JAX:
A Java AgentX Sub-Agent Toolkit

Frank Strauß

Department of Computer Science
TU Braunschweig
Bültenweg 74/75
38106 Braunschweig
Germany

strauss@ibr.cs.tu-bs.de
http://www.ibr.cs.tu-bs.de/~strauss/

MIB Implementation Strategies

Classification of MIB Deployment

1. Standard MIBs: Some of them are widely implemented and deployed.
2. Vendor MIBs: Only feasible if costs look reasonable.
3. Site-specific MIBs: Requires detailed know-how, usually not reasonable.

Problems with Agent MIB Extensions

- Different incompatible proprietary technologies for agent extensions.
- Complicated and error-prone agent integration.
- $\rightarrow$ High development costs.

Especially vendor and site-specific MIBs are dependent on easy to use high-level SNMP agent toolkits.

Solution: Agent Extensibility Protocols

Master/Sub-Agent architecture:

- Separation of the MIB implementation from the SNMP agent.
- MIB implementations can be added to the agent dynamically.
- Complexity of security, access control and management of MIB regions is limited to the master agent.

Agent extensibility protocol efforts:

- SNMP Multiplexing Protocol (SMUX) (RFC 1227, 1991)
- SNMP Distributed Program Interface (DPI) (RFC 1228, 1991)
- Distributed Program Interface V2.0 (DPIv2) (RFC 1592, 1994)

Proprietary architectures:

- Simplified DPI by Digital
- Envoy by Epilogue Technologies
- EMANATE by SNMP Research

Problems

- Incompatible proprietary architectures.
- Standardized efforts suffer from performance, fault tolerance and data consistency problems.

Solution: A Standardized Agent Extensibility Protocol

- IETF AgentX Working Group, chartered in 1995
- Goals:
  - a *standardized* agent extensibilty framework → RFC 2741, Jan-2000, PS
  - an accompanied MIB → RFC 2742, Jan-2000, PS
- Advantages:
  - Independent from programming languages.
  - The master agent handles all security and access control processing.
  - Simplified encoding (no BER).
  - Independent from the SNMP version.

The Agent Extensibility (AgentX) Specification

- Processing entities: master-agent, sub-agents.
- Encoding of all data types in the AgentX protocol.
- 9 administrative PDUs:
  - open/close AgentX sessions,
  - register/unregister MIB regions with the master,
  - allocate/deallocate indices of tables,
  - add/remove agent capabilites in the sysORTable,
  - check state of the master.
- 9 management processing PDUs:
  Get, GetNext, GetBulk, Response, Notify,
  4 PDUs to handle Set requests by a 3-phase-commit protocol.
- Elements of procedure.
- Transport Mappings: TCP, UNIX-domain sockets.

Using Java for Sub-Agent Implementations — Why?

