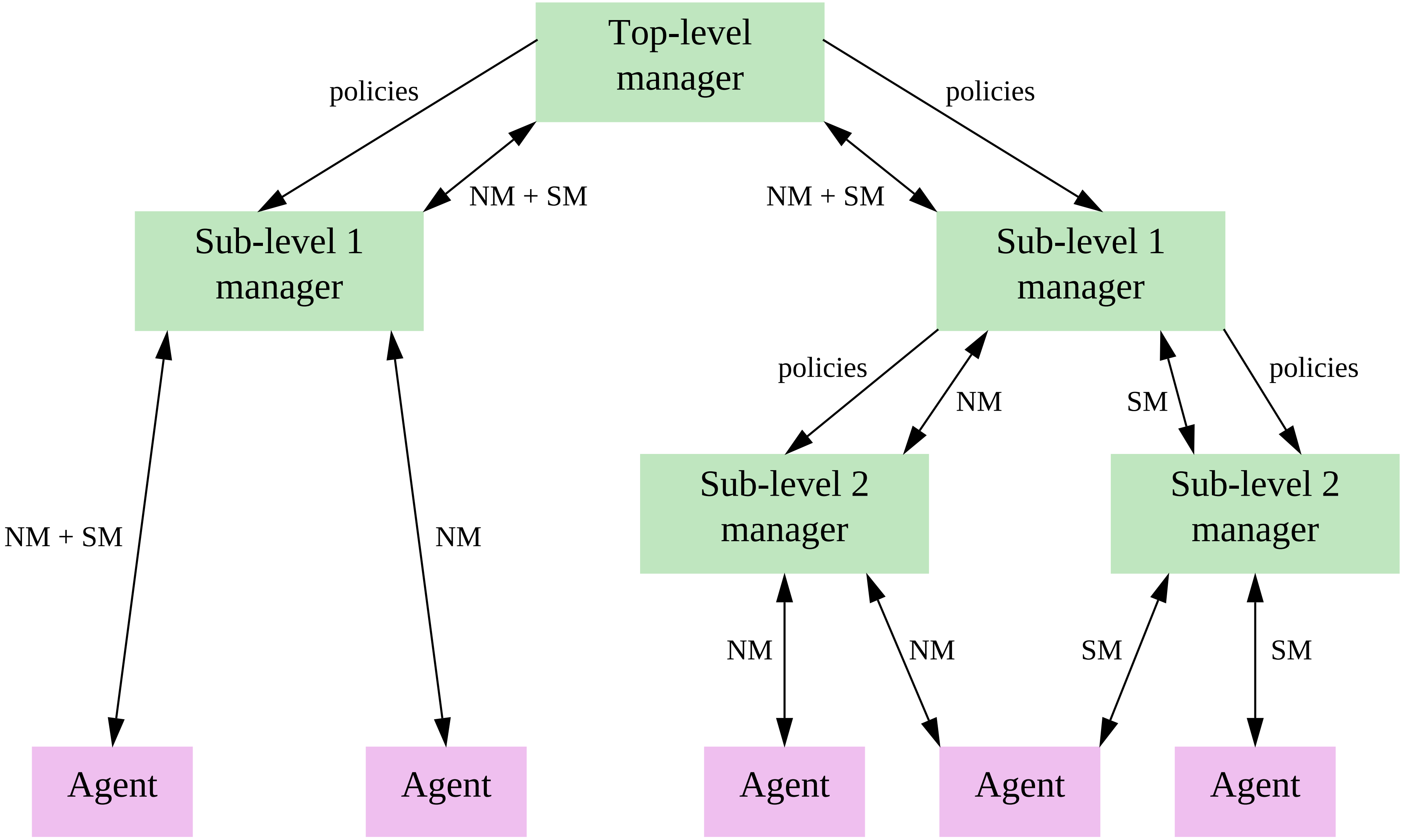
Timeouts and Reconnections

- Persistent connections:
  - timeouts by operating system and HTTP server?
  - how does the manager reconnect in case of teardown?
- The agent detects a transmission problem after 9 minutes (or TCP_MAXRT in Posix.1g), but the manager does not
- The agent knows when it reboots, but the manager does not
- Three solutions:
  - per kernel: keepalives (SO_KEEPALIVE):
    - ➠ Linux kernel 2.3.28: `tcp_keepalive_time` (7200 s), `tcp_keepalive_intvl` (75 s), `tcp_keepalive_probes` (9)
  - per socket: read timeout (SO_RCVTIMEO or select(...,timer))
  - per socket: keepalives (TCP_KEEPALIVE in Posix.1g)

Part 4: XML

- Why use XML?
  - A truce in the middleware war
  - More generic than IIOP and JRMP
  - Low footprint on agents and managers
  - Cost =~ zero:
    - a lot of freeware available
  - Demanded by customers:
    - becoming ubiquitous in software eng.
  - Feature rich:
    - state: transfer data
    - behavior: invoke remote methods

