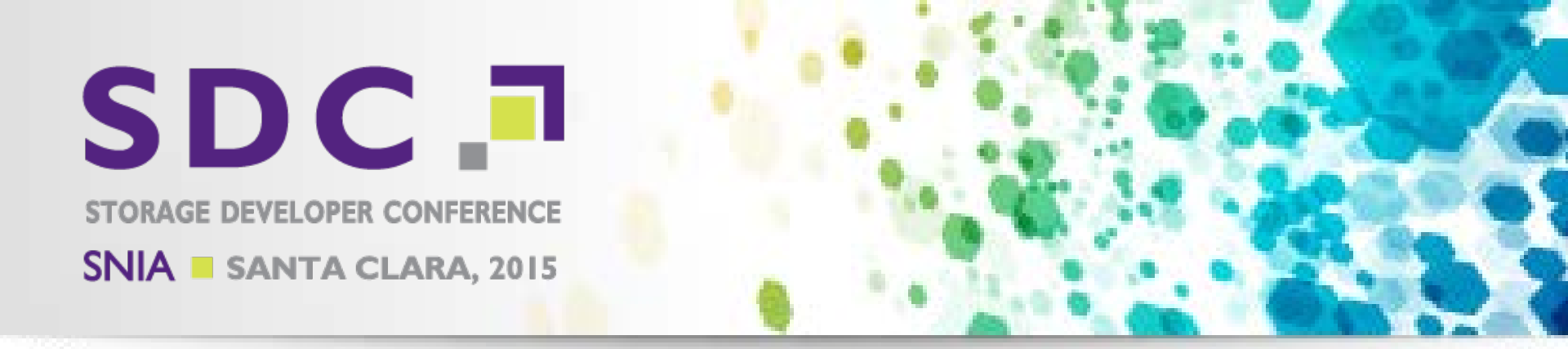


Manila File Sharing Services – Driver and Back-end Integration

Vinod Eswaraprasad

Principal Architect,

Wipro Technologies

Agenda : Driver for Manila Services

1. File Sharing Services on Cloud
2. OpenStack and Manila
3. Manila Architecture
4. Manila Driver – Deep-Dive
5. Getting it Right – Template Driven
6. Future – Towards SDS Controller

File Services – On Cloud

65%
2012

65% of Storage Sold is used For file based use cases
- IDC , 2012

NAS storage market to grow at a CAGR of 26% over the period 2014-2019, - Research and Markets

26%
2014

Increasing relevance of file based storage services

NAS is basically a personal cloud … !

