NETCONF (draft) Protocol Overview

Phil Shafer
Juniper Networks
phil@juniper.net

Design Team (in no particular order)

Andy Bierman <abierman@cisco.com>
Eliot Lear <lear@cisco.com>
David Perkins <dperkins@dsperkins.com>
Ted Goddard <ted.goddard@windriver.com>
Phil Shafer <phil@juniper.net>
Rob Enns <rpe@juniper.net>
Ken Crozier <kcrozier@cisco.com>
Steve Waldbusser <waldbusser@nextbeacon.com>
Margaret Wasserman <mrw@windriver.com>

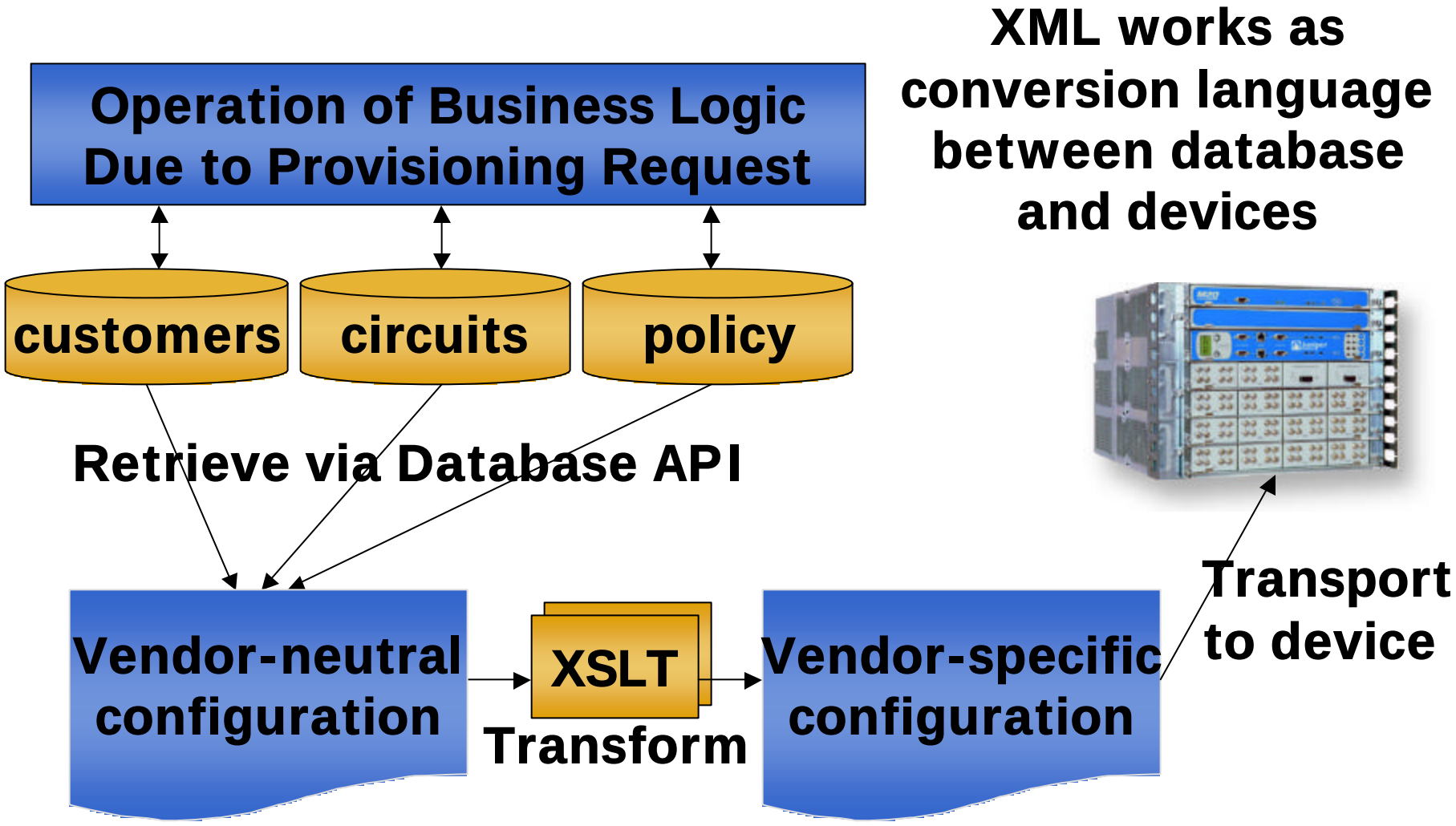
Strategy

- **Allow implementation to mirror native capabilities of device**
  - **Text-based technology such as XML permits tight integration with CLI**
  - **No feature lag between NETCONF and CLI**

So why XML?

- XML is all about data portability and data malleability
- Commercial databases support XML APIs
  - XML is the lingua franca of data exchange
    - Not just network configuration
- Transformation (via XSLT) of the data is well supported, easy to write and maintain
- XML well-suited to expressing configuration hierarchy in complex devices
  - Can handle non-hierarchical configuration too

The Ideal World



XMLCONF Protocol Features

- Channels
- Simple RPC Mechanism
- Capabilities
- Base Operation

Session Management: Channels

- **Management channel**
  - **Session control; creation of other channels**
  - **Abort command kills current command on the operations channel**
  - **Kill-session used to terminate the session of another user**
- **Operations channel**
  - **Used for RPC requests and replies**
- **Notification channel**
  - **Optional channel for asynchronous messages**

RPC Mechanism

<rpc>

- **Request on operations channel**

<rpc-reply>

- **Reply sent on operations channel**

<rpc-progress>

- **Provides progress reports (percentage completion) for long duration RPC operations, sent on the management channel**

<rpc-abort>

- **Provides a way to abort an RPC in progress, or queued for processing, sent on the management channel**

<rpc-abort-reply>

- **Abort RPC reply, sent on the management channel**

Capabilities Exchange

- **Each peer sends capabilities summary during session startup**
- **Capability is used only if both parties advertise the same version of the same capability**
- **Capability expressed as a URI**

```
<capability>http:/ietf.org/xmlconf/1.0/base</capability>
<capability>http:/ietf.org/xmlconf/1.0/base#notifications</capability>
<capability>http:/ietf.org/xmlconf/1.0/base#candidate</capability>
<capability>http:/ietf.org/xmlconf/1.0/base#lock</capability>
```

- **At least 1 version of the base protocol must be advertised**
- **Vendor specific capabilities may also be advertised**

```
<capability>http:/example.com/xmlconf/1.0/extensions</capability>
```

Initial Capabilities List

Base protocol

- Set, Get and Copy configuration commands
- Get system state information

Optional Capabilities

- Notification channel supported
- Separate candidate configuration
- Lock configuration for exclusive writing
- Candidate configuration (commit and rollback)
- Configuration Validation
- Separate startup (NV-stored) configuration
- Named Configurations can be stored on the device

Operational Model: Datastores

candidate

- Collects changes that are applied all at once to the running config
- Exists only if candidate capability is supported

running

- Current device configuration; changes to this config take effect immediately
- Exists on all devices

startup

- Configuration to apply to device upon next reboot
- Exists only if distinct startup capability is supported

Thank you.....