Controller Projects' Modules/Bundles and Interfaces

From Daylight Project
(Redirected from Controller Project's Modules/Bundles and Interfaces)

This is a list of the modules/bundles in each controller, the interfaces they export, the backing implementations of those interfaces, and a brief description of the functionality provided. The three controllers currently sketched out are: the OpenDaylight `controller` (https://git.opendaylight.org/gerrit/#/admin/projects/controller) and the OpenDaylight `net-virt-platform` (https://git.opendaylight.org/gerrit/#/admin/projects/net-virt-platform) . (For posterity's sake, a similar analysis of the Floodlight open source controller can be found at the bottom of the page.)

The current lists are a work in progress and may be either incomplete or out of date. For `net-virt-platform`, the list of modules, interfaces and implementations should be complete as of 4/30/2013. The similar list of package/bundles and interfaces (but not implementations) for `controller` should be complete as of 4/30/2013.

The lists were generated by searching for implementations of org.sdnplatform.core.module.IModule.getModuleServices() in `net-virt-platform` and all calls to org.osgi.framework.ServiceRegistration.Component.setInterface() in `controller`. Note that there are two implementations of the second function: one that takes a single interface and one that takes a list.

The `net-virt-platform` code is presented in a hierarchy of package => module => interface => implementation since that was pretty fairly easy to extract from the code.

The `controller` code is presented in a hierarchy of package/bundle => interface. In practice, there is a 1:1 correspondence between packages and bundles in the code, so that's fine. However, there is no real notion of a "module" which is easy to pull out. The result is that there is only currently per-bundle information.

> In reality, it seems like there's something of a concept of a module in the Daylight code, but it's weakly codified. Basically, in implementations of org.opendaylight.controller.sal.core.ComponentActivatorAbstractBase.configureInstance() and org.opendaylight.controller.sal.core.ComponentActivatorAbstractBase.configureGlobalInstance(), they pass the particular implementation being activated and based on that they change the interfaces. In some cases, the package/bundle only has one such expected implementation and thus they don't ever use it. In other cases it has multiple, which are effectively modules, and it modifies it's behavior depending on which implementation is passed back.

Hopefully, this will provide at least a sound technical backing for discussions going forward.

Daylight

OpenDaylight Controller

Package/OSGi Bundle	Exported Interfaces	Description
org.opendaylight.controller.arphandler.internal	▪ **IHostFinder** ▪ IListenDataPacket	Component responsible for learning about host location. It achieve to goal by looking at the ARP conversation between an host and the controller. This is an application that show case a possible host tracking mechanism, especially useful in the cases like OpenFlow where the controller can see all the packets if instructed to.
org.opendaylight.controller.clustering.services_implementation.internal	▪ **IClusterContainerServices** ▪ **IClusterServices** ▪ **IClusterGlobalServices**	Allow to other components of the controller to allocate/deallocate clustered ConcurrentMap. Also provides a set of utility functions to know about the configuration of the cluster and for interacting with the Java transaction manager
org.opendaylight.controller.clustering.stub.internal	▪ **IClusterContainerServices** ▪ **IClusterGlobalServices**	Stub version of the cluster manager, to be used mostly in integration tests.
org.opendaylight.controller.configuration.internal	▪ **IConfigurationService** ▪ **IConfigurationContainerService** ▪ IConfigurationAware	Provide configuration services to the other bundles. Via this bundle the components can manage their configuration need and just worry about the data to be persisted without worrying about location of where the data will be persisted.
	▪ **IContainerManager**	Provide the implementation of