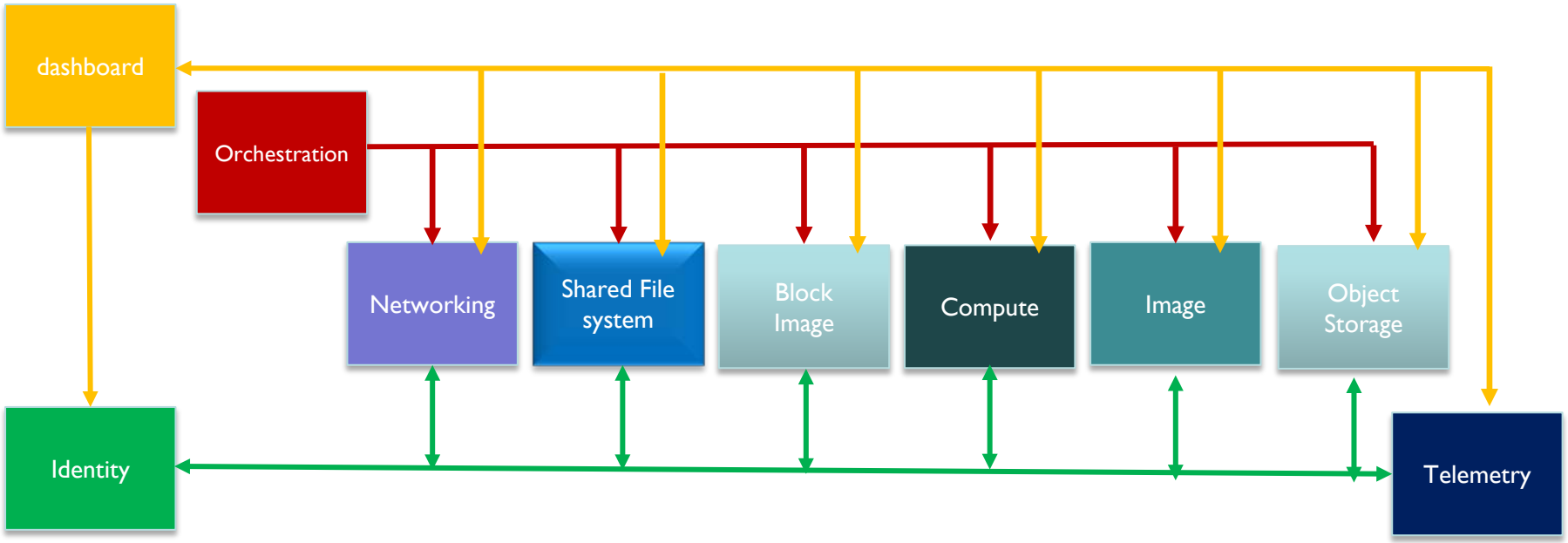
Cloud Shared File Services

- Cloud File Services
  - AWS – EFS (NFSv4)
  - Azure – Azure Files (CIFS/SMB 2.1)
  - OpenStack - Manila
- Access data through
  - Standard File operation APIs
  - Rest interface
- File Services Use cases
    - Home Directory services
    - Content Repositories
    - Development Environments
    - BigData-Applications

Manila – The OpenStack Shared File Services

- Open Standard APIs
  - File System Provisioning
  - Life cycle Management
- Shared File system service is not limited to OpenStack
  - OpenStack
  - Standalone
- SDS controller for file services
  - Vendor neutral
  - Extensible

Manila – Within OpenStack



OpenStack – Generic Service Architecture

- Message based
  - AMQP Messaging based
- Loosely coupled
  - De-coupled, and asynchronous
  - Publish/subscribe
  - Topic-Based Exchanges
    - RPC.call
    - RPC.cast
    - Topic, Topic-Host, Direct

