Traditional Approaches to Distributed Management

Jürgen Schönwälder

j.schoenwaelder@iu-bremen.de

International University Bremen

Campus Ring 1

28725 Bremen, Germany

http://www.faculty.iu-bremen.de/schoenw/

Classification and Approaches

Classification

- Let m be the total number of managers, a the total number of agents, and let $n = m + a$ denote the total number of elements in the management system.
- We can distinguish four classes of distributed network management systems:

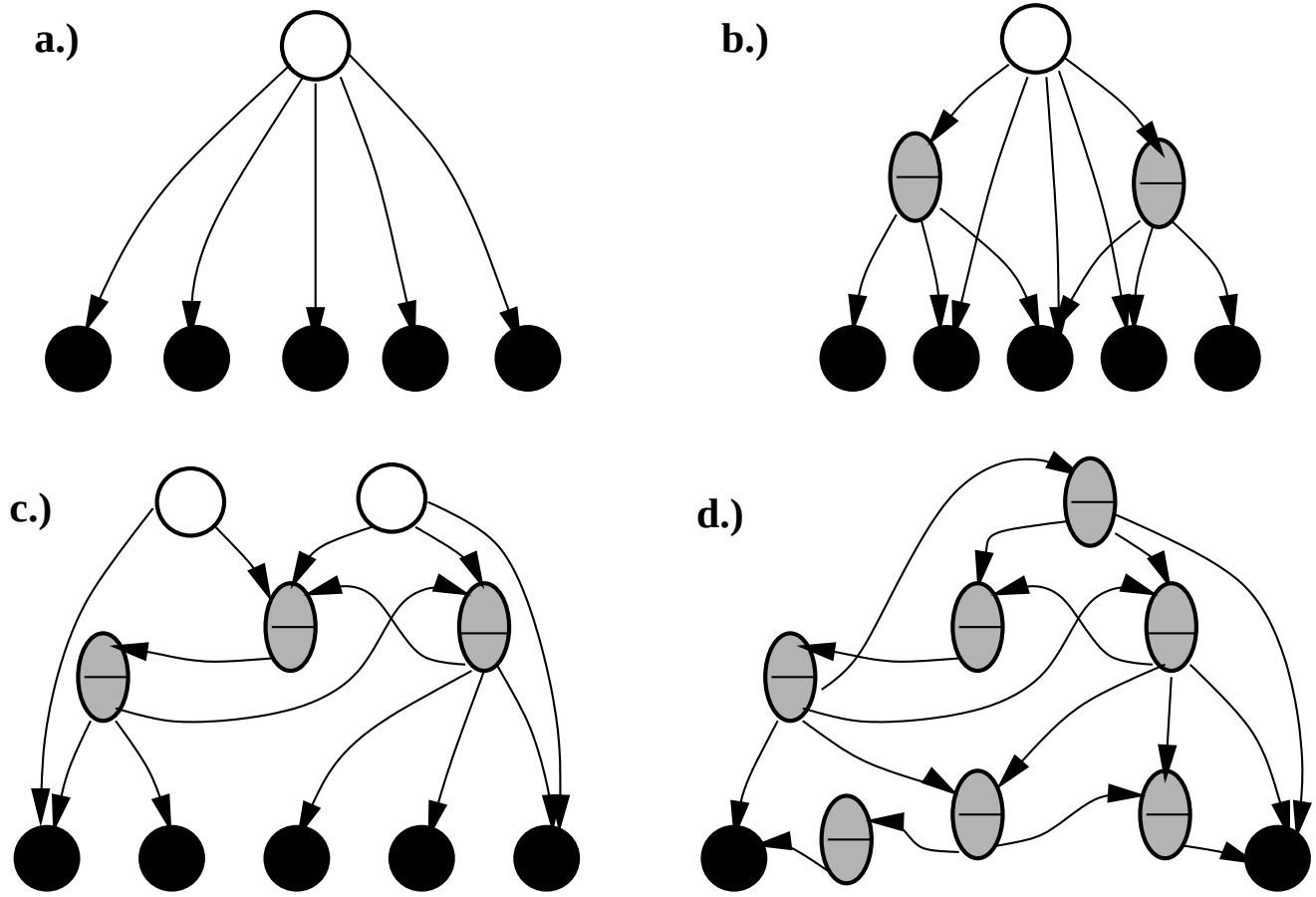
  a.) $1 = m$: centralized management

  b.) $1 < m \ll n$: weakly distributed management

  c.) $1 \ll m < n$: strongly distributed management

  d.) $m \approx n$: cooperative management

Classification (cont.)



