Status of the Proxy Printer Provider Prototype

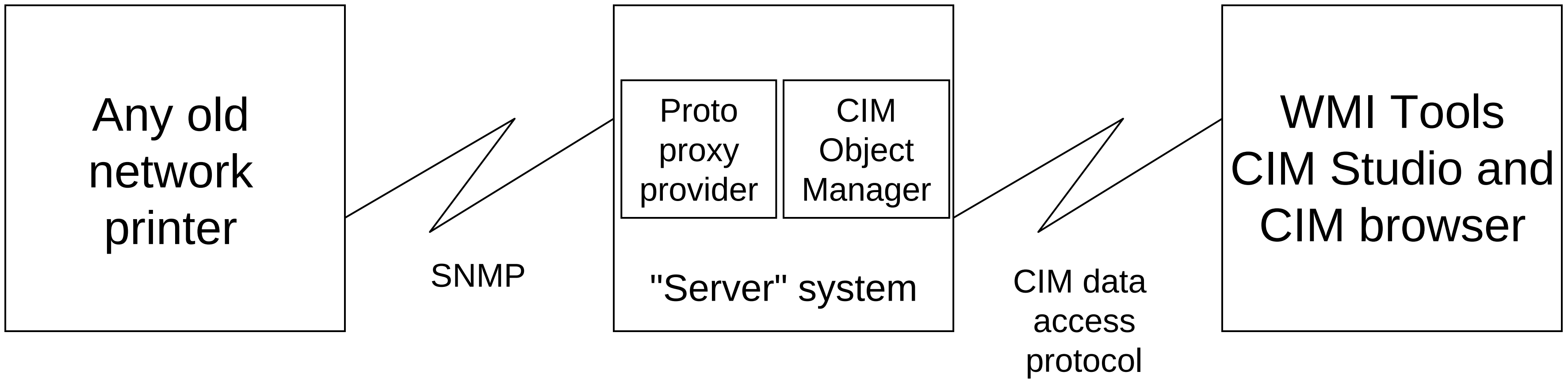
PWG WIMS-CIM Working Group

Rick Landau
Dell, CTO Office

2008/10/21 v0.2

Prototype Being Constructed

- Network printer: whatever is at the end of the aisle
- "Server" system: Windows box of some sort
- Management client: el cheapo CIM data browser application, not printer-specific, not suitable for actual management
  - Probably local on server system



What is a CIM Provider?

- A provider implements one or more classes under the CIMOM
  - Only one provider permitted to implement a class (in a namespace)
- A mapping layer between a driver and a data repository
  - Maps from physical representation to logical representation
    - Physical = hardware, driver, device-dependent
    - Logical = device-independent model, data represented using CIM schema standard classes
- Façade only
  - Passive code, called when needed, no active business logic, no active management, translation interface only