XML for Distribution

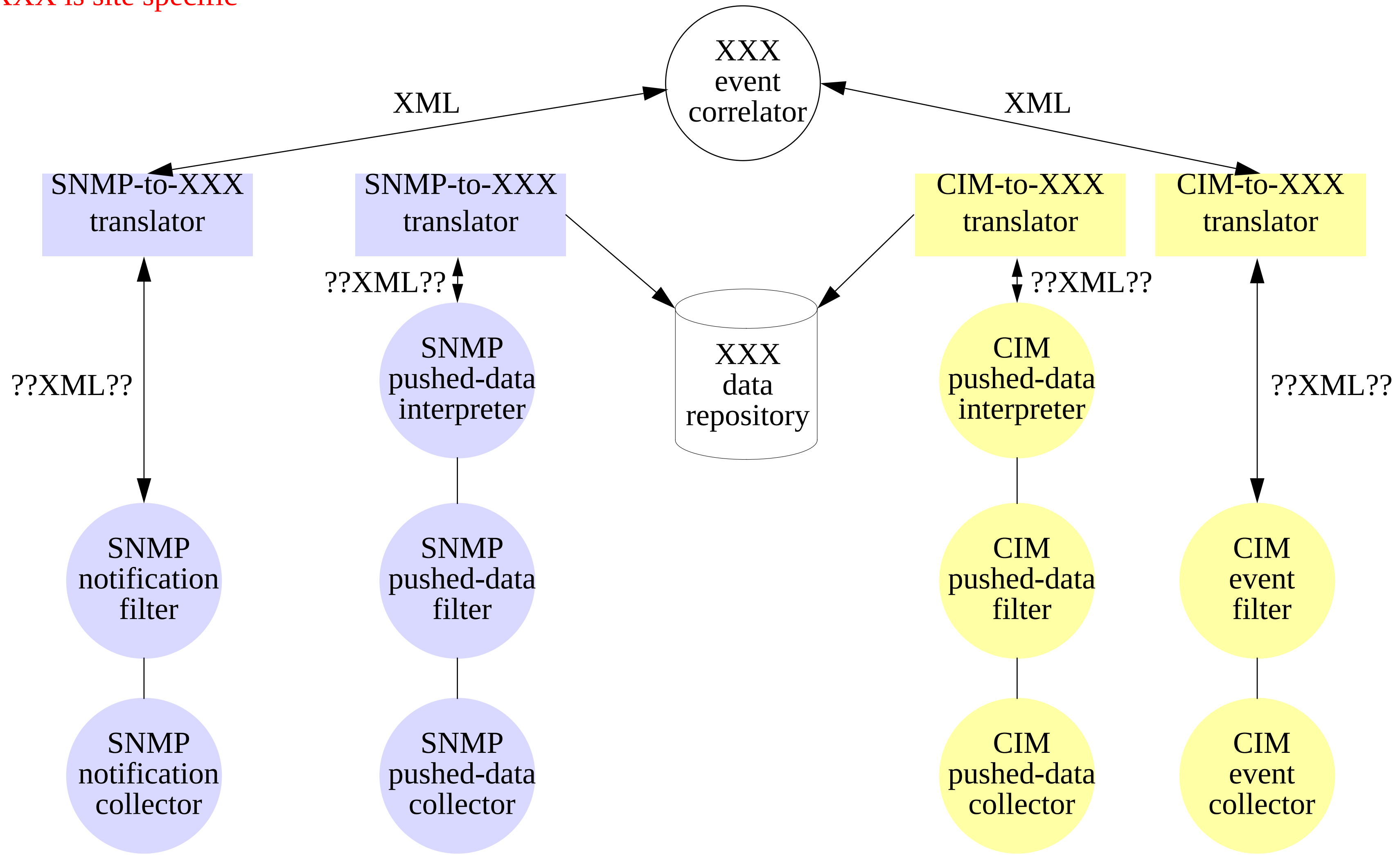


XML for High-Level Semantics

- Clean invocation of remote methods:
  - no need to resort to SNMP’s programming by side effect
- The DMTF learned from the IETF’s mistakes:
  - working on instrumentation MIBs *and* high-level MIBs
- XML renders easy many tasks that are not with SNMP:
  - transfer SNMP MIB table in one bulk (no more “holes”)
  - transfer entire time series for 24h in one bulk
  - ...
- XML interfaces nicely with OO info. models (e.g., CIM), which offer high-level semantics to mgmt applications designers