- For other (more detailed) classifications see [1, 2].

Approach #0: Remote Operations

- Idea: Execute a management operation on a remote network element and retrieve the results.
- Classic remote procedure call (RPC) idea, can be realized using standard RPC protocols or middleware frameworks.
- Remote procedure call semantics are fixed at design time.
- The set of available RPCs together with their fixed semantics determine the distribution that can be achieved.

Approach #1: Management by Delegation

- Idea: Dynamically delegate management functions (scripts) to remote elements, execute them and retrieve their results.
- Classic remote evaluation (REV) idea which requires
  - a remote execution environment and
  - a mechanism to transfer executable content in addition to parameters and results.
- Dynamic adaption of distributed management functions possible.
- Raises some security concerns (safe execution environments, signed code, . . .).

Approach #2: Mobile Agents

- Idea: The code implementing a management function together with the state data produced by the execution of the code travels through the network in order to solve a management problem.
- Requires a remote execution environment capable to
  - snapshot an execution state,
  - serialize and transfer the snapshot state, and
  - restore the execution state on a remote node.
- Not all management functions benefit from execution mobility.
- Raises additional security issues (control of mobile agents).

Approach #3: Active Networks

- Idea: Devices handling packets are dynamically (re-)programmed and in the extreme case, every packet contains executable code which instructs network devices how to handle the packet.
- Requires execution platforms on (ideally) all network elements.
- Raises next to security issues significant performance issues (think about devices handling 10+ GByte/s).
- Interesting research idea, but not really practical (Rolf might disagree)

References

[1] J. P. Martin-Flatin, S. Znaty, and J. P. Hubaux. A Survey of Distributed Enterprise Network and Systems Management Paradigms. *Journal of Network and Systems Management*, 7(1):9–26, March 1999.

[2] M. Kahani and H. W. P. Beadle. Decentralized Approaches for Network Management. *Computer Communication Review*, 27(3):36–47, July 1997.

[3] A. Birrell and P. Nelson. Implementing Remote Procedure Calls. *ACM Transactions on Computer Systems*, 2(1):39–59, 1984.

[4] J. W. Stamos and D. K. Gifford. Implementing Remote Evaluation. *IEEE Transactions on Software Engineering*, 16(7):710–722, July 1990.

[5] Y. Yemini, G. Goldszmidt, and S. Yemini. Network Management by Delegation. In *Proc. International Symposium on Integrated Network Management*, April 1991.