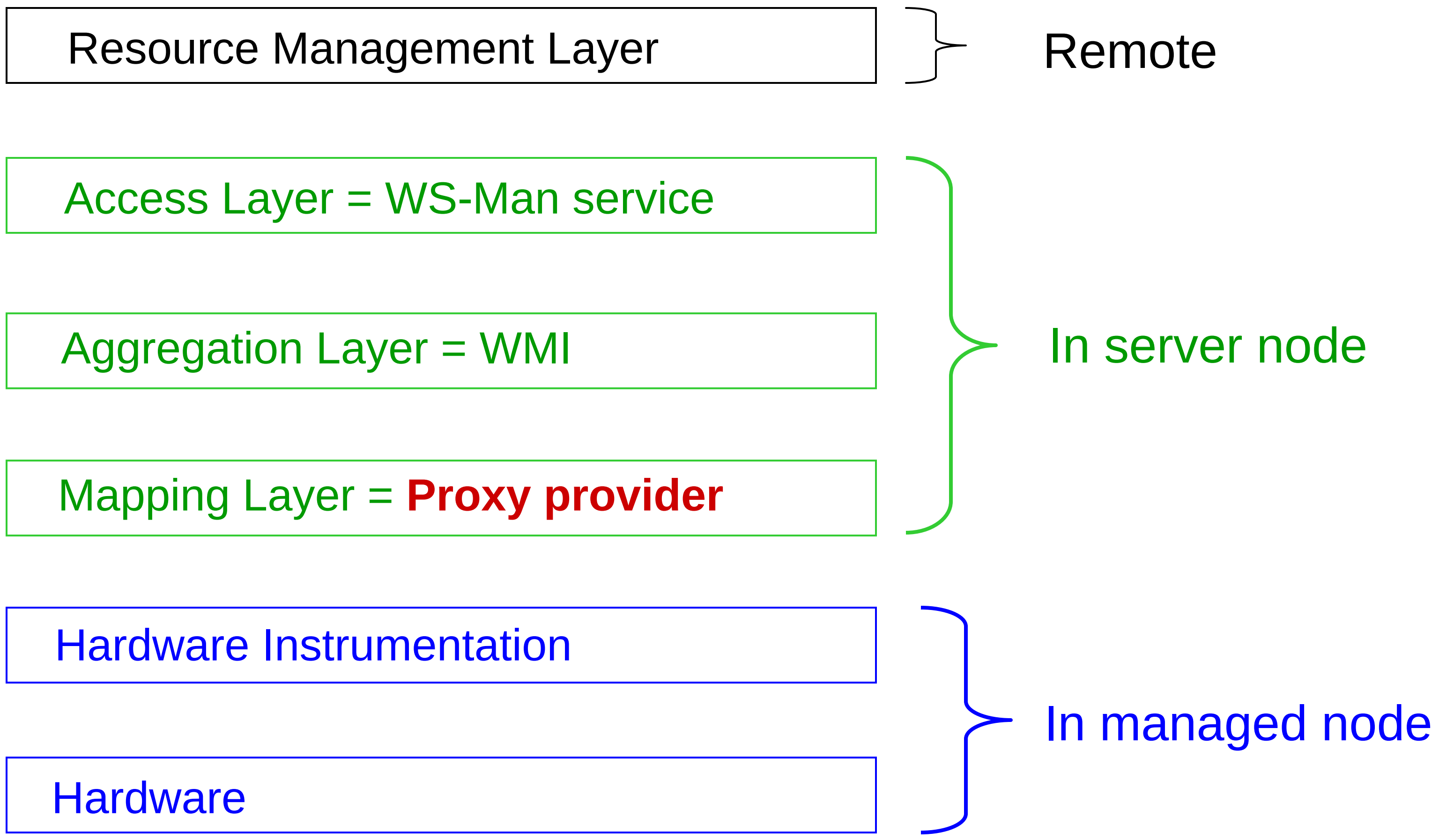
Doing a CIM Proxy Provider

- What is *this* CIM provider?
  - This provider implements a number of the new printer classes
- What's a proxy provider? An SNMP-to-CIM proxy agent
  - Reads SNMP data from a network printer
  - Republishes that data in CIM format
- Management applications using WBEM protocols can then access the data in CIM schema format
  - WS-Management protocol in particular
  - However, I'm not building an application as part of the prototype