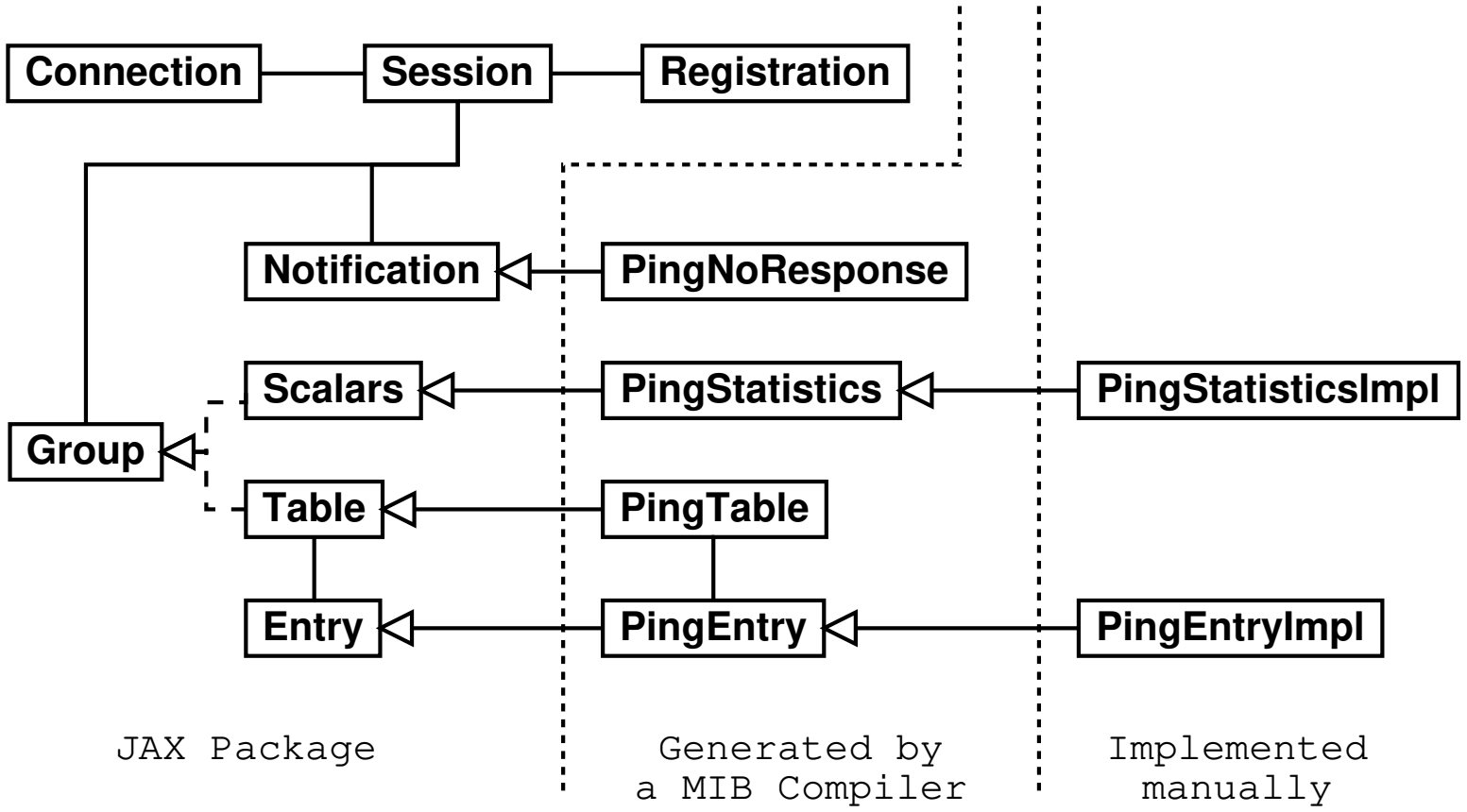
- Java allows rapid prototyping / low development costs.
- There are many criterions why people do ‘like’ Java
  (object oriented, platform independent, widely deployed)
- More and more applications are written in Java.
  People ask for management facilities for these applications.

→ Java AgentX Sub-Agent toolkit ‘JAX’

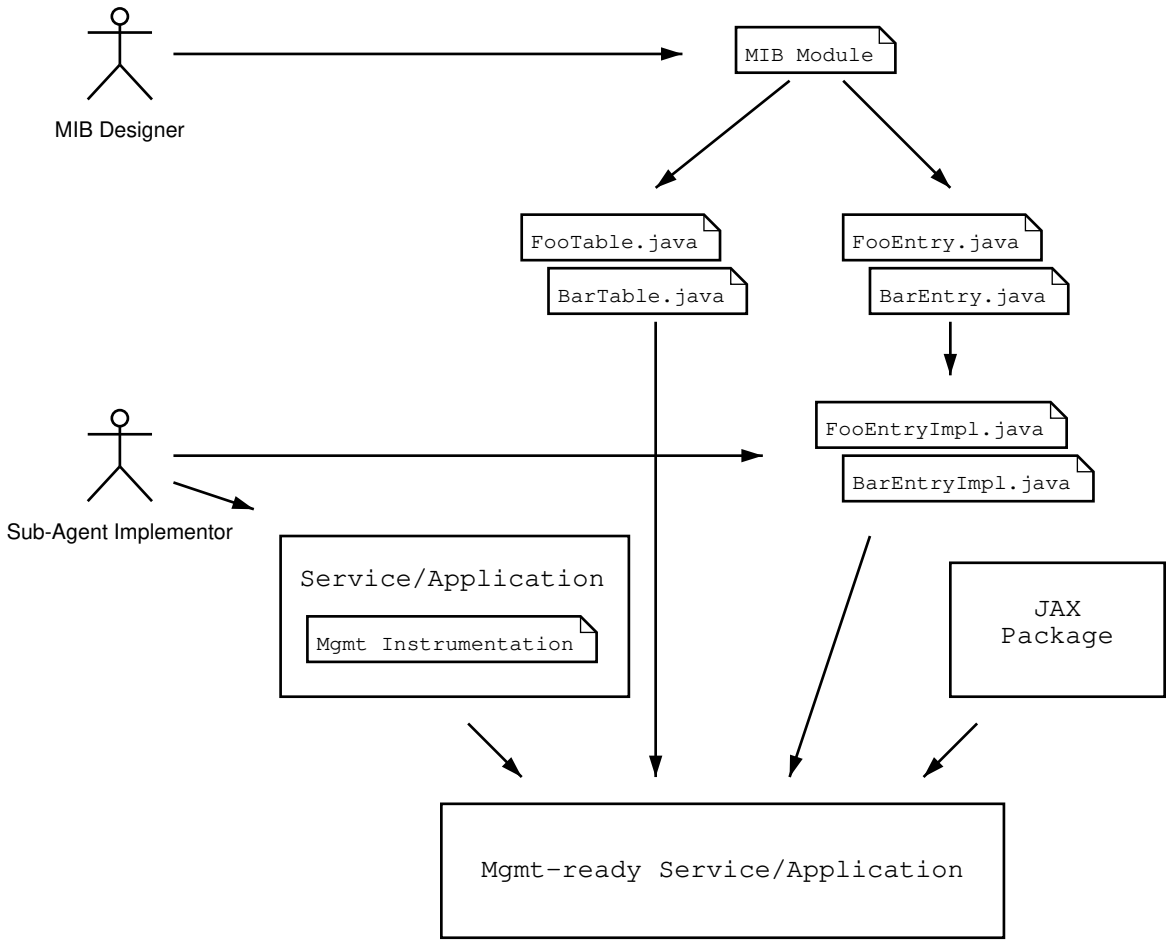
- A Java class package (`jax`) with classes representing
  - connections, sessions, registrations,
  - AgentX PDUs,
  - generic MIB information structures: scalar groups, tables, and notifications.