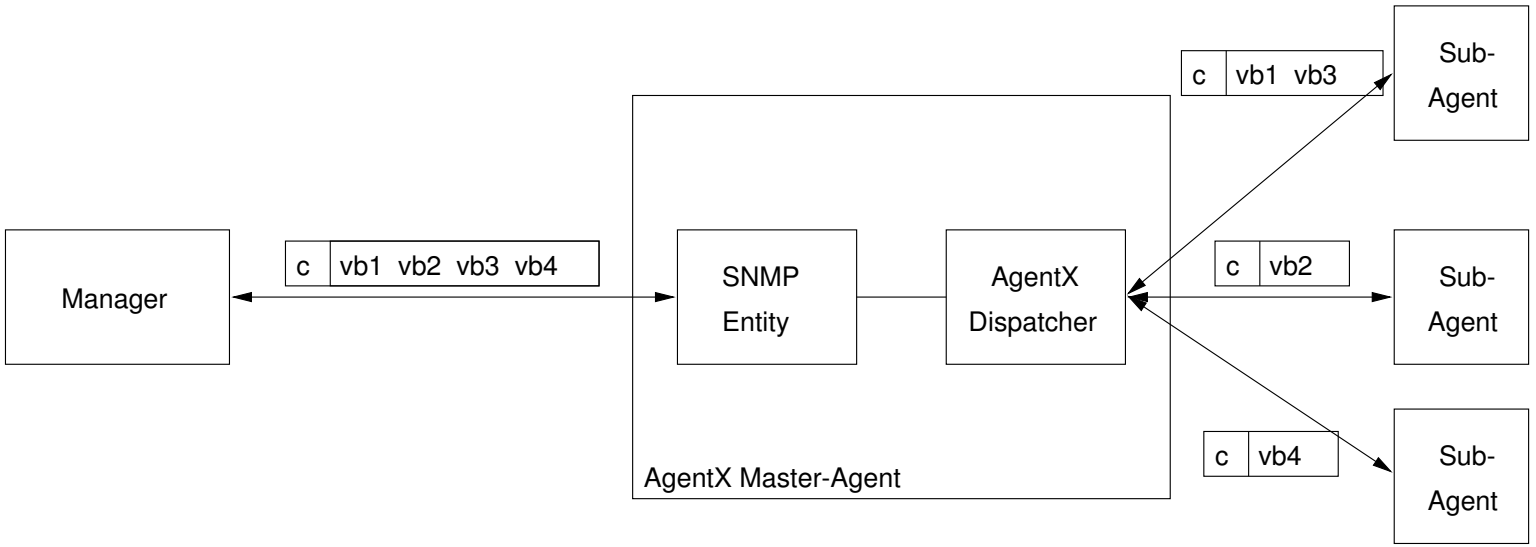
[6] A. Bieszczad, B. Pagurek, and T. White. Mobile Agents for Network Management . *IEEE Communications Surveys*, 1(1), 1998.

[7] A. Puliafito, S. Riccobene, and M. Scarpa. Which paradigm should I use? An analytical comparison of the client-server, remote evaluation and mobile agent paradigms. *Concurrency and Computation*, 13(1), January 2001.

[8] R. Kawamura and R. Stadler. Active Distributed Management for IP Networks. *IEEE Communications Magazine*, 38(4):114–120, April 2000.

Internet Standardization

Extensible Agents



- Separation of the management protocol from the instrumentation.
- Instrumentations can be added dynamically.
- Extensible agents are transparent for management applications.