org.opendaylight.controller.containermanager.internal	▪ **IContainer**	"default" container for the controller.
org.opendaylight.controller.forwarding.staticrouting.internal	▪ **IForwardingStaticRouting** ▪ ICacheUpdateAware ▪ IfNewHostNotify ▪ IConfigurationContainerAware	Provide the necessary hooks to inject in the area controlled by the controller, routes to reach traditional IP networks.
org.opendaylight.controller.forwardingrulesmanager.internal	▪ IContainerListener ▪ ISwitchManagerAware ▪ **IForwardingRulesManager** ▪ IInventoryListener ▪ ICacheUpdateAware ▪ IConfigurationContainerAware ▪ IFlowProgrammerListener	Manager of all the Forwarding Rules, this component take case of forwarding rules and is the one that manage conflicts between them.
org.opendaylight.controller.hosttracker.internal	▪ ISwitchManagerAware ▪ IInventoryListener ▪ **IfIptoHost** ▪ IfHostListener ▪ ITopologyManagerAware	Track the location of the host relatively to the SDN network.
org.opendaylight.controller.protocol_plugin.openflow.internal	▪ IContainerListener ▪ **IController** ▪ IDataPacketListen ▪ **IDataPacketMux** ▪ **IDiscoveryService** ▪ IFlowProgrammerNotifier ▪ IInventoryShimExternalListener ▪ IMessageListener ▪ **IOFStatisticsManager** ▪ **IPluginInDataPacketService** ▪ **IPluginInFlowProgrammerService** ▪ **IPluginInInventoryService**	Protocol plugin for OpenFlow 1.0. Include the openflowJ library as well the necessary glue logic to adapt to SAL layer along with a discovery mechanism for learning the graph of the OpenFlow

	• **IPluginInReadService** • **IPluginInTopologyService** • **IPluginReadServiceFilter** • **IRefreshInternalProvider** • IStatisticsListener • ITopologyServiceShimListener	network portion.
org.opendaylight.controller.protocol_plugins.stub.internal	• **IPluginInReadService**	Sample Protocol plugin
org.opendaylight.controller.routing.dijkstra_implementation.internal	• ITopologyManagerAware • **IRouting**	Implementation of Dijkstra routing algorithm over the network graph as seen by the topology manager. The component keeps a cache of the topology in order to save on calculation time. Make use of the Jung2 third-party library for the calculation.
org.opendaylight.controller.sal.implementation.internal	• **IReadService** • **IPluginOutTopologyService** • **ITopologyService** • **IInventoryService** • **IPluginOutInventoryService** • **IFlowProgrammerService** • **IPluginOutFlowProgrammerService** • **IPluginOutDataPacketService** • **IDataPacketService**	Implements the services that SAL export to the applications using it as well to the protocol plugins. It essentially implements the necessary logic to mux all the notifications from the protocol plugins toward the applications, and demux the requests from applications sitting on SAL toward the correct protocol plugin.
org.opendaylight.controller.samples.loadbalancer.internal	• IListenDataPacket • **IConfigManager**	Implementation of a simple load-balancer. This component wants to show case the usage of the functional modules.
		Sample implementation of an

org.opendaylight.controller.samples.simpleforwarding.internal	▪ IInventoryListener ▪ IfNewHostNotify ▪ IListenRoutingUpdates	application simulating a traditional IP network. This component wants to show case the usage of the functional modules.
org.opendaylight.controller.statisticsmanager.internal	▪ **IStatisticsManager**	Component in charge of using the ReadService from SAL, in order to collect several statistics from the SDN network.
org.opendaylight.controller.switchmanager.internal	▪ IListenInventoryUpdates ▪ **ISwitchManager** ▪ ICacheUpdateAware ▪ IConfigurationContainerAware	Component holding the inventory information for all the known nodes in the controller. All the components that wants to have access to let say a port name or node name or any inventory information, will find them via query to this component.
org.opendaylight.controller.topologymanager.internal	▪ IListenTopoUpdates ▪ **ITopologyManager** ▪ IConfigurationContainerAware	Component holding the whole network graph. Provide notifications on edges updates to who wants to listen about it.
org.opendaylight.controller.usermanager.internal	▪ ICacheUpdateAware ▪ **IUserManager** ▪ IConfigurationAware	Component taking care of user management. Every component in a need of user authentication/authorization and accounting will use the services of this component.
org.opendaylight.controller.security		Fragment of the embedded Tomcat web server, interacting with the usermanager in order to validate the incoming REST/UI calls to be coming from valid users.
		JAXRS implementation of REST API for several modules. There

org.opendaylight.controller.*.northbound		are several component implementing the REST API because each component can be removed or upgraded independent from others.
org.opendaylight.controller.web	▪ **IDaylightWeb**	Host for the root UI as well component in charge of tracking the several pieces of the UI depending on bundles installed on the system.
org.opendaylight.controller.*.web	▪ IDaylightWeb	Bundles implementing the several pieces of the UI.