Manila – Key Concepts

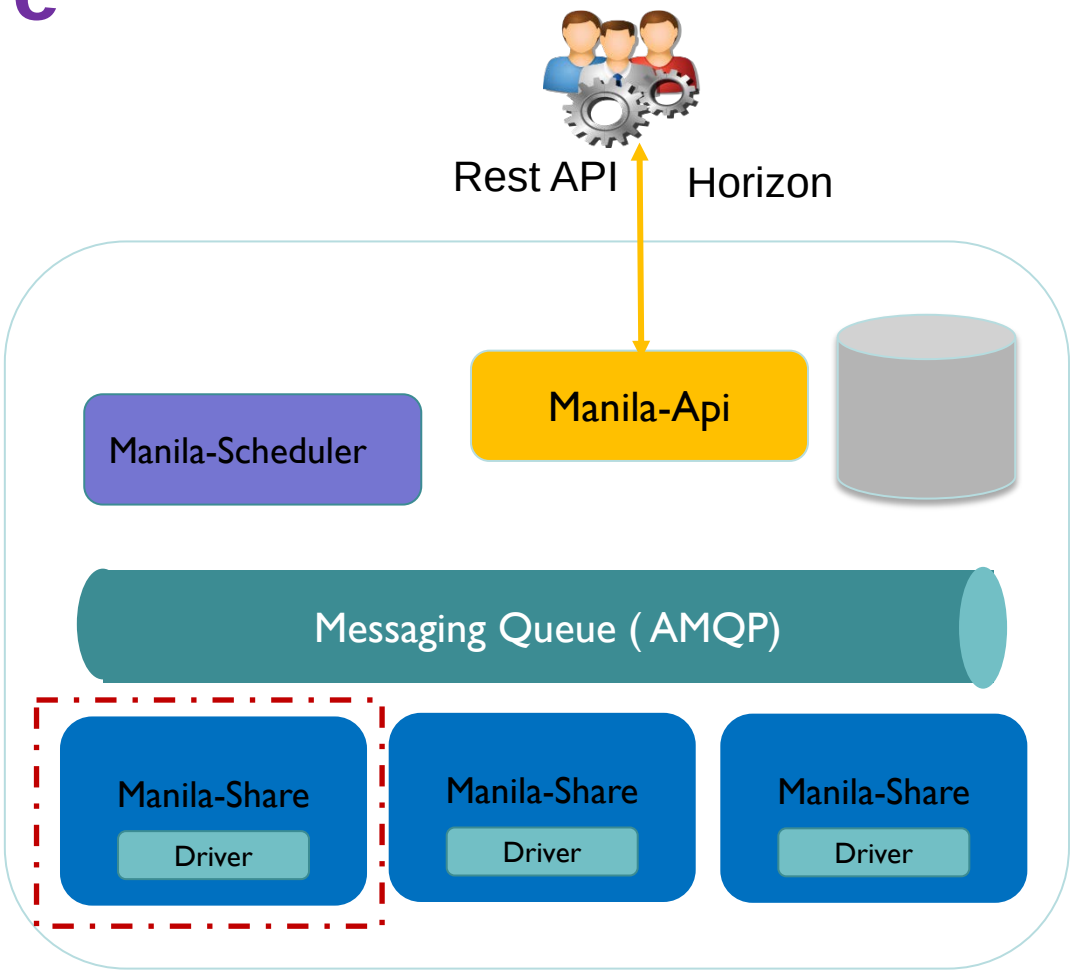
- Share – Instance of Shared File System
  - User specifies size, access protocol, share type
  - Accessible by multiple instances
- ACL
  - Defines client access to shares
  - Specified in IP or CIDR notation
- Network definition
  - Specify the neutron network and subnet through which the share is accessed
- Security Services
  - Client access rules for authentication (LDAP, AD, Kerberos)
- Snapshots
  - Read-only share copy

Manila – Key Concepts

- Backend
  - Provider of share – one per share
- Driver
  - Vendor/Technology specific implementation of the backend
- Pools
  - one or more logical storage resource pools
- Share Types
  - characterize Manila shares (visible to clients)
  - List of key value pair
- Extra_specs
  - key/value pairs associated with share_types

Manila - Architecture

- Manila-api
  - Exposes Rest API through WSGI application
- Manila-Scheduler
  - Makes provisioning decision based on share requests
- Manila-Share
  - Manager Process
  - Open Process per backend
- Driver
  - Vendor specific
  - Backend connection



Not in the data path – only control path

Basic Operations

Operation	CLI command	REST API
Create share	manila create	POST /shares
Delete share	manila delete <id>	DELETE /shares/{id}
List shares	manila list	GET /shares
Show share details	manila show <id>	GET /shares/{id}
Rename share	manila rename	PUT /shares/{id}
Edit share metadata	manila metadata	PUT /shares/{id}/metadata
Show share metadata	manila metadata-show	POST /shares/{id}/metadata