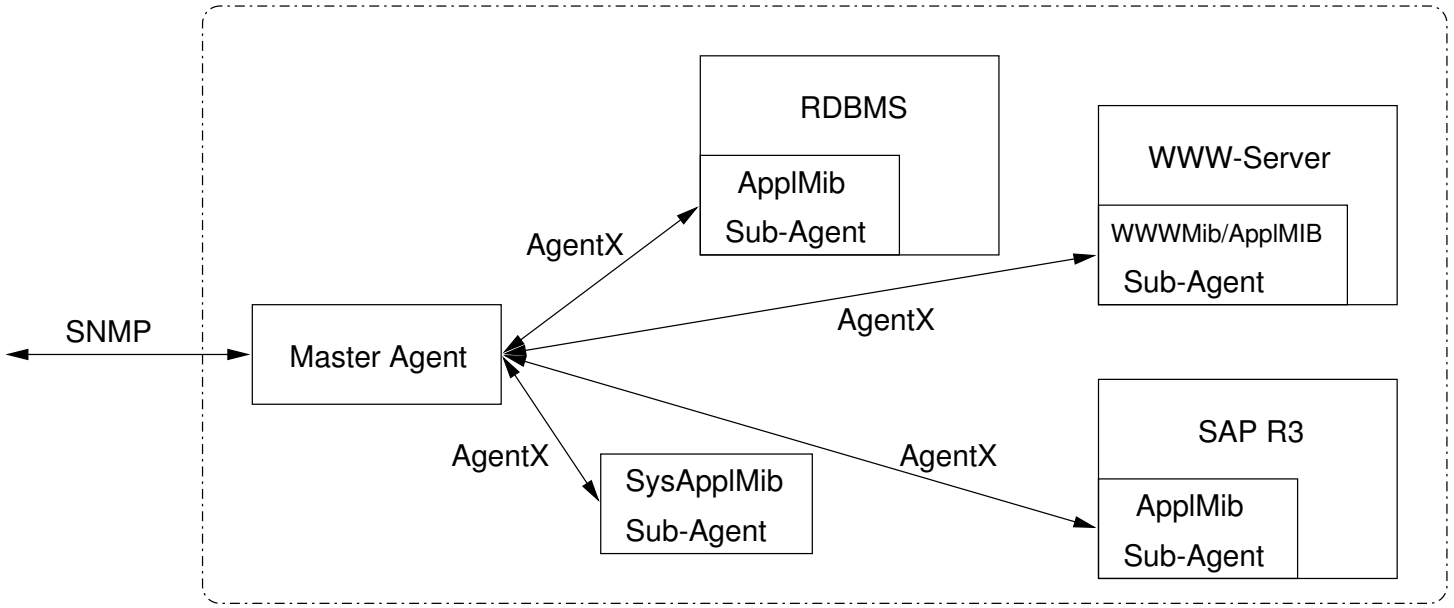
History of Extensible Agent Technology

- SNMP MUX Protocol (SMUX) circa 1991 (RFC 1227)
- Proprietary solutions:
  - IBM's Distributed Protocol Interface (DPI)
  - SNMP Research's Emanate
  - Digital's extensible SNMP agent (eSNMP)
  - ...
- Agent Extensibility Protocol (AgentX) circa 1998 (RFC 2741)

Agent Extensibility Protocol Version 1 (RFC 2741)

- Open standard for extensible agents, based on experience with non-standard solutions.
- Core technology for modular networking devices.
- Required for portable system and application management agents.
- The AgentX master agent is MIB ignorant and SNMP omniscient.
- The AgentX sub-agent is SNMP ignorant and MIB omniscient.
- AgentX supports sub-agent integration through index allocation.
- Efficient AgentX message formats and encodings.

Application Management with AgentX



- Application management MIBs require instrumentation in the applications.
- AgentX provides the infrastructure for implementing application management MIBs.

AgentX Status

- Several AgentX implementations are available (including NET-SNMP).
- The most widely used operating systems do not yet support AgentX natively.
- An experimental Linux kernel implementation of AgentX sub-agents has been done as a research project.
- Limitations:
  - No access to security/access control related information from the sub-agent.
  - No communication/coordination facilities between sub-agents.
  - No support for invoking SNMP command generator operations from a sub-agent.