Net Virt Platform

Net Virt Platform

Package	Module/Interface Class - Implementation Class	Description
org.sdnplatform	BetterDriverManager	
org.sdnplatform.addressspace	AddressSpaceManagerImpl ▪ IAddressSpaceManagerService - AddressSpaceManagerImpl ▪ IEntityClassifierService - AddressSpaceManagerImpl	Manages devices with overlapping identifiers (e.g., allows two hosts with the same MAC to live in different address spaces). For multi-tenancy.
org.sdnplatform.core	ControllerProvider ▪ IControllerService - Controller OFMessageFilterManager ▪ IOFMessageFilterManagerService - OFMessageFilterManager	Core control loops for OpenFlow switches, HA, application registration and message dispatch.

org.sdnplatform.counter	CounterStore • ICounterStoreService - CounterStore NullCounterStore • ICounterStoreService - NullCounterStore	Interface for handling counters and relevant operations on them.
org.sdnplatform.devicemanager.internal	BetterDeviceManagerImpl • ITagManagerService - BetterDeviceManagerImpl • IDeviceService - DeviceManagerImpl DefaultEntityClassifier • IEntityClassifierService - DefaultEntityClassifier DeviceManagerImpl • IDeviceService - DeviceManagerImpl	Maintains a unique ID for each device and any associated state, e.g., it's attachment point in the network, discovered IP addresses, vlans, and MAC addresses.
org.sdnplatform.flowcache	BetterFlowCache • IFlowCacheService - BetterFlowCache FlowCache • IFlowCacheService - FlowCache FlowReconcileManager	

	▪ IFlowReconcileService - FlowReconcileManager PortDownReconciliation	
org.sdnplatform.forwarding	Forwarding ▪ IForwardingService - Forwarding RewriteServiceImpl ▪ IRewriteService - RewriteServiceImpl	A packet forwarding module that determines the shortest path between two hosts (Dijkstra) and lays down a path with flows accordingly.
org.sdnplatform.hub	Hub	A packet forwarding module implemented to function like a hub
org.sdnplatform.jython	JythonDebugInterface	A debug module that launches a Jython debug server.
org.sdnplatform.learningswitch	LearningSwitch ▪ ILearningSwitchService - LearningSwitch	
org.sdnplatform.linkdiscovery	BetterLinkDiscoveryManager	
org.sdnplatform.linkdiscovery.internal	LinkDiscoveryManager ▪ ILinkDiscoveryService - LinkDiscoveryManager	
org.sdnplatform.loadbalancer	LoadBalancer ▪ ILoadBalancerService - LoadBalancer	A simple load balancer module for ping, tcp, and udp flows. This module is exposed via a REST API and is defined closely to the OpenStack Quantum LBaaS (Load-balancer-as-a-Service) v1.0 API proposal. See http://wiki.openstack.org/Quantum/LBaaS.
	NetVirtManagerImpl	

org.sdnplatform.netvirt.manager.internal	▪ INetVirtManagerService - NetVirtManagerImpl	
org.sdnplatform.netvirt.virtualrouting.internal	VirtualRouting ▪ IVirtualRoutingService - VirtualRouting ▪ IVirtualMacService - VirtualRouting	
org.sdnplatform.ovsdb.internal	OVSDBManagerImpl ▪ IOVSDBManagerService - OVSDBManagerImpl	
org.sdnplatform.perfmon	NullPktInProcessingTime ▪ IPktInProcessingTimeService - NullPktInProcessingTime PktInProcessingTime ▪ IPktInProcessingTimeService - PktInProcessingTime	
org.sdnplatform.restserver	RestApiServer ▪ IRestApiService - RestApiServer	Defines a REST API server implementation and provides an interface for modules that wish to expose REST APIs.
org.sdnplatform.staticflowentry	StaticFlowEntryPusher ▪ IStaticFlowEntryPusherService - StaticFlowEntryPusher	A service that is exposed internally as well as externally via a REST API. It allows for other modules or users to easily insert flows into an OpenFlow network.
	AbstractStorageSource	