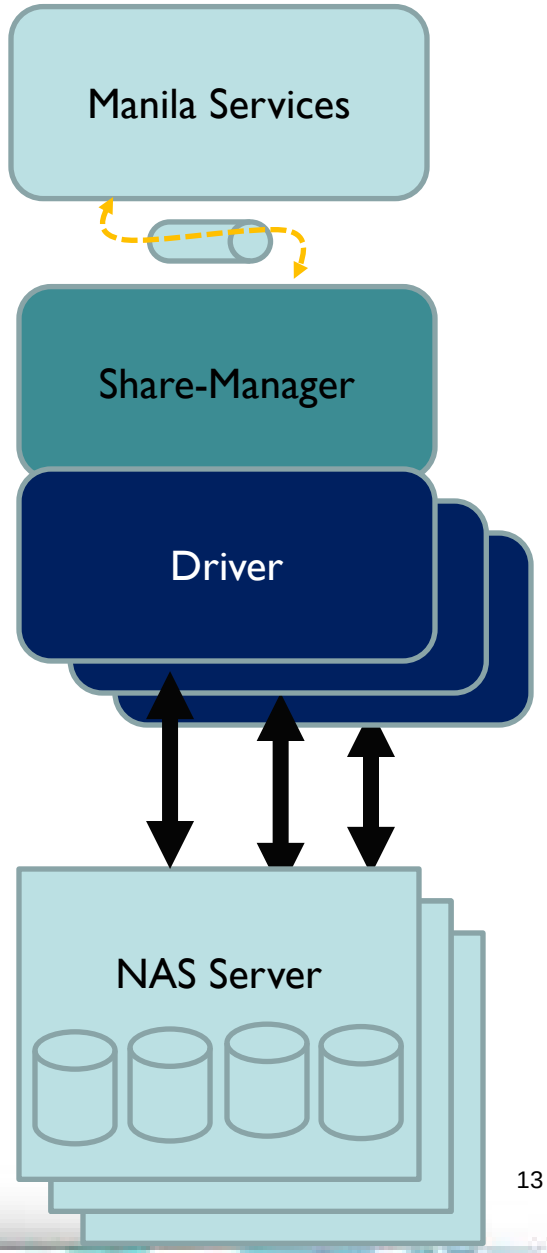
Manila – Architecture Tenets

- Share Network
- Security Services
- Multiple Backend
  - Filter Scheduler
  - manage large-scale shared storage by filtering back-ends
  - Support Pools – capabilities per pool

create, delete, list, get details, snapshot, and modify access for shares

Manila Driver

- Pluggable model
  - Reference Implementation
  - Vendor specific
- Multi-protocol support
- Share networking model
  - Flat, segment - multitenant



ShareDriver - Base Class

- Main APIs to be implemented
  - Create_share
  - delete_share
  - allow_access
  - deny_access
  - create_snapshot
  - create_share_from_snapshot
  - delete_snapshot
  - _update_share_stats
  - ensure_share