XML: Dealing with Multiple Information Models

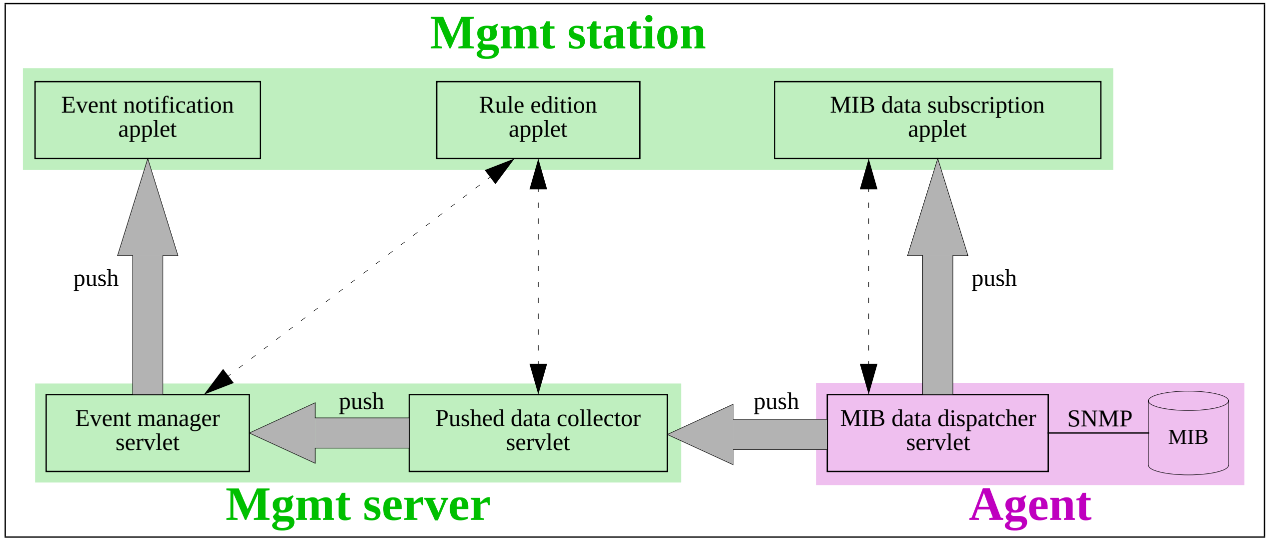
XXX is site specific

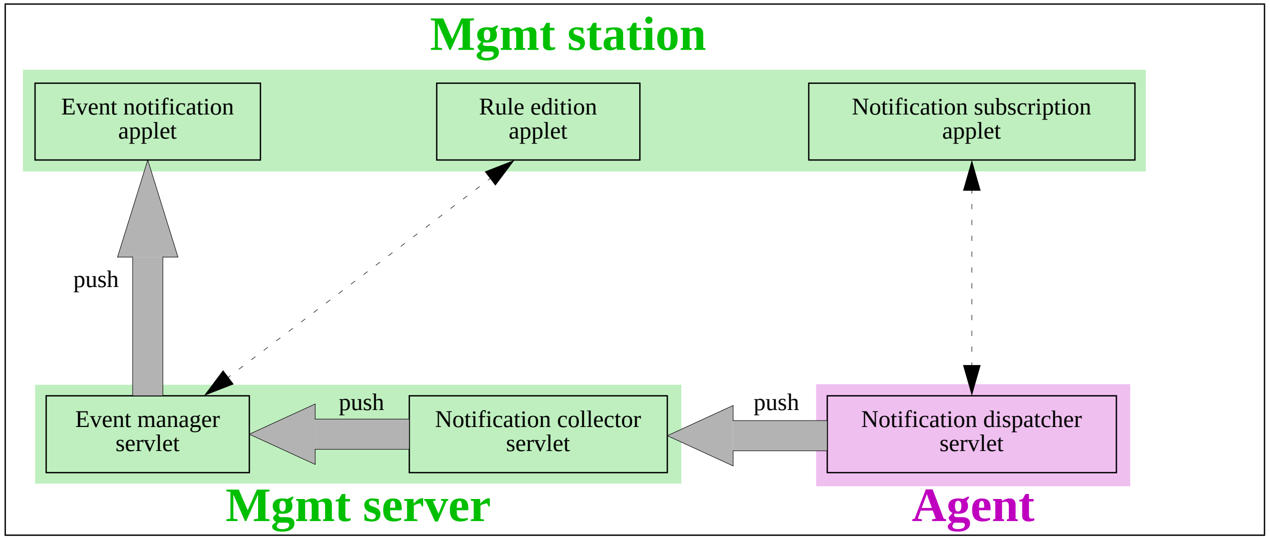


New MIME Types for Part Headers

- Three levels of granularity:
    - information model:
        - e.g., CIM-to-XML, SNMPv1-to-BER, SNMPv2c-to-string
    - RFC:
        - e.g., RFC2261-to-Java, RFC2271-to-string, RFC2571-to-XML
    - XML mapping:
        - e.g., CIM2.2-to-XML-v2.0, CIM3.0-to-XML-v0.1
- Two naming schemes for the new MIME types:
    - Content-Type=”CIM2.2-to-XML-v2.0”
        - poor scalability and scalability (constant flow of updates by IANA/ICANN)
    - Content-Type=”application/mgmt”; mapping=”CIM2.2-to-XML” version=”2.0”
        - our solution

Part 5: JAMAP


Mgmt station
Event notification applet
Rule edition applet
MIB data subscription applet
push
push
Event manager servlet
push
Pushed data collector servlet
push
MIB data dispatcher servlet
SNMP
MIB
Mgmt server
Agent


Mgmt station
Event notification applet
Rule edition applet
Notification subscription applet
push
Event manager servlet
push
Notification collector servlet
push
Notification dispatcher servlet
Mgmt server
Agent

JAMAP: A Research Prototype

- Purpose:
  - demonstrate push and MIME multipart
  - demonstrate simplicity of implementation:
    - the core was coded in 2 weeks
- Many simplifications:
  - NFS instead of JDBC
  - only SNMP MIBs, no CIM MIBs
  - only serialized Java objects, no XML
  - simplistic event correlator
  - partial support for notifications
  - one OID per MIME part
- Wanted: manpower!