org.sdnplatform.storage	▪ IStorageSourceService - AbstractStorageSource	Abstract platform storage service.
org.sdnplatform.storage.cassandra	CassandraStorageSource ▪ IStorageSourceService - CassandraStorageSource	Concrete Cassandra-based NoSql storage service (extends NoSqlStorageSource).
org.sdnplatform.storage.nosql	NoSqlStorageSource ▪ IStorageSourceService - NoSqlStorageSource	Abstract NoSql storage service (extends AbstractStorageSource).
org.sdnplatform.storage.memory	MemoryStorageSource ▪ IStorageSourceService - MemoryStorageSource	Concrete, memory-based NoSql storage service (extends NoSqlStorageSource).
org.sdnplatform.threadpool	ThreadPool ▪ IThreadPoolService - ThreadPool	Implements platform's thread pooling service. By default it creates an ScheduledExecutorService with 15 threads. The thread pool can be used to schedule commands to run after the caller-specified delay or to be executed periodically.
org.sdnplatform.topology	BetterTopologyManager ▪ ITopologyService - BetterTopologyManager ▪ IBetterTopologyService - BetterTopologyManager ▪ IRoutingService - BetterTopologyManager TopologyManager ▪ ITopologyService - TopologyManager ▪ IRoutingService - TopologyManager	

org.sdnplatform.tunnelmanager	TunnelManager - ITunnelManagerService - TunnelManager	

Floodlight

packages => modules => exported interfaces

- net.floodlightcontroller.core
  - OFMessageFilterManager
    - IOFMessageFilterManagerService
  - FloodlightProvider/Controller
    - IFloodlightProviderService
- net.floodlightcontroller.counter
  - CounterStore
    - ICounterStoreService
  - NullCounterStore
    - ICounterStoreService
- net.floodlightcontroller.devicemanager.internal
  - DefaultEntityClassifier
    - IEntityClassifierService
  - DeviceManagerImpl
    - IDeviceService
- net.floodlightcontroller.firewall
  - Firewall
    - IFirewallService
- net.floodlightcontroller.flowcache
  - FlowCache
    - IFlowCacheService
  - FlowReconcileManager
    - IFlowReconcileService
  - PortDownReconciliation
    - (none)

- net.floodlightcontroller.forwarding
  - Forwarding
    - (none)
- net.floodlightcontroller.hub
  - Hub
    - (none)
- net.floodlightcontroller.jython
  - JythonDebugInterface
    - (none)
- net.floodlightcontroller.learningswitch
  - LearningSwitch
    - ILearningSwitchService
- net.floodlightcontroller.linkdiscovery.internal
  - LinkDiscoveryManager
    - ILinkDiscoveryService
- net.floodlightcontroller.loadbalancer
  - LoadBalancer
    - ILoadBalancerService
- net.floodlightcontroller.perfmon
  - NullPktInProcessingTime
    - IPktInProcessingTimeService
  - PktInProcessingTime
    - IPktInProcessingTimeService
- net.floodlightcontroller.restserver
  - RestApiServer
    - IRestApiService
- net.floodlightcontroller.staticflowentry
  - StaticFlowEntryPusher
    - IStaticFlowEntryPusherService
- net.floodlightcontroller.storage
  - AbstractStorageSource
    - IStorageSourceService
  - NoSqlStorageSource
    - IStorageSourceService
- net.floodlightcontroller.threadpool
  - ThreadPool
    - IThreadPoolService
- net.floodlightcontroller.topology

        - TopologyManager
            - ITopologyService
            - IRoutingService
- net.floodlightcontroller.ui.web
    - StaticWebRoutable
        - (none)
- net.floodlightcontroller.virtualnetwork
    - VirtualNetworkFilter
        - IVirtualNetworkService