ShareDriver - Properties

1. DHSS - driver_handles_share_servers

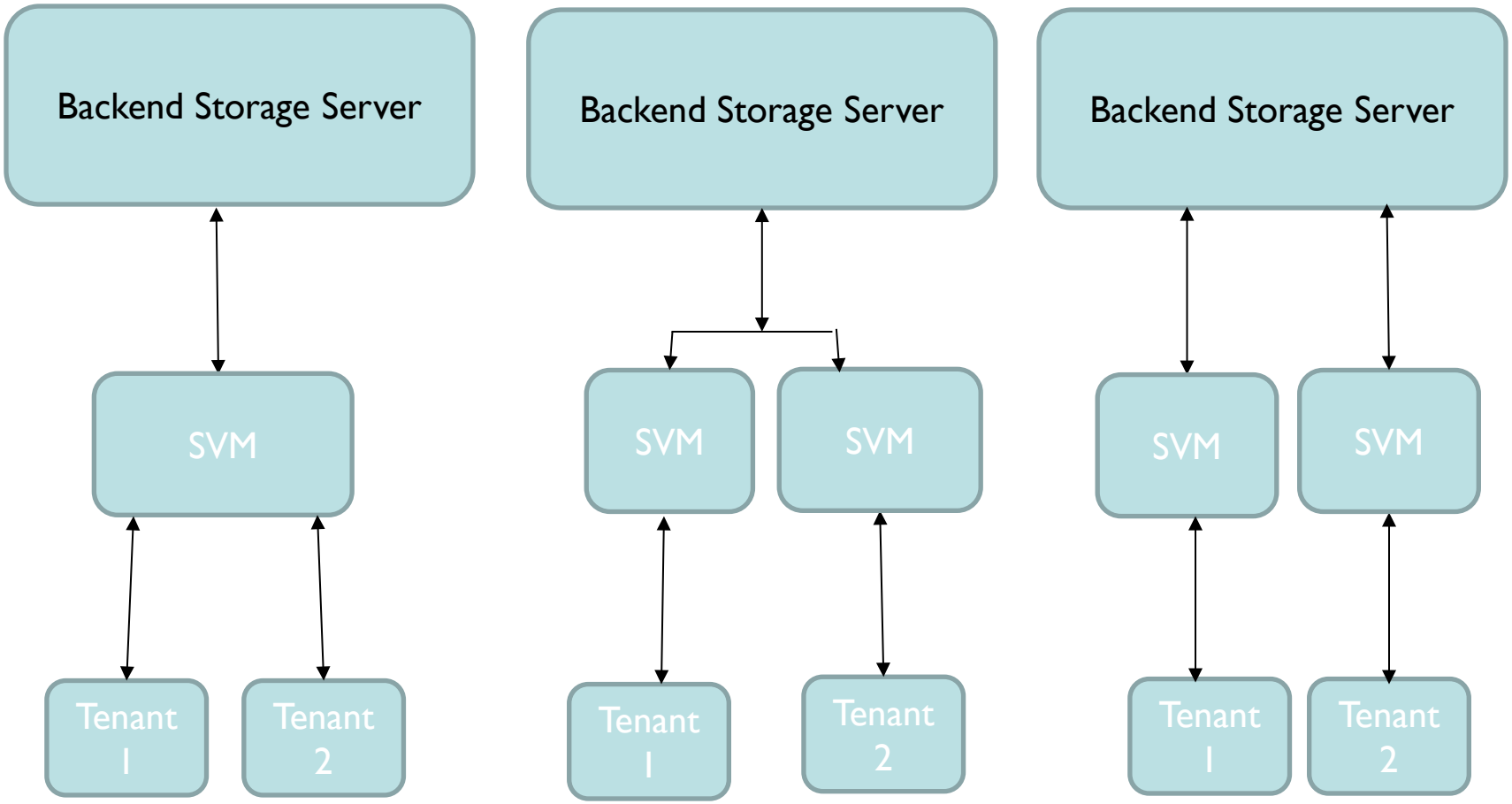
   a. Who creates the share server
   b. How share networking is controlled

2. reserved_share_percentage

Driver Networking Modes

Single SVM	Flat Multiple SVM	Segmented Multiple SVM
▪ All of Network Available ▪ Network opened to all tenets ▪ One single SVM	▪ One SVM Per Tenant ▪ SVMs on a flat network ▪ IP allocation	▪ One SVM Per Tenant ▪ SVMs on different networks ▪ Subnet allocation

Driver Networking Modes



Access Control

- Access rules define which clients can access
  - NFS – IP Address based (CIDR)
  - CIFS – Windows SID
- Support Security Services
  - AD
  - LDAP
  - Kerberos
  - Extensible