  Not yet: index allocation, agent capabilites registration
- A MIB compiler to generate high-level MIB-specific stub classes.

Simplified JAX Class Diagram:



JAX Sub-Agent Development Process:



Example: Adding WWW-MIB Support to the Jigsaw HTTP Server

Adding support for 5 major tables of the WWW-MIB to the W3C Jigsaw HTTP server required minimal individual coding:

- 20 lines added to existing Jigsaw code,
- 2 new files containing 250 lines of code, and
- a few lines filled in the 5 compiler generated table entry stub classes.

The MIB Compiler

- For JAX, we did not reinvent a new MIB compiler.
- The JAX MIB compiler is based on *libsmi*:
  - libsmi can parse multiple management information languages, (SMIv1/v2, SMIng).
  - libsmi has a C API to facilitate the retrieval of MIB module information.
  - Based on this API, libsmi includes a tool (smidump) that can dump MIB modules in various output formats ($\rightarrow$ MIB compiler).
  - smidump can easily be extended to support additional output formats.
  - The JAX MIB compiler is implemented as such a new output format.
- For a given MIB module, the compiler generates Java source files:
  - a class for every group of scalar objects,
  - a class for every table,
  - a class for every table entry (group of columnar objects),
  - a class for every notification or trap,
  - an associated stub class for every scalar group class,
  - an associated stub class for every table entry class.

Performance

Native monolithic agent	79 $\mu secs^*$
Native AgentX sub-agent	99 $\mu secs^*$
JAX sub-agent	120 $\mu secs^*$

* Average var-bind processing time within the master agent's request dispatcher.
The request reads a single octet string from the proc filesystem
on a 400 MHz PentiumII/Linux system.

Related Work

- Some SNMP agent toolkits support AgentX without requiring any changes to the existing SNMP agent API, e.g. UCD-SNMP (C), Agent++ (C++).
- The AdventNet SNMP toolkit supports a proprietary concept based on Java Beans to implement MIBs. It includes a MIB compiler.
- The Java Management Extensions (JMX) is also based on Java Beans, but in a more flexible way. It does not support AgentX, as well.
- J.AgentX (University of Coimbra, Portugal) contains a Java AgentX master and a Java sub-agent API. The API is relatively low-level. It does not contain a MIB compiler.

Open Issues / Outlook

- Table index allocation is not yet implemented.
- Registration of agent capabilities is not yet implemented.
- Checking the sanity of AgentX sessions through Ping PDUs is not yet implemented.

These issues could easily be solved without modifying the JAX architecture.
They just did not yet become important in our project.

Other interesting features would be:

- High level support for objects of well known types, e.g. generation of special code for RowStatus objects.
- Make the MIB compiler dependent on SMI compliance statements.

Conclusion

- JAX uses *current technology* (AgentX, Java) to allow *cost-efficient* development of *portable* MIB implementations.
- A *MIB compiler* limits coding work significantly to application specific pieces.
- JAX is *small and simple*, though
  it offers a high-level interface that *does not require SNMP experts*.
- JAX (and libsmi) is *open source* software.

Web Resources

- AgentX: `http://www.scguild.com/agentx/`
- JAX: `http://www.ibr.cs.tu-bs.de/projects/jasmin/jax.html`
- libsmi: `http://www.ibr.cs.tu-bs.de/projects/libsmi/`