# The Jasmin Script MIB Implementation and its Use for Policy-based Management

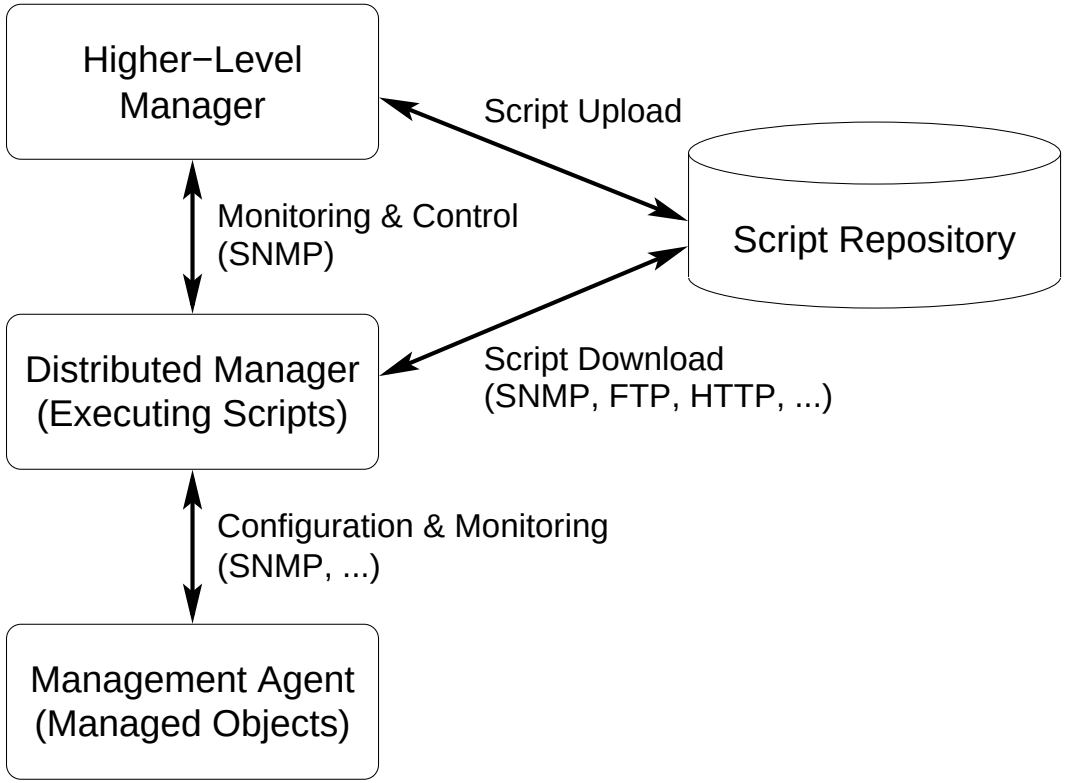
*Frank Strauß*
*Institute of Operating Systems and Computer Networks*
*Technical University Braunschweig*
*Mühlenpfordtstraße 23*
*38106 Braunschweig*
*Germany*
*strauss@ibr.cs.tu-bs.de*
*http://www.ibr.cs.tu-bs.de/users/strauss/*

1. The Script MIB
2. The Jasmin Project
3. Application for Policy-based Management

# The Script MIB

- Designed and standardized by the IETF Distributed Management (DISMAN) Working Group
- First Proposed Standard: RFC 2592, May 1999
- Updated Proposed Standard: RFC 3165, August 2001
- A MIB for the delegation of management functions based on the Internet management framework:
  - Transfer of management scripts to a distributed manager (push and pull model),
  - Initiating, suspending, resuming and terminating management scripts,
  - Accessing results of running and terminated management scripts.
- Security based on
  - SNMPv3 security (USM and VACM)
  - Script runtime engine security models (sandbox)
- There six tables:
  - `smLangTable` and `smExtsnTable`: supported script languages and language extensions
  - `smScriptTable` and optional `smCodeTable`: scripts known to the agent
  - `smLaunchTable`: characteristics to start a script and control its lifetime
  - `smRunTable`: ‘process table’ with some additional object to control ‘processes’ and represent results

