

An Object Oriented Framework of EPICS for MicroTCA Based Control System

Z. Geng, SLAC, Menlo Park, California, U.S.A.

Introduction

EPICS has introduced object oriented (C++) interfaces to most of the core services. But the major part of EPICS, the run-time database, only provides C interfaces, which is hard to involve the EPICS record concerned data and routines in the object oriented architecture of the software. This paper presents an object oriented framework which contains some abstract classes to encapsulate the EPICS record concerned data and routines in C++ classes so that full OOA (Object Oriented Analysis) and OOD (Object Oriented Design) methodologies can be used for EPICS IOC design. We also present a dynamic device management scheme for the hot swap capability of the MicroTCA based control system.

OOEPICS Framework

The Object Oriented framework for EPICS (OOEPICS) provides several base classes to encapsulate the details of the record processing. The EPICS records and the devices are designed as objects which enable the direct translation of the UML model into EPICS applications. The framework hides most of the details of the EPICS which enable the user to develop his EPICS device driver without knowing much of EPICS.

Base Classes

EPICS Data (epicsData)

EPICS records were defined based on object oriented concept.

Table 1: Comparisons of EPICS Records and C++ Classes

EPICS Records	C++ Classes
Fields	Attributes
Record Support	Methods (platform independent)
Device Support	Methods (platform dependent)

The class of epicsData is defined for the EPICS record. They are linked together by the record name.

EPICS Device (epicsDevice)

The epicsDevice class is an empty class where the user codes need to be implemented for the control of specific devices.

The epicsDevice class may contain:

- epicsData objects for uplink control and monitoring.
- Special logic to perform the device control.
- Algorithms to do signal processing, calibration and optimization.
- Common logical interfaces to interact with physical device.

Device Management (deviceManager and deviceConfig)

The deviceManager and deviceConfig provide a common way to create, associate, initialize and set up a device control module in user IOC. They also provides the EPICS API that can be called in the IOC shell. In principle, the user does not need to create any IOC shell commands.

Codes Generation

The tool of "epics_driver_template" is designed to generate the source code template from the EPICS database file.

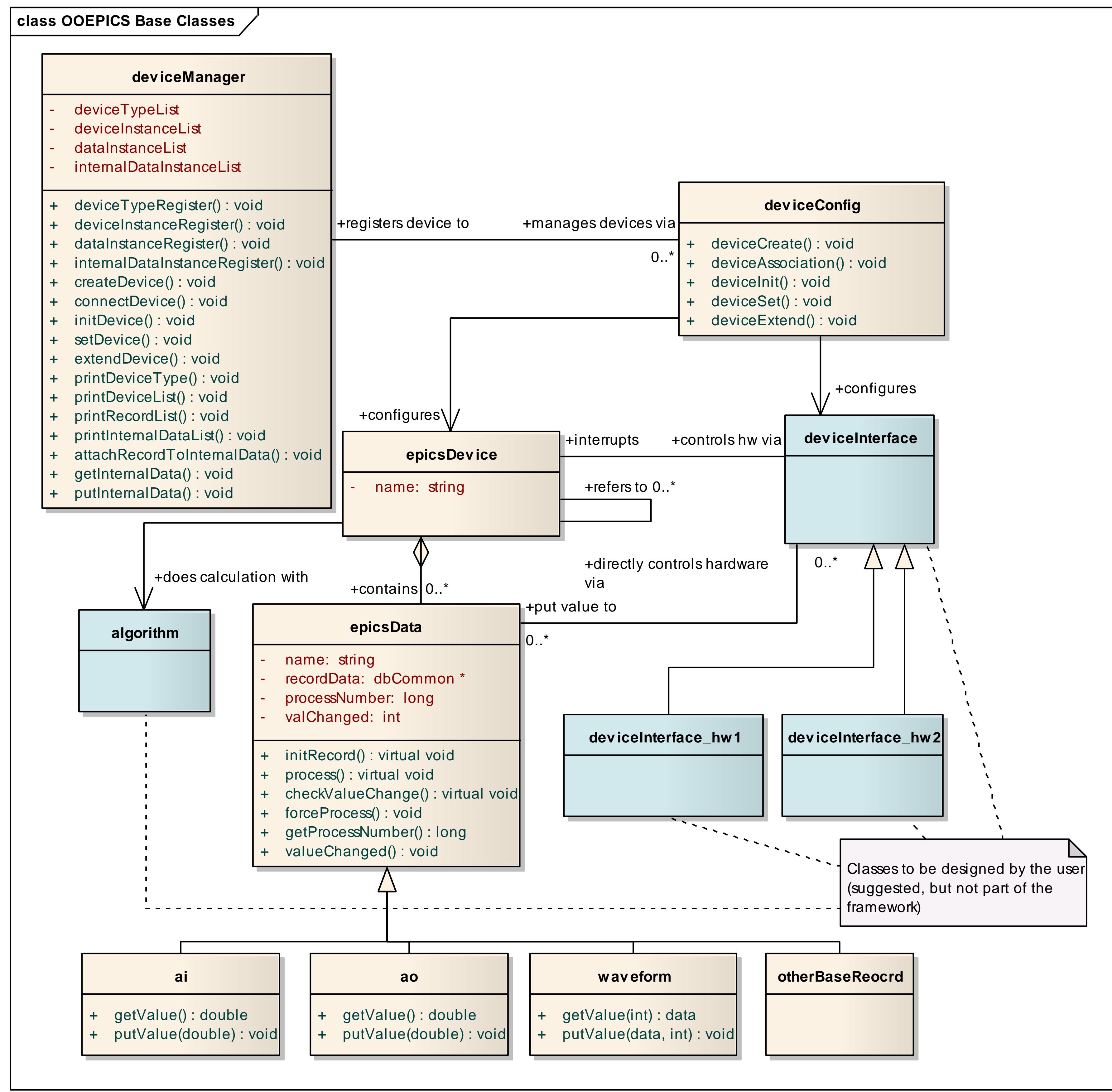
The tool will compile the database file into C strings and then use them to initialize the epicsData objects. A derived device class will be generated with the epicsData objects as attributes. A derived deviceConfig class will be generated which allows to define the procedure to create, initialize and setup the device objects.

