```
//  (c) 2010 Fermi Research Alliance
//  $Id: DAQData.proto,v 1.1 2011/07/08 18:41:31 kingc Exp $

//
// DPM load balancing messages
//

request LoadMessage {
}

reply LoadMessage {
}

//
// Common DAQ data types....
//

request ReadingRequest {
    string dataRequest[];
}

reply ReadingRequestStatus {
    int16 status[];
}

request SettingRequest {
    string dataRequest[];
}

reply BinarySample {
    binary value;
    int64 time;
    optional string unit;
    optional int16 format_hint;
    optional int16 ref_id;
}

reply BooleanSample {
    bool value;
    int64 time;
    optional string unit;
    optional int16 format_hint;
    optional int16 ref_id;
}

reply BooleanArraySample {
    bool value[];
    int64 time;
    optional string unit;
    optional int16 format_hint;
    optional int16 ref_id;
}

reply ShortSample {
    int16 value;
    int64 time;
    optional string unit;
    optional int16 format_hint;
    optional int16 ref_id;
}

reply ShortArraySample {
    int16 value[];
    int64 time;
```

```
        optional string unit;
        optional int16 format_hint;
        optional int16 ref_id;
    }

    reply IntegerSample {
        int32 value;
        int64 time;
        optional string unit;
        optional int16 format_hint;
        optional int16 ref_id;
    }

    reply IntegerArraySample {
        int32 value[];
        int64 time;
        optional string unit;
        optional int16 format_hint;
        optional int16 ref_id;
    }

    reply LongSample {
        int64 value;
        int64 time;
        optional string unit;
        optional int16 format_hint;
        optional int16 ref_id;
    }

    reply LongArraySample {
        int64 value[];
        int64 time;
        optional string unit;
        optional int16 format_hint;
        optional int16 ref_id;
    }

    reply DoubleSample {
        double value;
        int64 time;
        optional string unit;
        optional int16 format_hint;
        optional int16 ref_id;
    }

    reply DoubleArraySample {
        double value[];
        int64 time;
        optional string unit;
        optional int16 format_hint;
        optional int16 ref_id;
    }

    reply StringSample {
        string value;
        int64 time;
        optional int16 ref_id;
    }

    struct BasicStatus {
        bool on;
        bool positive;
        bool ready;
        bool remote;
        bool ramp;
        string text;
        string colors;
    }

    reply BasicStatusSample {
        BasicStatus value;
        int64 time;
        optional int16 ref_id;
    }

    struct AnalogAlarm {
        double minimum;
        double maximum;
        bool alarm_enable;
        bool alarm_status;
        bool abort;
        bool abort_inhibit;
        int32 tries_needed;
        int32 tries_now;
    }

    reply AnalogAlarmSample {
        AnalogAlarm value;
        int64 time;
        optional int16 ref_id;
    }

    struct DigitalAlarm {
        int32 nominal;
        int32 mask;
        bool alarm_enable;
        bool alarm_status;
        bool abort;
        bool abort_inhibit;
        int32 tries_needed;
        int32 tries_now;
    }

    reply DigitalAlarmSample {
        DigitalAlarm value;
        int64 time;
        optional int16 ref_id;
    }

    reply ControlCommand {
        string command;
        int64 time;
        optional int16 ref_id;
    }

    reply StatusSample {
        int32 facilityCode;
        int32 errorNumber;
        int64 time;
        optional string message;
        optional int16 ref_id;
    }
```