# The Distributed Management by Delegation (MbD) Architecture



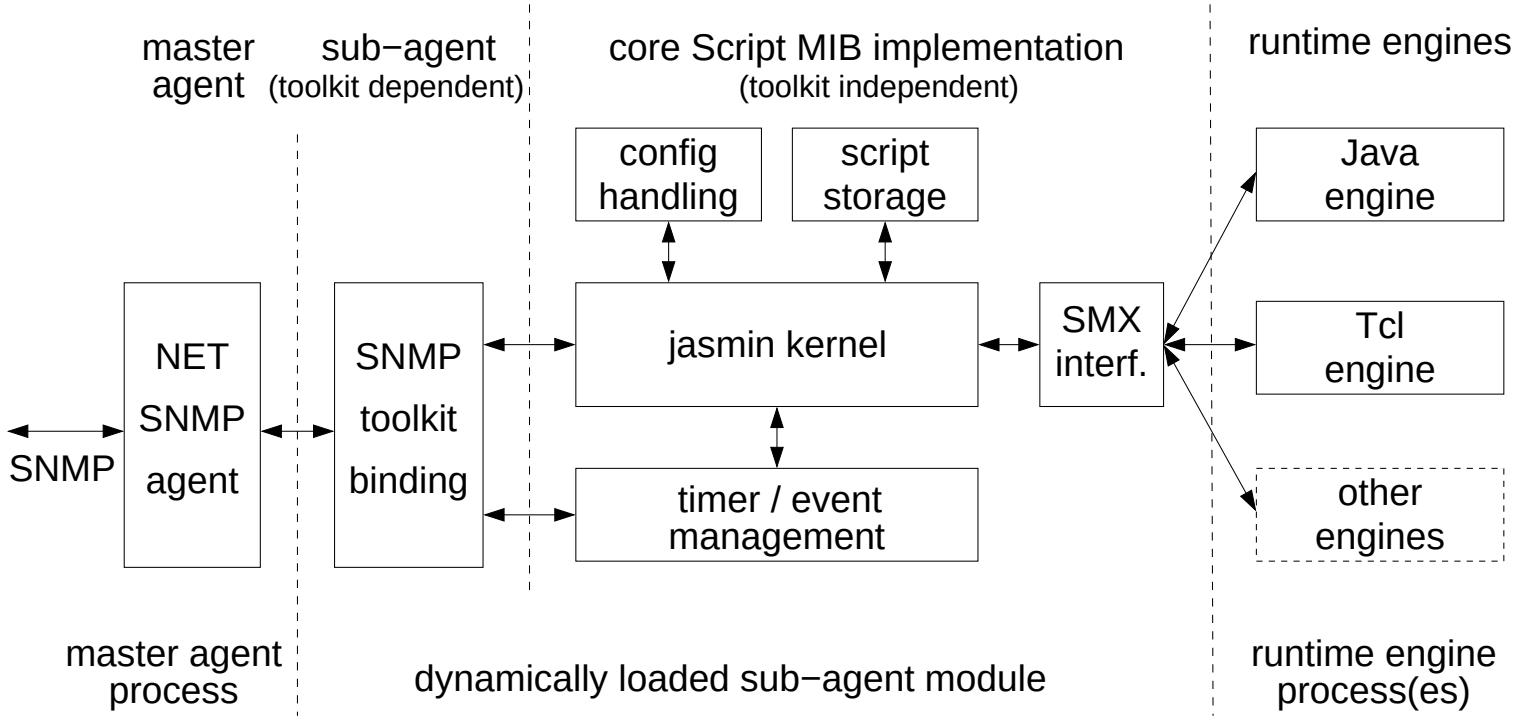
# The Jasmin Project

- A joint project:
  - Technical University of Braunschweig
  - Network Laboratories, NEC Europe Ltd.
- Goals: Evaluate and enhance the Script MIB Standard
  by providing an implementation and studying use-cases
- Developed several open source (GPLed) software components
- Contributions to the IETF DISMAN Working Group
- Various conference and journal publications
- Raised significant interest in our prototype implementations,
  primarily for interoperability tests and educational purposes
- Project members:

  @NEC: Marcus Brunner, Cornelia Kappler, Paloma Martinez, Jürgen Quittek,
  Thiemo Schwarz, Raghuveer Singh (and others?)

  @IBR: Matthias Bolz, Sven Brandenburg, Torsten Klie, Sven Mertens,
  Jürgen Schönwälder, Frank Strauß

# The Jasmin Script MIB Agent Implementation



# A Schedule MIB Implementation (RFC 2591)

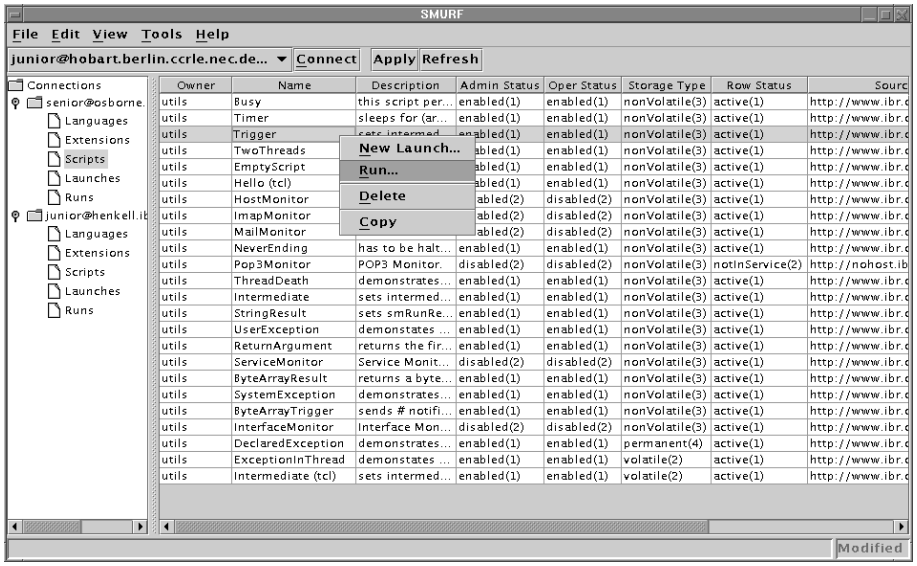
- Also based the NET-SNMP agent.

# The `disman` Java Package

- A high level API to manage Script MIB and Schedule MIB objects in an OO-fashion.