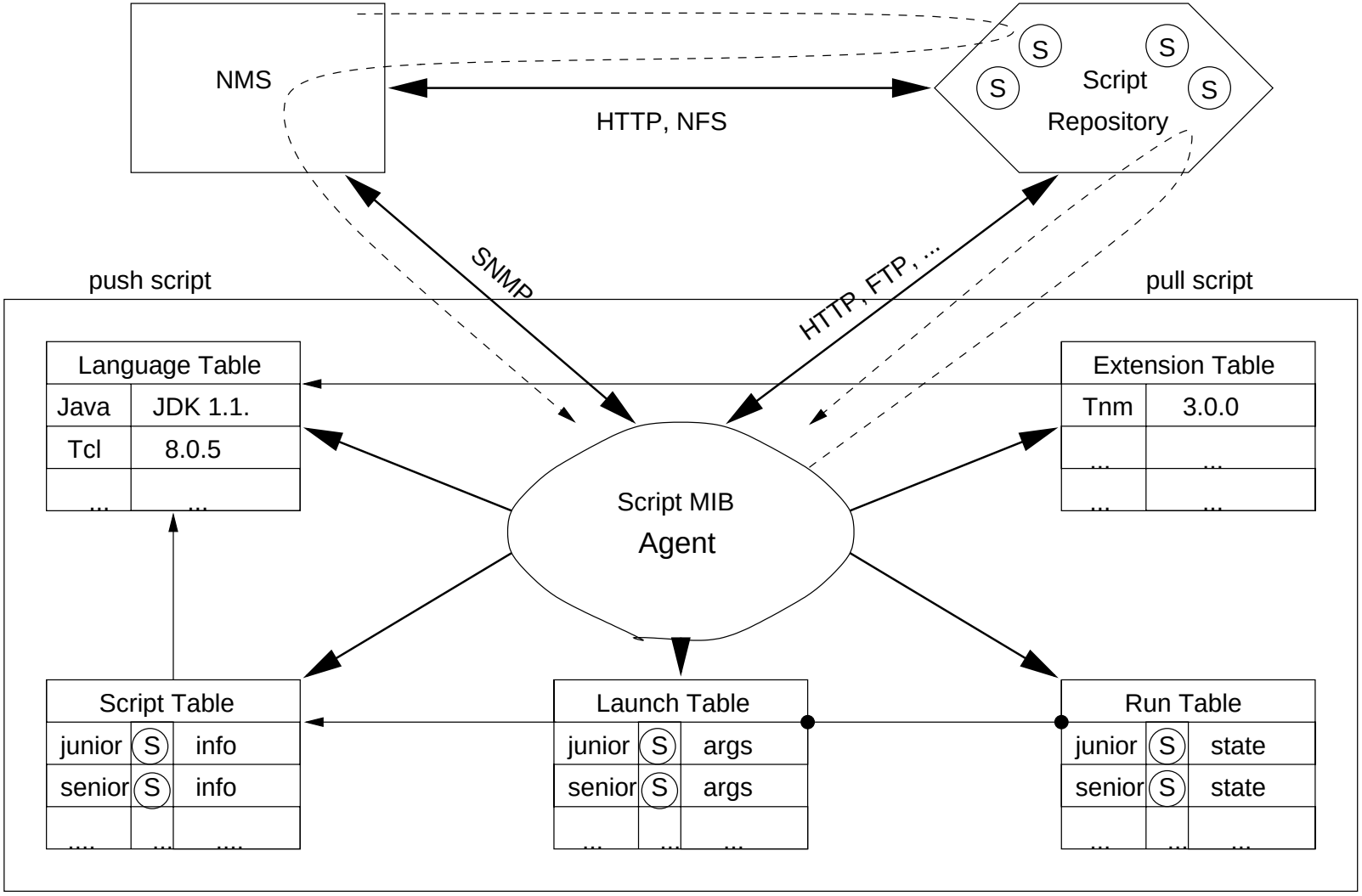
Remote Operations

- Remote Operations MIBs (RFC 2925)
  - Enables management applications to perform a ping, traceroute, or name lookup operations on a remote system.
- Expression MIB (RFC 2982)
  - Computation of expressions over MIB variables.
  - Wildcarding can be used to apply a single expression to a complete table.
  - Expressions are intended to operate on local MIB data.
  - Expressions over counter objects require continuous sampling and maintenance of state information.

Remote Operations (cont.)

- Event MIB (RFC 2981)
  - Generation of an event if a MIB variable changes or crosses thresholds.
  - Events may cause actions such as notifications or set operations.
  - Triggers on counter variables require continuous sampling and state information.
- Scheduling MIB (RFC 3231)
  - Scheduled actions (setting a MIB variable) based on periodic schedules and calendar schedules.
  - One-shot schedules are calendar driven schedules that fire only once.
  - Handles time transitions (ambiguous and nonexistent times).