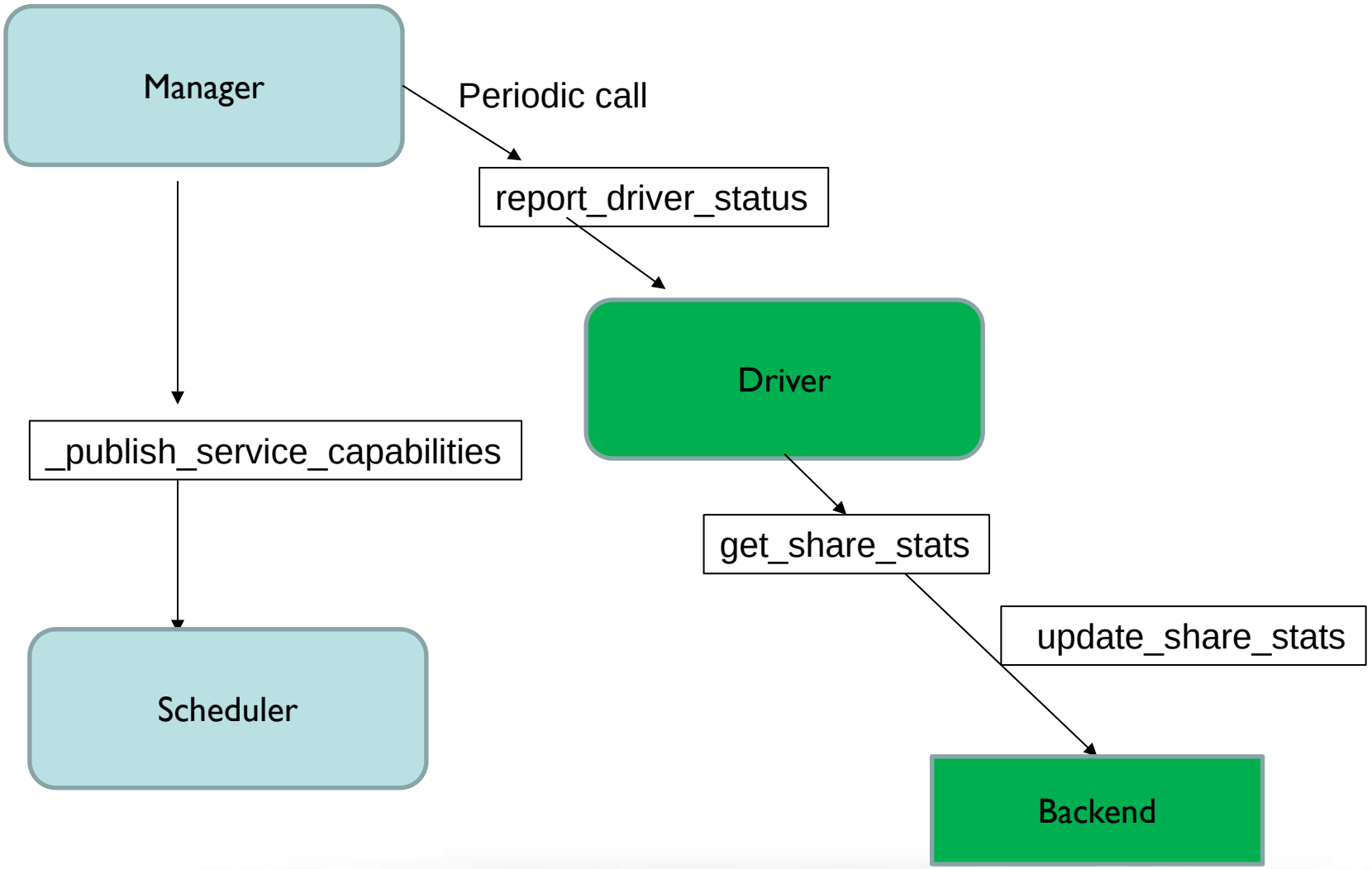
Scheduler – Know your properties

- Selects the backend based on the request
- Filter Scheduler
  - AvailabilityZoneFilter',
  - CapacityFilter',
  - CapabilitiesFilter‘ - Types, Extra-spec
- Pool Aware Support
  - Should come from backend
  - Expose the pools – driver stats

Share - ExtraSpec

- extra_specs to make scheduling decisions
  - Unlike share_types – not visible to clients
  - update_share_stats
- Driver/Backend – publishes the capabilities
  - Key for right scheduling choice
  - Expose backend capability