# Smurf

- A human friendly GUI application
- Allows to manage Script MIB and Schedule MIB agents
- Based on the `disman` package

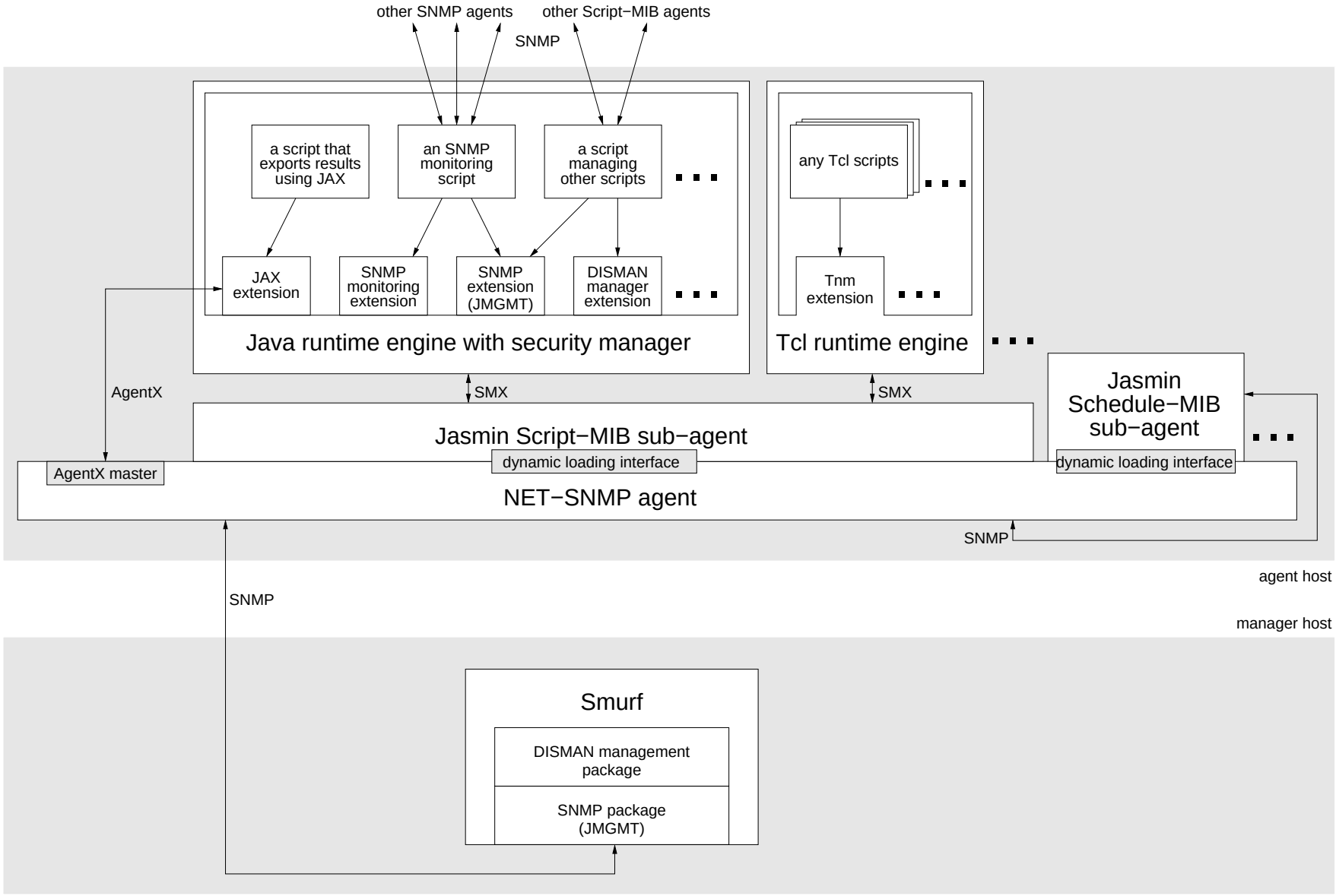


# JAX

- A Java toolkit for high-level AgentX (RFC 2741) sub-agent development
- Components:
  - a class package for the core AgentX sub-agent functions
  - a MIB compiler (based on `libsmi`) to generate Java stub and skeleton classes from MIB definitions.
- Example: A prototype implementation of the 5 core tables of the WWW-MIB for the W3C Jigsaw HTTP server took just 20 lines of code added to existing Jigsaw code, approx. 250 lines of two new classes, and a few lines filled into the generated skeleton classes.

# Java Monitoring Scripts

- A set of Java scripts for some distributed monitoring functions, e.g.
  - interface and process load monitors,
  - SMTP, HTTP, FTP, POP3, NNTP service monitors,
  - TCP connection monitors,
  - Mail server monitors, etc.
- Based on core classes for monitor initialization and scheduling

