# Open Control Architecture 1.2

**OCA ALLIANCE**
**OPEN CONTROL ARCHITECTURE**

Control Classes Overview

## 1. Introduction

This document gives a brief overview of the OCA Control Class hierarchy. This hierarchy is commonly called **OCC**. OCC is the set of classes that are instantiated to define the control objects that constitute the OCA device model.

By **device model**, we mean the abstraction that OCA uses to describe the remote control functions of a device.

We use the terms **control class** and **control object** to emphasize that OCC defines control and monitoring operations only. OCC classes are not programming classes and do not define the structures of OCA controller or device implementations.

This document is a companion document to the full OCC specification documents, which are:

| | |
|---|---|
| **[occ1]** | The full UML description of OCC, as an Enterprise Architect *.EAP file. |
| **[occ2]** | The full set of class descriptions for OCC, in PDF form. Generated from [occ1]. |
| **[occ3]** | The UML class diagrams that go with [occ2]. |
| **[ocf1]** | The OCA Framework document (commonly called **OCF**) that describes the structures and mechanisms of OCA. |

These documents are available from the OCA Alliance website, http://OCA-Alliance.org.

## 2. Kinds of Classes

In OCC, there are three main kinds of control classes:

- **Workers**
- **Managers**
- **Agents**

These are the classes that define OCA device model.

There is also two sets of supporting definitions:

- **Control Datatypes**
- **Control Class Construction Parameters**

These elements and their contents will be summarized below. The entire hierarchy is shown in Figure 1.

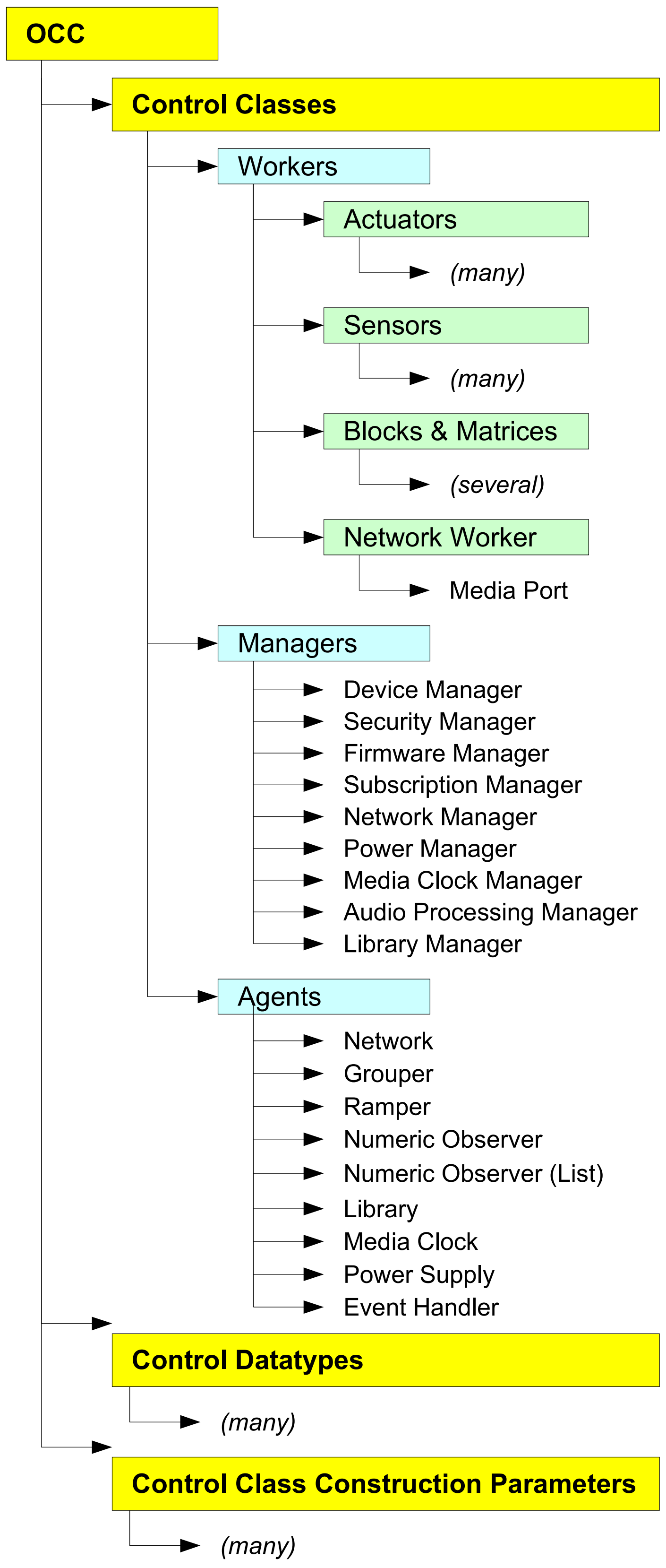
OCC
Control Classes
Workers
Actuators
(many)
Sensors
(many)
Blocks & Matrices
(several)
Network Worker
Media Port
Managers
Device Manager
Security Manager
Firmware Manager
Subscription Manager
Network Manager
Power Manager
Media Clock Manager
Audio Processing Manager
Library Manager
Agents
Network
Grouper
Ramper
Numeric Observer
Numeric Observer (List)
Library
Media Clock
Power Supply
Event Handler
Control Datatypes
(many)
Control Class Construction Parameters
(many)

**Figure 1. OCC**

## 2.1. Worker Classes

These are the sets of classes that describe device application functions. There are four kinds of workers:

| | |
|---|---|
| • **Actuators** | Signal processing and routing functions. The current version of OCC defines 23 actuator classes. |
| • **Sensors** | Detectors and monitors of various types, e.g. signal level, gain reduction, temperature. The current version of OCC defines 7 sensor classes. |
| • **Blocks and Matrices** | Special classes that aggregate objects into structured collections. Used for modeling and managing structures of complex devices. The current version of OCC defines 3 block and matrix classes. |
| • **Network Worker** | Media port class that allows connections to external incoming and outgoing media streams. |

For any given device, a worker class may be instantiated as many times as necessary to model the device's functions.