Dynamic Records Loading

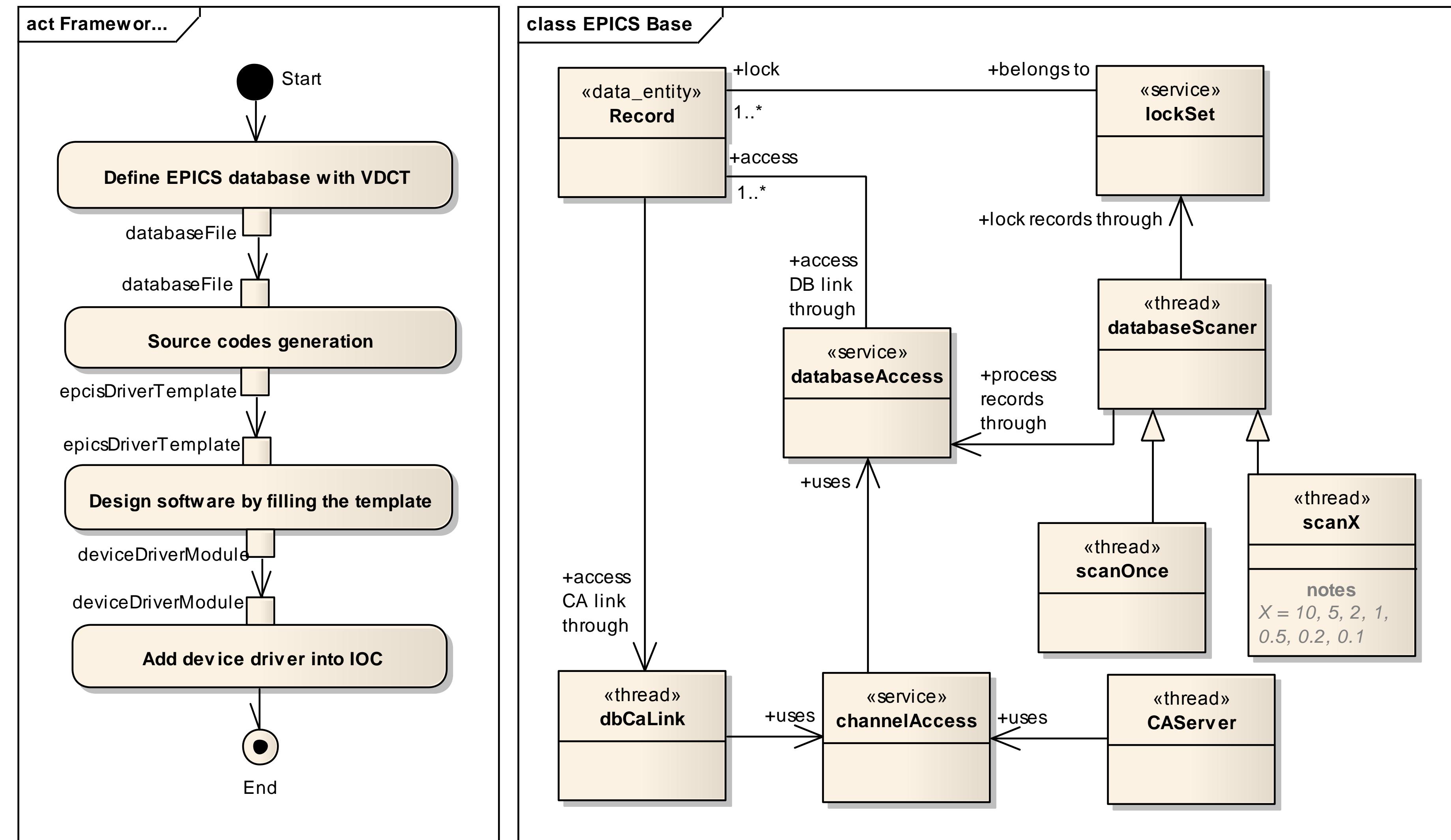
The EPICS IOC needs to support hot swap feature of MicroTCA systems, which means, loading the device drivers for the new boards without rebooting the IOC software. So loading EPICS records after iocInit() should be supported. Several challenges need to be resolved to load a record during run time:

- Setup the locksets for the newly loaded records.
- Convert PV links to DB link or CA link during record initialization.
- Modify the links of existing records.

For solving the problems listed above, a new source file (dbRecordDynamic.c) is added to the EPICS base. (<https://blueprints.launchpad.net/epics-base/+spec/dynamic-record-loading>)



Base classes of OOEPICS framework



Design procedure with OOEPICS

EPICS base modules acting on records

Conclusion

The OOEPICS framework provides a way to fully access the EPICS records from the user code. The object oriented technology can directly map to EPICS design. It also provides a common way to create, initialize and setup the device driver from the IOC shell. It also enables the user to do EPICS development without knowing much of EPICS, avoiding the long learning curve of EPICS.

The dynamic records loading development provides a good support for hot-swappable system control which enables a complete EPICS based solution for MicroTCA system in the following projects at SLAC.