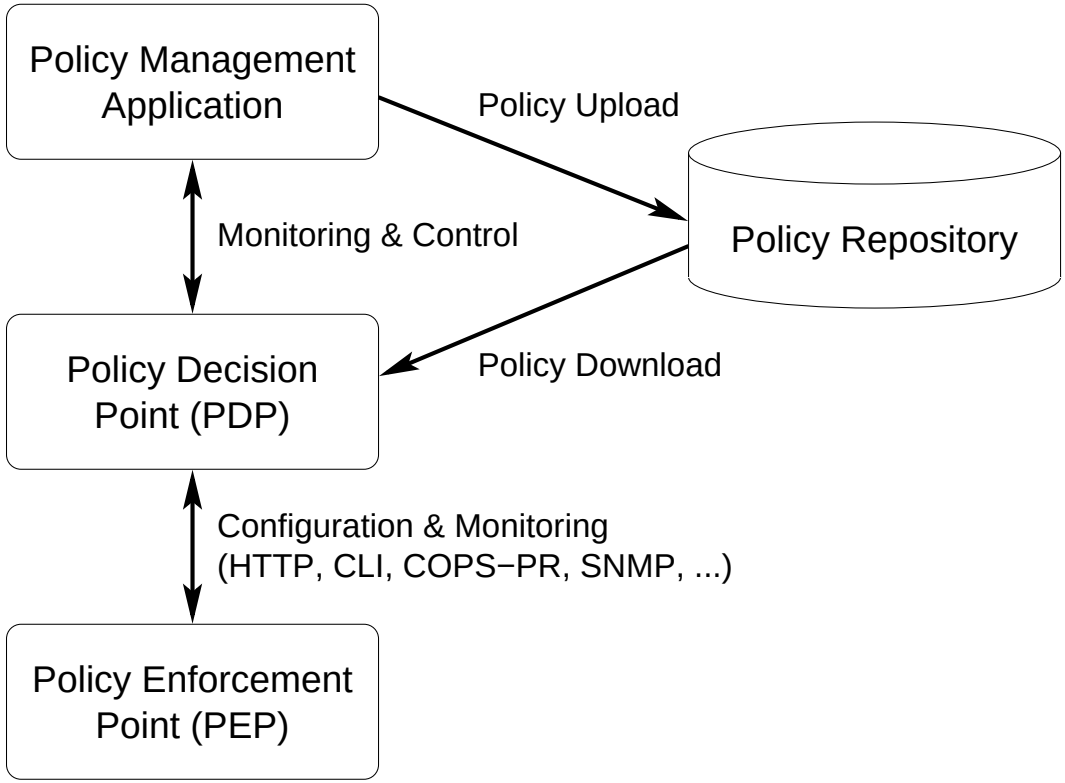

other SNMP agents
other Script−MIB agents
SNMP
a script that exports results using JAX
an SNMP monitoring script
a script managing other scripts
any Tcl scripts
JAX extension
SNMP monitoring extension
SNMP extension (JMGMT)
DISMAN manager extension
Tnm extension
Java runtime engine with security manager
Tcl runtime engine
AgentX
SMX
SMX
Jasmin Schedule−MIB sub−agent
Jasmin Script−MIB sub−agent
dynamic loading interface
dynamic loading interface
AgentX master
NET−SNMP agent
SNMP
agent host
SNMP
manager host
Smurf
DISMAN management package
SNMP package (JMGMT)

# Policy-based Management

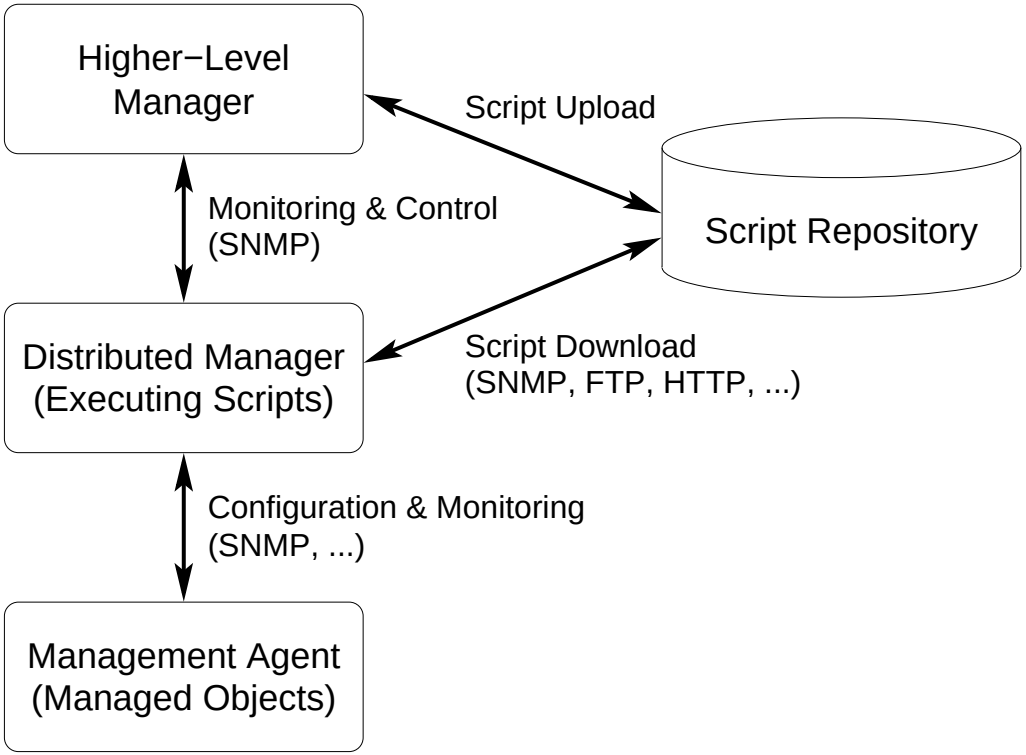
- Motivation and general concepts:
  - The traditional management of individual device-specific configurations is complex and error-prone.
  - However, the general policies behind those configurations are almost always relatively simple.
  - → Let the administrator manage just those policies, and
  - → use automagic to apply them to the individual devices.
  - Common approach:
    A *Policy* represents a number of *Rules*, where each rule is triggered by an *Event* and consists of an *Action* if a *Condition* is evaluated to true:

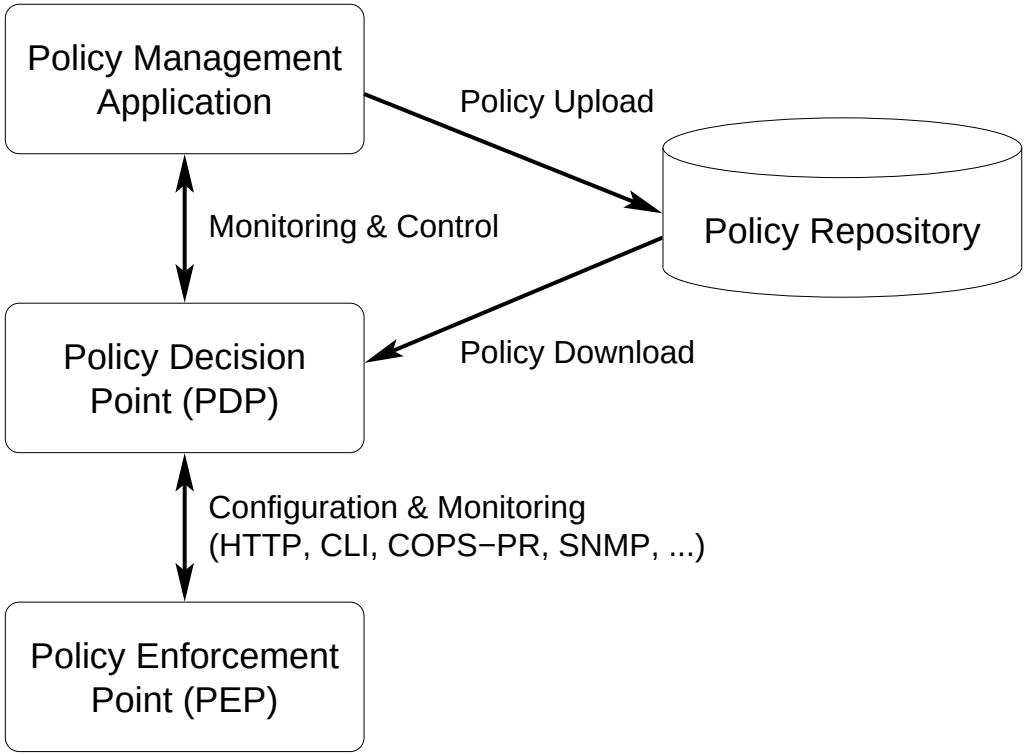
    `on` <*event(s)*> `if` <*condition*> `do` <*action(s)*>
- There are several approaches to express policies:
  - A traditional programming language + a language extension for policies
  - A specific policy definition language, e.g. PONDER
  - The Policy Core Information Model (PCIM) – an extension to the IETF/DMTF Core Information Model (CIM)