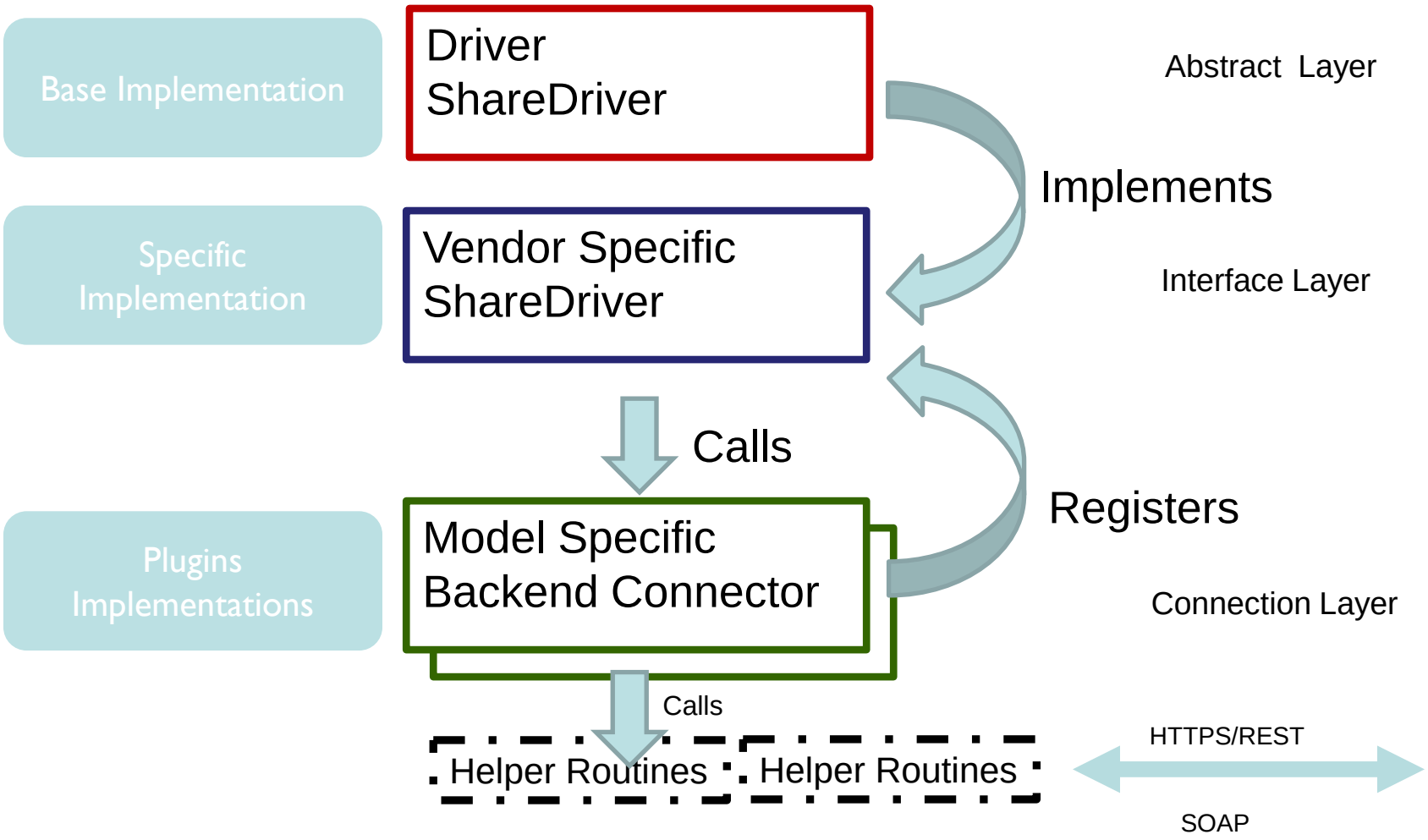
Share Property updates – Get it right !



Driver Templates for Backend

- Interface Layer
  - Common
  - Extensible.
- Connection/Mediator layer
  - Plugin based
  - Backend specific
  - Support multiple versions
- Helper routines
  - Handle Connections
  - REST
  - SOAP

Driver Template



Driver template - Details

- Provide storage device capability in XML form.
  - The interface code is generated
  - Update the extra-spec as required.
- The Command request/response format in XML
  - Connection class is generated.
  - Utility stub generated – needs modification
- Be Thread aware
- No DB access

Future

- Share migration
- Data replication
- QoS - Handling
- Thin provisioning
- And more…

Manila Driver - Recap

- Pluggable module
  - Models, versions
- Publish features
  - Ensure stat/capability update
- Performance
  - Not in data path
  - Check your timeouts
- Template driven

Thank You.