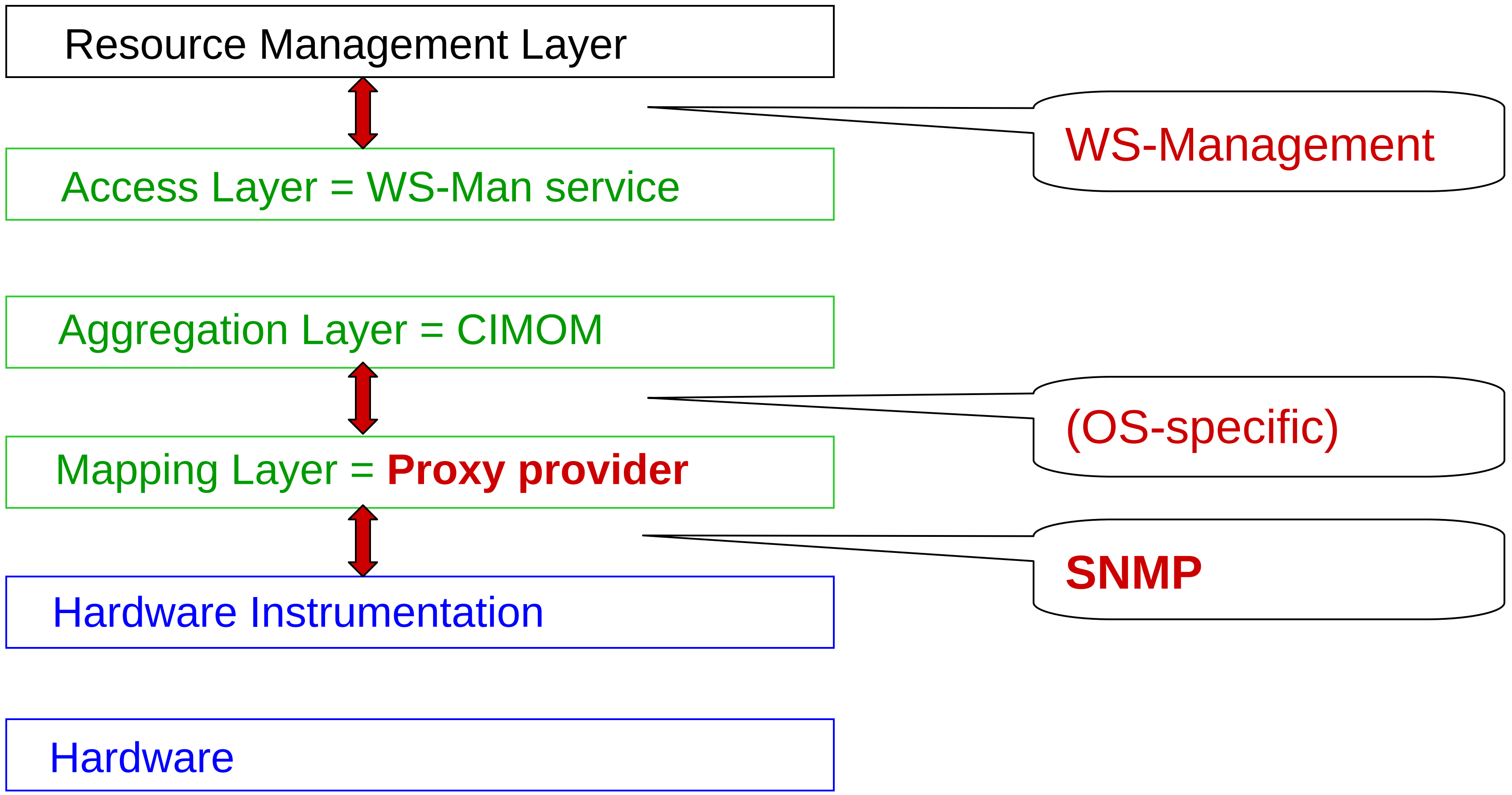
Structure of Provider

- Provider statically registers definitions of classes to be instantiated in what namespaces
  - CIMOM uses this to determine what provider to call for what data requests: instances (of classes) in CIM namespaces
- WBEM protocols are all "pull" model
  - Data doesn't have to exist until someone asks for it, never stale
  - Data isn't stored, it's "instantiated" when requested
  - Fewer db race conditions (locking being addressed just now)
- Uses an interface up to CIMOM
- Uses an interface down to "driver(s)"
- The cheese in the sandwich translates the data

Management Stack - Proxy Provider



Management Stack - Protocols Between Layers


Resource Management Layer
Access Layer = WS-Man service
Aggregation Layer = CIMOM
Mapping Layer = **Proxy provider**
Hardware Instrumentation
Hardware
WS-Management
(OS-specific)
SNMP

What to Implement

- Substantive classes
- Association classes
- Properties
- Operations

Printer Classes

- (All begin with CIM_Print)
- InputTray
- OutputTray
- MediaPath
- Marker
- Channel
- Interpreter
- Finisher
- Interlock
- AlertRecord

- (Printer-specific Associations)
- PrinterComponent
- AssociatedPrintSupply
- AssociatedPrintInterpreter

- (Other CIM Associations)
- ConcreteComponent
- Dependency
- UseOfLog
- LogManagesRecord

Classes Currently Implemented

- InputTray (almost complete)
- OutputTray
- Interlock (almost complete)
- MediaPath
- Marker
- Supply
- Channel
- Interpreter
- AlertRecord

Properties Implemented

- Important structural properties included
  - InstanceID
  - SNMPRowID
  - Name
- Easy mappings
  - Direct mapping: CIM property value = SNMP variable value
  - Constant value
  - Null value
- Algorithmic mappings
  - XxxBasis
  - AvailabilityStatus
  - PrimaryStatus
  - CriticalAlertsPresent
  - NoncriticalAlertsPresent
- Selected others