Conclusion: The Problem Is Solved (1/4)

- For customers:
  - platforms are too expensive (hardware and software):
    - ➠ mgmt GUIs are less expensive (applets)
    - ➠ different vendors write different parts of the mgmt application --> less costly
    - ➠ capitalize on previous investment (e.g., use in-house RDBMS)
  - limited support for third-party RDBMS vendors:
    - ➠ no need for peer-to-peer agreement, use JDBC or XML instead
  - insufficient integration:
    - ➠ flexible architecture for integrating network, systems, application, service, and policy mgmt (esp. SNMP and CIM MIBs)

Conclusion: The Problem Is Solved (2/4)

- For equipment vendors:
  - the support for device-specific mgmt GUIs is too expensive:
    - ➠ single applet
- For customers and equipment vendors:
  - poor time-to-market for mgmt GUIs:
    - ➠ zero time-to-market, whatever the market share
    - ➠ access to integrated mgmt for startup companies -> fair competition
  - MIB versioning:
    - ➠ the manager retrieves the mgmt GUI from the agent --> no version mismatch
  - investment bound to a specific operating system:
    - ➠ Java, HTTP, HTML, MIME, and XML are independent of the OS
    - ➠ still the problem of the JVM version

Conclusion: The Problem Is Solved (3/4)

- SNMP expertise is domain specific:
  - Web expertise is generic
- Scalability, network overhead, and latency problems:
  - BER encoding no longer used
  - SNMP protocol replaced with HTTP
  - compressed mgmt data
  - distribution with XML
- Low-level semantics:
  - the DMTF is currently working on instrumentation MIBs *and* high-level MIBs
  - site-specific applications now depend on standard technologies: XML, Java, etc.

Conclusion: The Problem Is Solved (4/4)

- Security:
  - HTTP security may be used instead of costly encryption hardware:
    - ➠ still weak security
    - ➠ better than SNMP’s community string
  - firewall setup: HTTP simpler than SNMP
- Unreliable transport protocol:
  - HTTP makes it possible to use TCP to transfer mgmt data
  - reliable transport layer for SNMP notifications:
    - ➠ important notifications are no longer lost for silly reasons
    - ➠ still no guarantee of delivery
- Evolution of SNMP hampered by legacy systems:
  - Web-based mgmt: start with a clean slate but preserve SNMP MIBs

New Problems

- Reliability of new mgmt platforms based on COTS components and OO frameworks:
  - new means buggy
- Integration of components sold by multiple vendors:
  - it does not work, whose fault is it? who should fix it?
  - need integrators
- Synchronization of all clocks (managers, agents)
- Java is slow, even with JIT compiler:
  - scalability of the mgmt server?
  - may need to resort to C++ --> compiled

Related Work (1/2)

- Architectures:
  - Bruins, Deri, Harrison *et al.*, Maston, Mullaney, Thompson, etc.
- Prototypes:
  - Marvel by Anerousis, CyberAgent by Burns and Quinn, Webbin by Barillaud *et al.*, WbASM by Kasteleijn, NetFinity by Reed *et al.*, etc.
- Commercial offerings:
  - `http://joe.lindsay.net/webbased.html`

Related Work (2/2)

- WBEM:
  - DMTF
  - HMMP --> HTTP + XML
  - new OO info. model: CIM
  - CIM-to-XML mapping (meta level)
  - extensions to HTTP: new headers for firewalls
  - ongoing: working groups are defining CIM MIBs
- Java-based mgmt:
  - Sun Microsystems and the Java Community
  - OO mappings of existing info. models
  - communication via Java RMI (distributed OO)
  - ongoing: JMX (agent) and FMA (manager) are merging

Future Work

- Convince the DMTF and Sun Microsystems to adopt our mgmt architecture and communication model
- Convince startups to develop smart software for the mgmt server
- Register new MIME type with IANA/ICANN
- Define SNMP-to-XML mapping:
  - MIB level or meta level?
- Coexistence of SNMP and CIM MIBs:
  - what are the issues?
- Design patterns:
  - how to avoid well-known design mistakes?