Where necessary, a worker class may be refined and extended by a manufacturer-specific child class. This is explained further in [ocf1].

## 2.2. Manager Classes

These are the classes that describe a small set of basic objects which model basic overhead functions of the device. For any given device, each manager class will be instantiated once at most.

Some manager classes are required for OCA compliance and must be instantiated. Others are optional. As well, not all interface elements of all required classes must be implemented. Minimum device requirements for OCA compliance will be described in another document from the Alliance; it is currently under revision.

The manager classes are:

| | |
|---|---|
| • **Device Manager** | Contains manufacturer and model information and controls overall device state. |
| • **Security Manager** | Controls security features, or reports that there are none.. |
| • **Firmware Manager** | Manages device firmware updating, or reports that it is not implemented. |
| • **Subscription Manager** | Manages the reporting of device data back to controllers. |
| • **Network Manager** | Anchors the device's network interface(s), as defined by the network objects mentioned in section 2.1. |
| • **Power Manager** | Allows control and monitoring of device's power supply or supplies. |
| • **Media Clock Manager** | Gives access to device's media clocking features. |
| • **Audio Processing Manager** | Gives access to global parameters controlling audio processing. |
| • **Library Manager** | Controls creation, management, and use of stored parameter sets (aka "presets", "patches", etc.) |

## 2.3. Agent Classes

These are the classes that provide notable control features that are not directly related to signal processing. The agent classes are:

| | |
|---|---|
| • **Network** | Supports connections to control and media networks. |
| • **Grouper** | Supports control aggregation, allowing a single parameter change to affect many objects. Similar in effect to a VCA master on a mixing console. |
| • **Ramper** | Provides incremental parameter changes -- timed fades, for example. Also provides queuing of parameter changes to occur at specified times in the future. |
| • **NumericObserver** | A "watcher" that observes a particular parameter, and alerts controllers when it reaches a particular value. Also supports periodic reporting of parameter values (e.g. level meter readings) to controllers. |
| • **Library** | Supports a range of functions for pre-storing sets of parameters in the device and applying them when desired. |
| • **MediaClock** | Describes an internal or external media clock that the device uses. Can be multiply instantiated for devices which support more than one media clock. |
| • **EventHandler** | Describes the controller interface that handles incoming notifications from controlled devices. |

## 2.4. Control Datatypes

OCC defines a range of control datatypes. These datatypes are used in the definitions of the classes listed above. For details, please see the full report.

## 2.5. Control Class Construction Parameters

Some DSP-based products allow controllers to define their processing topologies. OCA refers to these as **configurable devices**. In some configurable devices, controllers can cause new processing objects to be created and deleted. OCA refers to these as **fully configurable devices.**

When a controller creates an object in a fully configurable device, certain parameters may be required. For example, if a controller creates a multiposition switch, it needs to specify how many positions the switch has and, optionally, a text label for each position. Such parameters are called **construction parameters**.

The number and kind of construction parameters varies from class to class. OCC maintains a separate subtree that specifies what construction parameters are required for each class. This subtree has an entry corresponding to every Worker class and every Agent class. OCA does not expect fully configurable devices to allow runtime creation of Manager objects, so these are not included in the construction parameter set.

# 3. Control Class and Element Identification

Control classes and their elements are identified by codes that reflect the class hierarchy level at which they are defined. These codes are used for efficient handling of properties, methods, and events in OCA network protocols.

The scheme is as follows:

## 3.1. Control Class Level Numbers

Each control classes is given a two-digit tree level number. Thus:

- The root class level number is **01.**
- **OcaRoot**'s children have level number **02**.
- **OcaRoot**'s grandchildren have level number **03**.
- ... and so on.

## 3.2. Property, Method, and Event IDs

Within each class, properties, methods, and events are identified by their **IDs**, which are strings of the form

**LL T NN**

where **LL** is the tree level number of the class, as described above;
**T** is **"p"** for property, **"m"** for method, or **"e"** for event;
**NN** is a sequence number that starts at 01 for each class.

for example

**03p02** is the second property of a class defined at tree level 03.
**04m05** is the fifth method of a class defined at tree level 04.
**02e03** is the third event of a class defined at tree level 02.

Control class and element name IDs are **distinct from OCA class IDs.**

## 3.3. Class IDs

A class ID is a vector of numbers, one for each tree level in the class's ancestry. In OCA documentation, this vector is written as a sequence of integers separated by dots, e.g.

| | |
|---|---|
| **1.3.2** | The second child of the third child of the root class. |
| **1** | The root class. |
| **1.1** | The first child of the root class. |
| **2** | Invalid -- there are never two root classes. |

In protocol implementations, class IDs will usually be serialized as variable-length arrays of 16-bit unsigned integers.

As the class tree grows in the future, class IDs will be assigned by the standards authority that will eventually be responsible for maintaining OCA.

Note that the number of elements in a class ID is equal to the class's level number. For example, a class whose level number is **03** will have an ID of the form **i.j.k**, e.g. **3.4.1**; a class whose level number is **05** will have an ID of the form **i.j.k.m.n**, e.g. **2.6.1.2.4**.