Operations Implemented

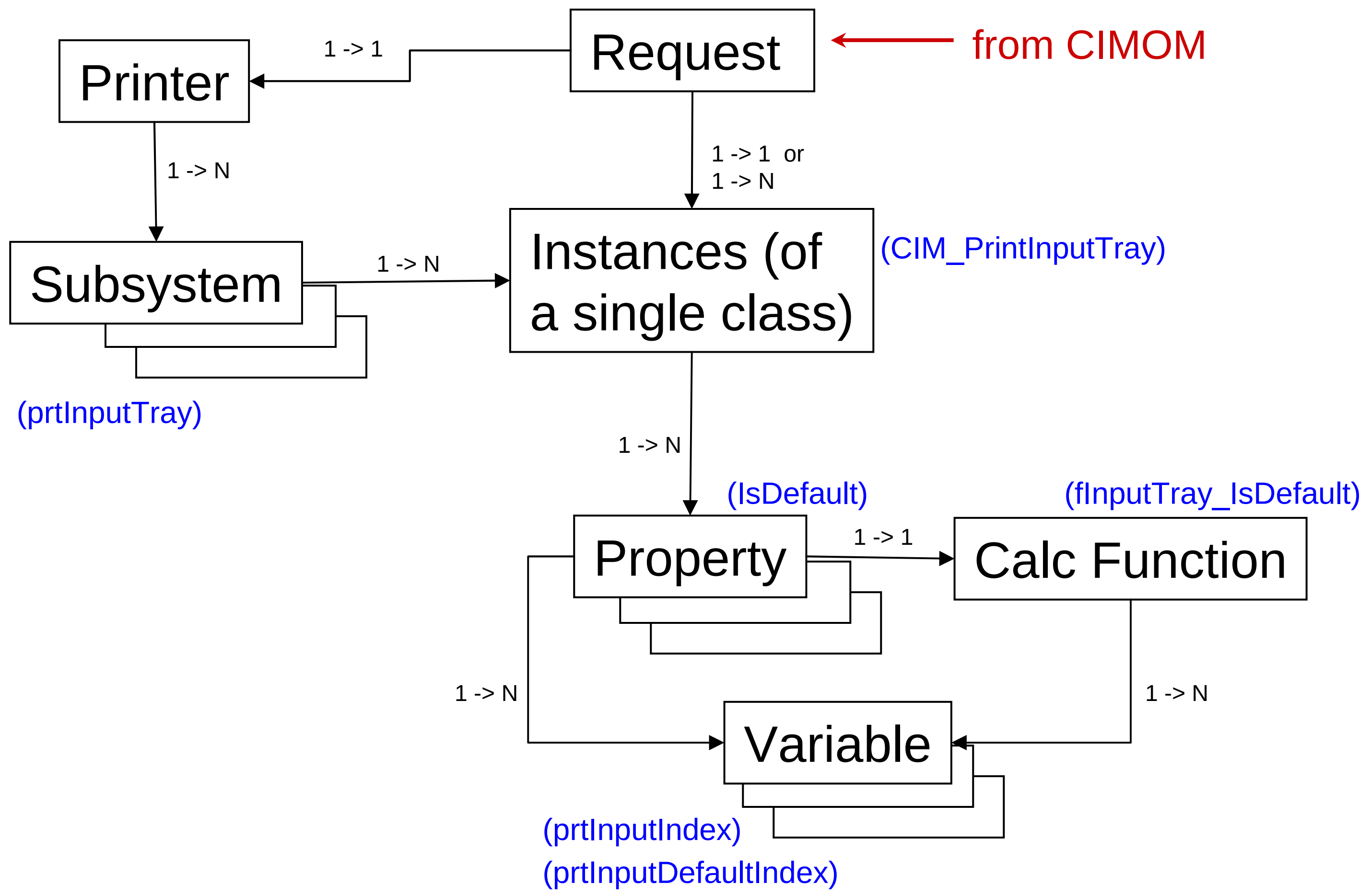
- Get Instance
  - Call Get(string className, string keyName[ ], string keyValue[ ])
  - Return all class properties
    - propertyName=propertyValue;…
- Enumerate Instances
  - Call Enumerate(string className)
  - Return all values of all class instances
    - instance1;propertyName=propertyValue;…
      instance2;propertyName=propertyValue;…
- All output in XML eventually

Structure of Provider

- Operation interface: Get, Enumerate
- Subsystem own class instances
- Class instance owns properties
- Property owns variables
- Variable retrieved by SNMP interface
- Property owns calculation function

- Many runtime dictionaries to guide program flow
  - Initialization phase turns ini files into runtime tables
- Initialization also finds all instances of subsystems, e.g., how many input trays

Class Relationships



Ini Files

- CIM classname -> list of CIM properties
- CIM property name -> list of SNMP variables
- SNMP variable name -> OID
- CIM classname -> SNMP table of instances
- CIM classname -> SNMP subscript structure
- SNMP variable name -> SNMP subscript structure (exceptions)
- CIM property name -> name of function to calculate CIM value from SNMP values
- Discovered printers: name, IP

Cutting Corners

- SNMP interface: Get and GetNext
  - Reads variable values from MIB dump files
  - Numeric OID string -> string value or error
- Output in debug form
  - Flat-file output: className propertyName propertyValue
- One printer at a time
  - No discovery of network printers
  - Manual input from ini file
  - Not thread-safe
- Hey, it's a p-r-o-t-o-t-y-p-e

To Do

- Enum mappings
  - Direct mappings already done
- Many, many properties
  - Including many inherited from parent
- Association classes
- Real interface to the Dell CIMOM interface ("secret sauce")
- Localhost IP communication between CIMOM and daemon

Look at Some Files

- ini files
- Calculation functions

Questions?