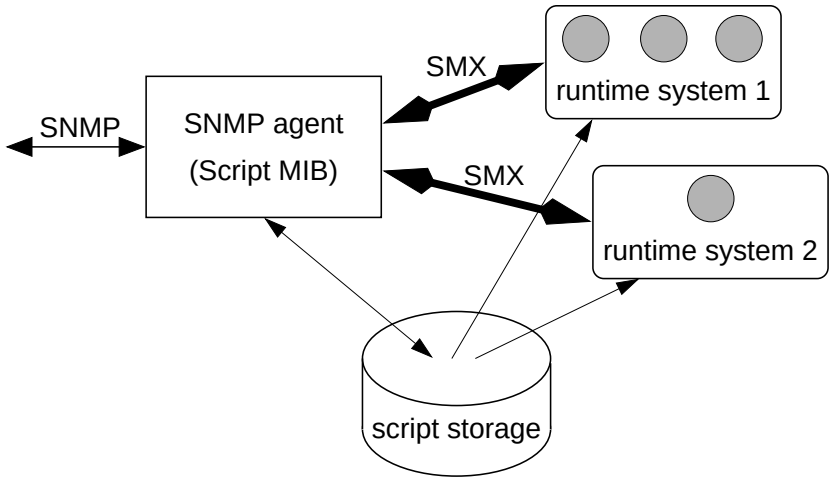
Management by Delegation



Properties of the Script MIB (RFC 3165)

- Language and runtime system independent
- Supports script push via SNMP and pull via URIs
- Script and language/runtime versioning support
- Table indexing supports the creation of “sandboxes”
- Resource controls to protect against faulty scripts

Script MIB Extensibility Protocol (RFC 3179)



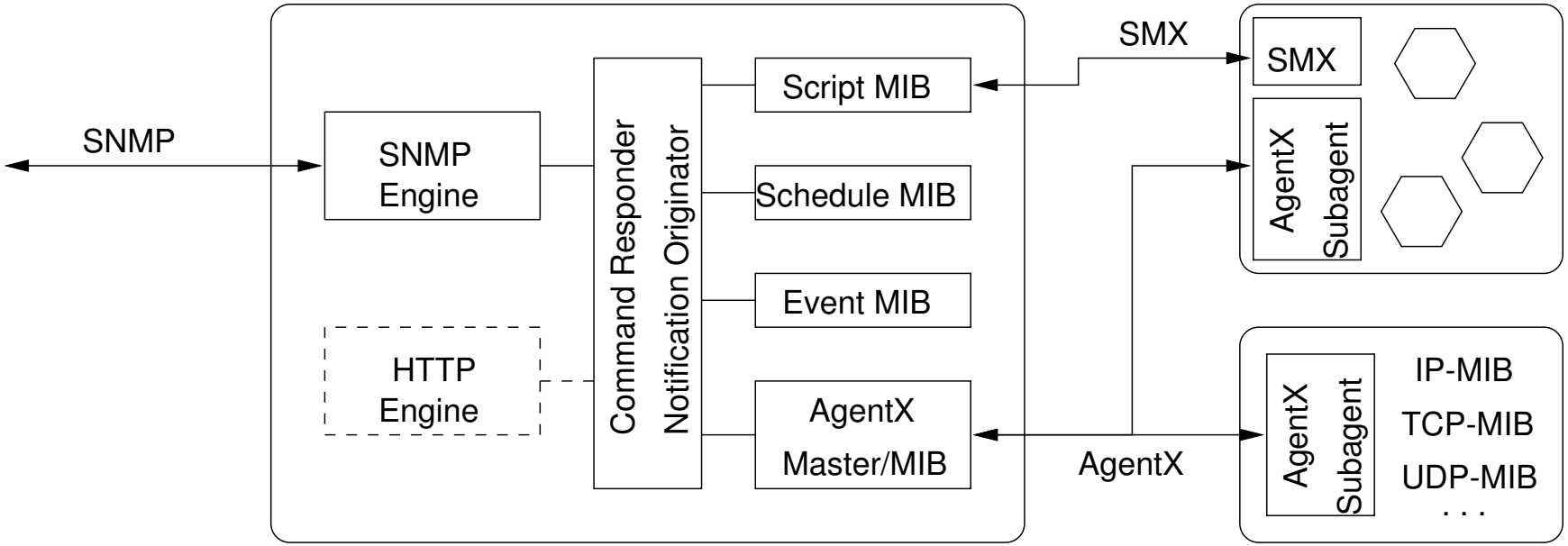
- Separates language specific runtime systems from the runtime system independent MIB implementation.
- Multiple runtimes with different security profiles.
- Simple protocol running over a local TCP connection.
- Initial handshake verifies a security cookie.
- Local file system used to pass executable code.