# The Policy-based Management (PBM) Architecture

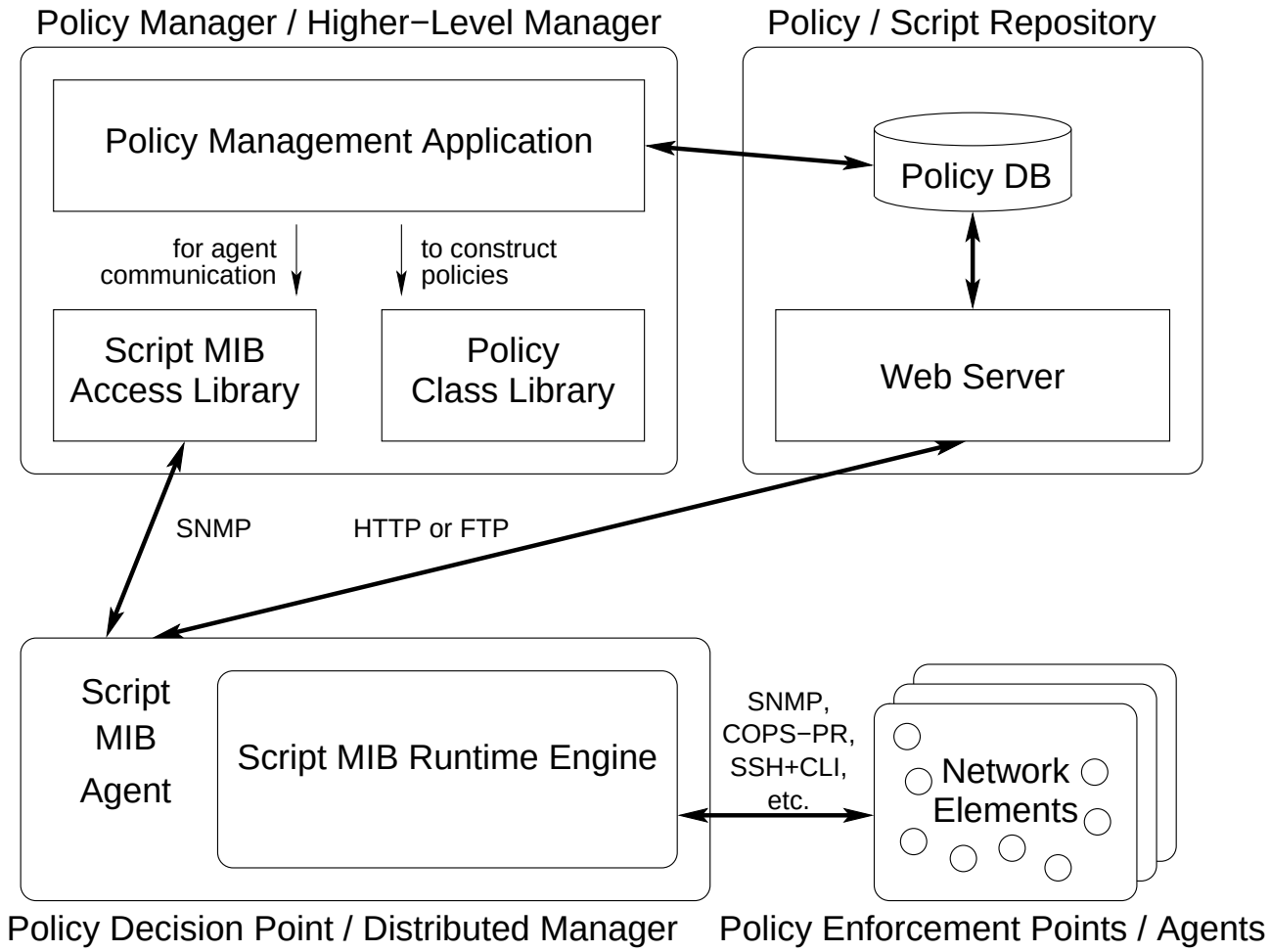


# MbD vs. PBM

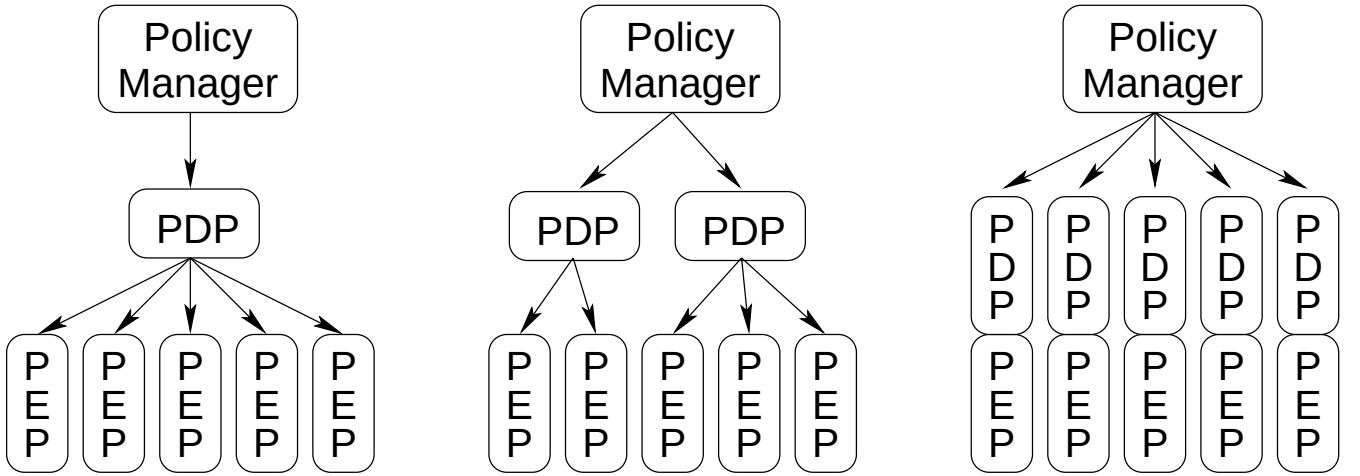

Higher−Level Manager
Script Upload
Monitoring & Control (SNMP)
Script Repository
Distributed Manager (Executing Scripts)
Script Download (SNMP, FTP, HTTP, ...)
Configuration & Monitoring (SNMP, ...)
Management Agent (Managed Objects)


Policy Management Application
Policy Upload
Monitoring & Control
Policy Repository
Policy Decision Point (PDP)
Policy Download
Configuration & Monitoring (HTTP, CLI, COPS−PR, SNMP, ...)
Policy Enforcement Point (PEP)

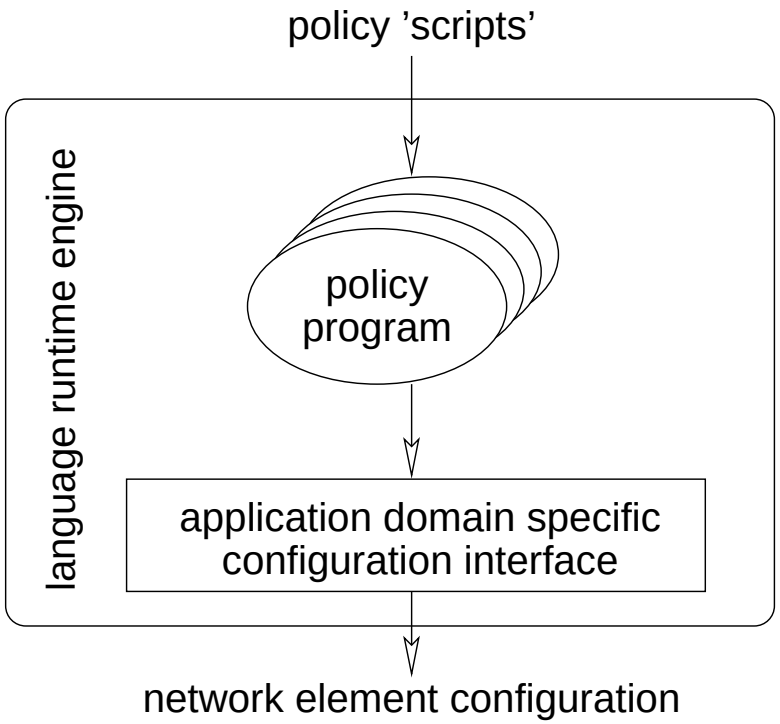
# Architecture of the Script MIB based PBM System



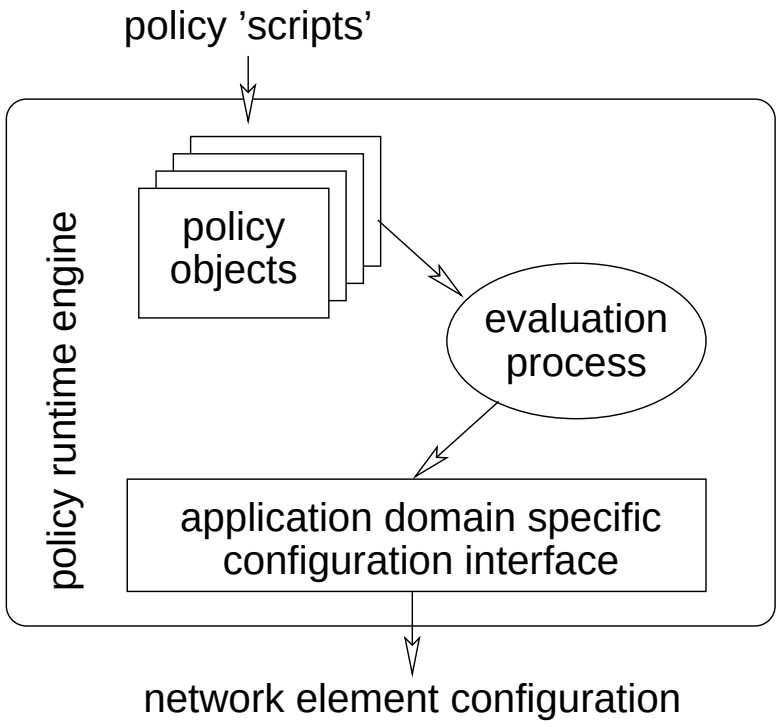
# Different Levels of PDP Distribution



# We Have Implemented Two Approaches

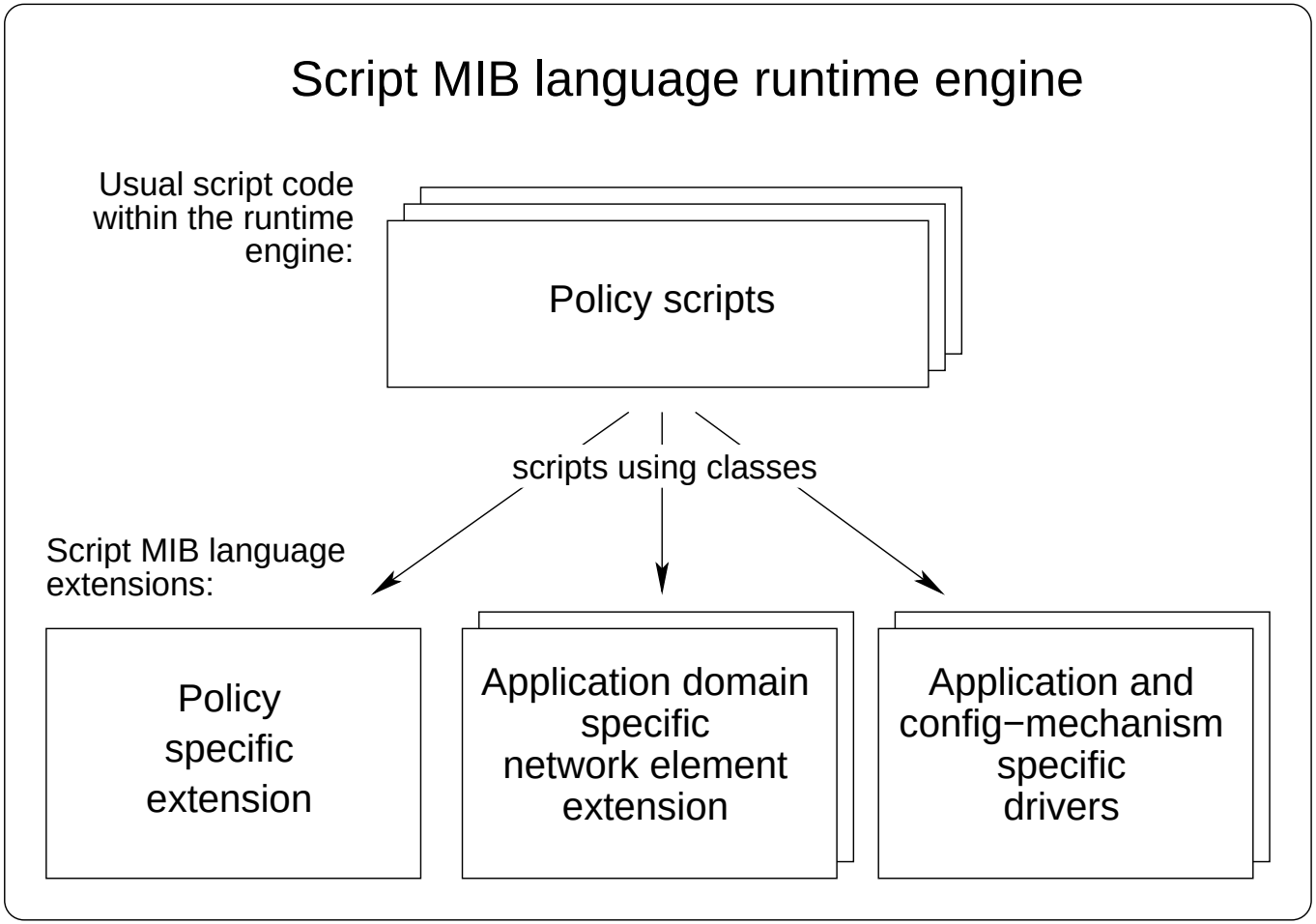



(a) policies as programs

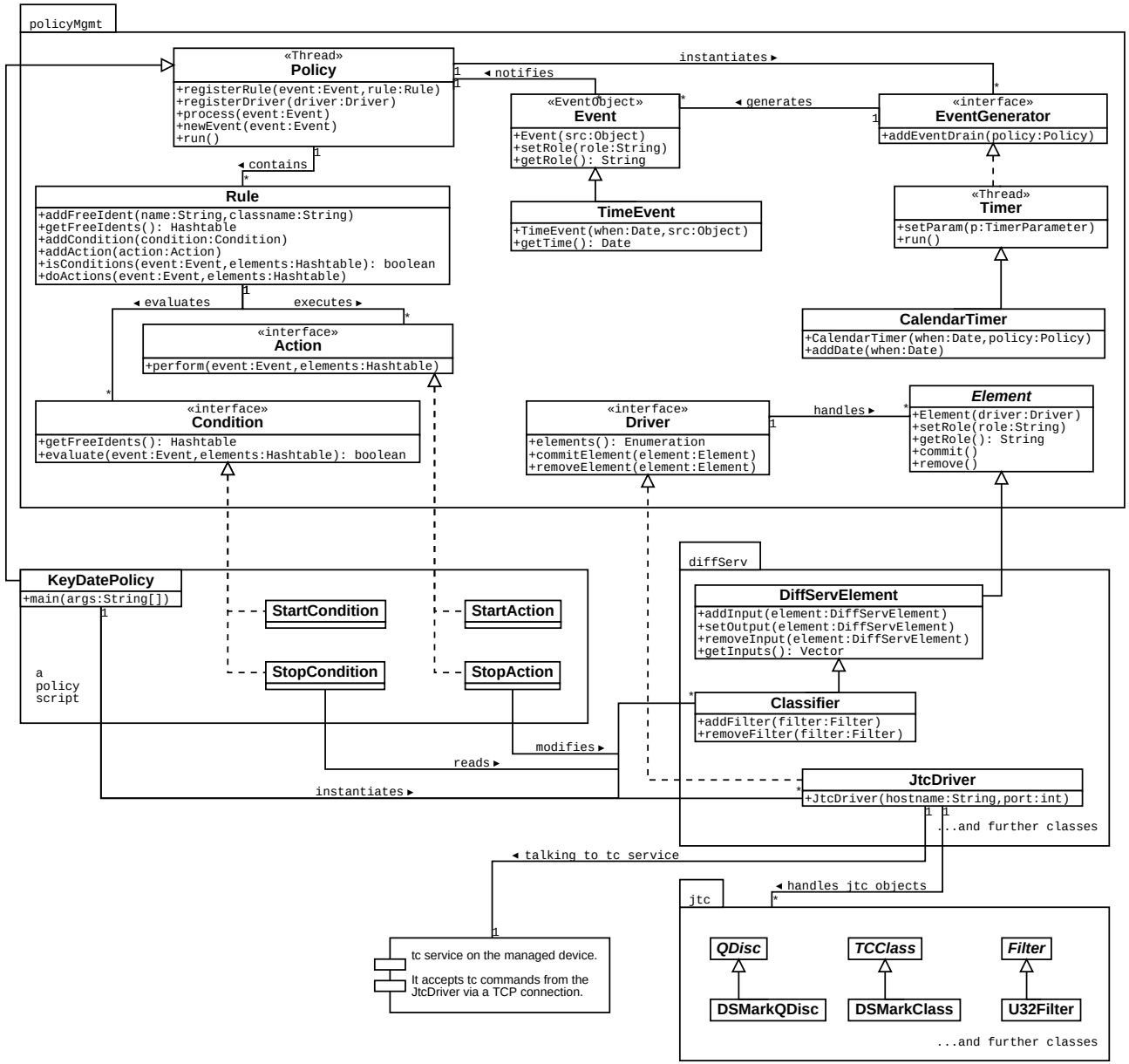



(b) policies as objects

# Scripts Using Policy-specific Language Extensions



# Classes for Policy-based DiffServ Configuration Management



# Conclusions

- The Script MIB has been designed for the Management-by-Delegation Paradigm.
- Tools on top of the Script MIB can make work with it quite trivial.
- Releasing the developed software under the GPL turned out to be the right decision.
- Using the Script MIB for Policy-based Management is an obvious and straight-forward alternative.
- No need to re-invent things like
  - a PDP internal architecture
  - a protocol to transfer policies to the PDP
  - a PDP-PEP protocol
  - a security model
- Depending on the chosen approach, it can be
  - *cheap*, by using the existing Script MIB and runtime infracstructure, while policy scripts become more complex,
  - *standards based*, by applying the PCIM and using a special policy runtime engine,
  - *user friendly*, by using a policy definition language (not implemented by us).

Project Web Page: http://www.ibr.cs.tu-bs.de/projects/jasmin/

Thanks!

Q & A