Putting Things Together…

- Smart SNMP Agents:
  - SNMPv3 provides message security and authorization services.
  - AgentX provides the services for dynamic agent extensions.
  - Distributed management MIBs realize the control infrastructure (scheduling, scripting, event binding).
- Smart SNMP agents that can perform some management tasks autonomously.

$\Rightarrow$ Lack of a “manager extensibility” protocol which allows “MIBlets” to access other SNMP capable devices via the existing SNMP engine.

Smart SNMP Agents



+ Secure & extensible & programmable
- Special rather than general purpose technology
- Difficult to understand and to debug
- **Too complex to be used outside of research labs**

References

[1] M. Daniele, B. Wijnen, M. Ellison, and D. Francisco. Agent Extensibility (AgentX) Protocol Version 1. RFC 2741, Digital Equipment Corporation, IBM T. J. Watson Research, Ellison Software Consulting, Cisco Systems, January 2000.

[2] D. Levi and J. Schönwälder. Definitions of Managed Objects for the Delegation of Management Scripts. RFC 3165, Nortel Networks, TU Braunschweig, August 2001.

[3] J. Schönwälder and J. Quittek. Script MIB Extensibility Protocol Version 1.1. RFC 3179, TU Braunschweig, NEC Europe Ltd., October 2001.

[4] D. Levi and J. Schönwälder. Definitions of Managed Objects for Scheduling Management Operations. RFC 3231, Nortel Networks, TU Braunschweig, January 2002.

[5] K. White. Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations. RFC 2925, IBM, September 2000.

[6] R. Kavasseri and B. Stewart. Distributed Management Expression MIB. RFC 2982, Cisco Systems, Inc., October 2000.

[7] F. Strauß, J. Schönwälder, and S. Mertens. JAX - A Java AgentX Subagent Toolkit. In *Proc. 1st IEEE Workshop on IP-oriented Operations & Management*, Cracow, September 2000.

[8] J. Schönwälder, J. Quittek, and C. Kappler. Building Distributed Management Applications with the IETF Script MIB. *IEEE Journal on Selected Areas in Communications*, 18(5):702–714, May 2000.

[9] J. Schönwälder and J. Quittek. Secure Internet Management By Delegation. *Computer*

Perspective

Distributed Algorithms

- Well established research area related to distributed systems since the 1980s.
- Algorithms for basic problems (wave, traversal, consensus, election, synchronization, decision, snapshot, . . .) are well understood.
- Recent research on overlay networks, distributed hash functions (CAN, Chord, Pasty, , . . .), and self-organization.
- However: Distributed algorithms have seen limited application to network management so far.

Resilience

- Robust Algorithms:
  - Distributed algorithms capable to produce correct output (e.g., decisions) under a given fault model.
  - No solution for purly asynchronous systems.
  - Depending on the fault model and the properties of synchronous systems, solutions with known properties exist.
- Self-Stabilizing Algorithms:
  - Distributed algorithms capable to recover a system from an illegal system state back into a legal state.
  - Systems may temporarily produce incorrect output, but they converges back to a stable state where they start to produce correct output again.

Next Steps in Distributed Management

- Start research on cooperative management
- Apply research results from distributed algorithms, and (distributed) dependability to network management problems
- Which assumptions are fair?
  1. Network nodes have significant resources available for management
  2. Network bandwidth and connectivity is always available
  3. Network nodes have autonomity (how much?)
  4. . . .

References

[1] A. Avizienis, J.-C. Laprie, and B. Randell. Fundamental Concepts of Dependability. In *Proc. 3rd Information Survivability Workshop (ISW-2000)*, October 2000.

[2] P. Jalote. *Fault Tolerance in Distributed Systems*. Prentice Hall, 1994.

[3] E.W. Dijkstra. Self-stabilizing Systems in Spite of Distributed Control. *Communications of the ACM*, 17(11):643–644, November 1974.

[4] D. Oppenheimer, A. Ganapathi, and D.A. Patterson. Why do Internet services fail, and what can be done about it? In *Proc. 4th Usenix Symposium on Internet Technologies and Systems*. Usenix, March 2003.

[5] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chuah, and C. Diot. Characterization of Failures in an IP Backbone. In *Proc. Infocom 2004*, 2004.