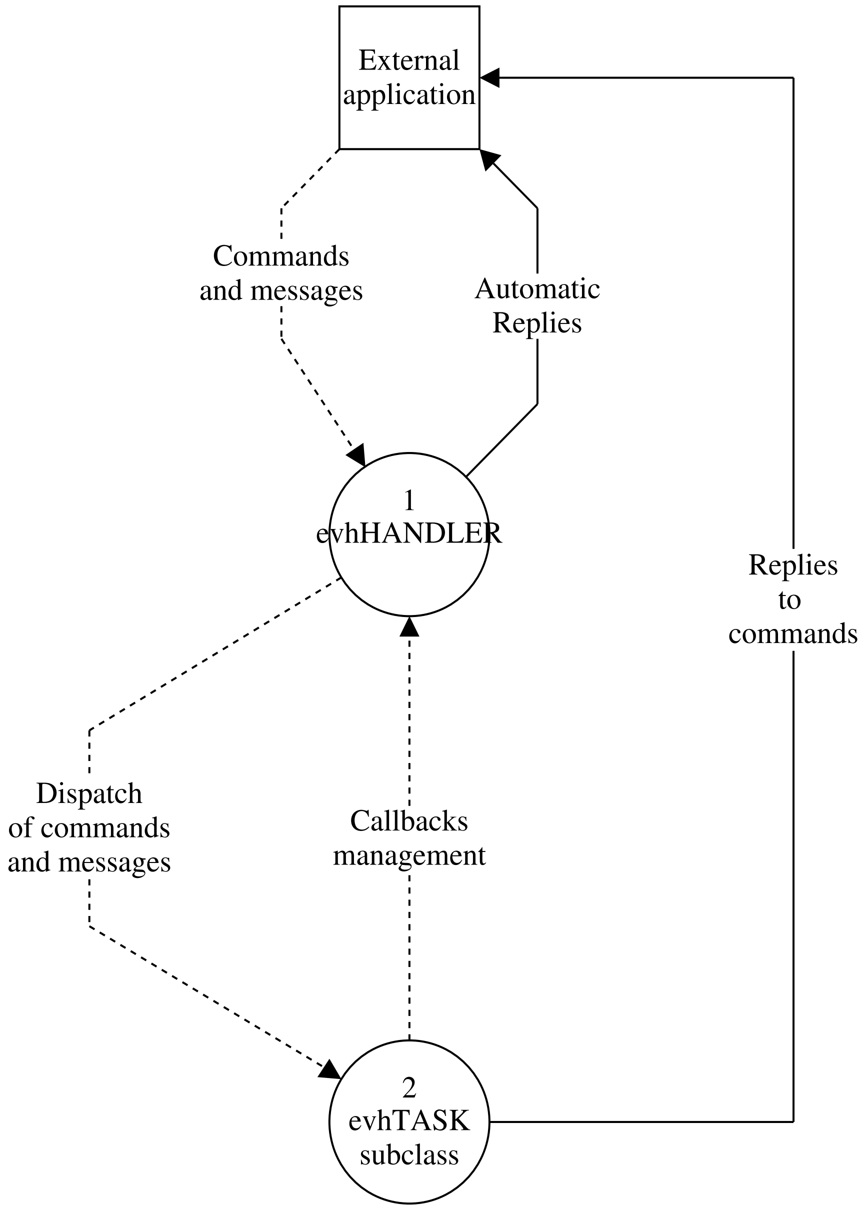
Event Handling Tool-hit Tutorial

G.Chiozzi

03/04/95

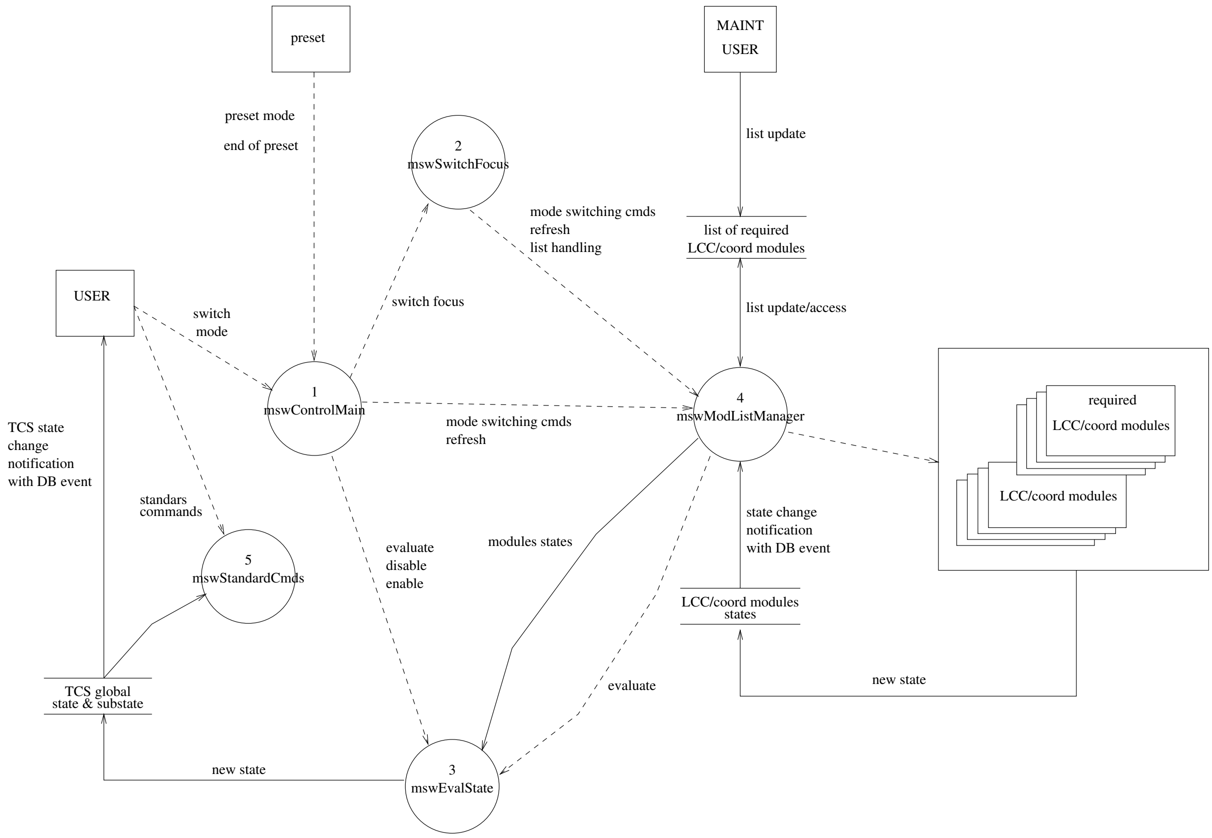
EVH Application Structure



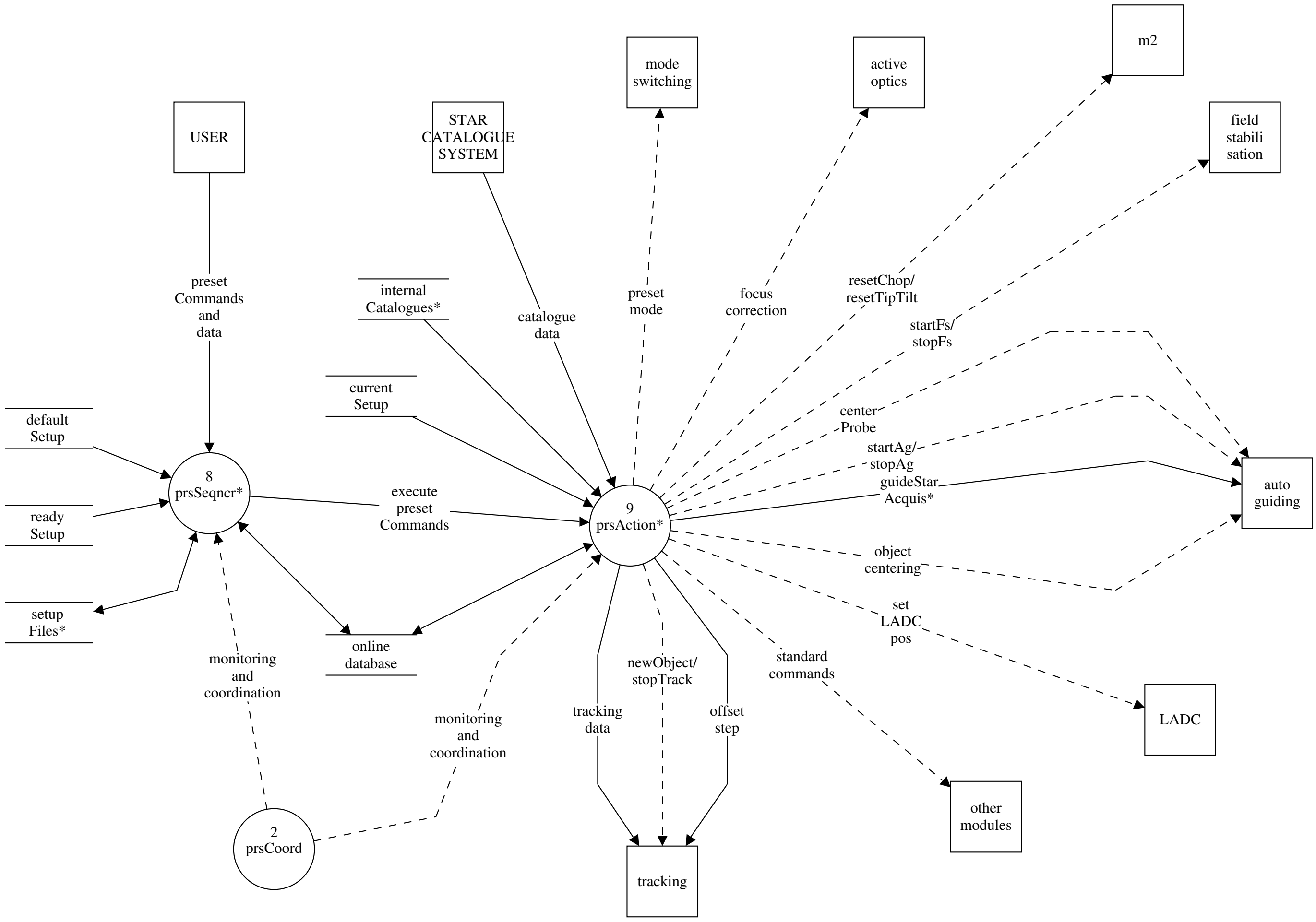
EVH Classes

- **Fundamental building blocks**
  – evhHANDLER
  – evhSIMPLE_TASK, evhTASK, *evhDB_TASK*
- **Basic classes**
  – evhCOMMAND
  – evhSYNC
  – evhTIMEOUT
  – evhTIMER
- **Special classes**
  – evhFILEIO
  – evhSIGNAL
- **Support classes**
  – evhCALLBACK, evhOBJ_CALLBACK
  – evhCONDITION
  – evhMSG_TYPE_KEY
  – evhSELECT

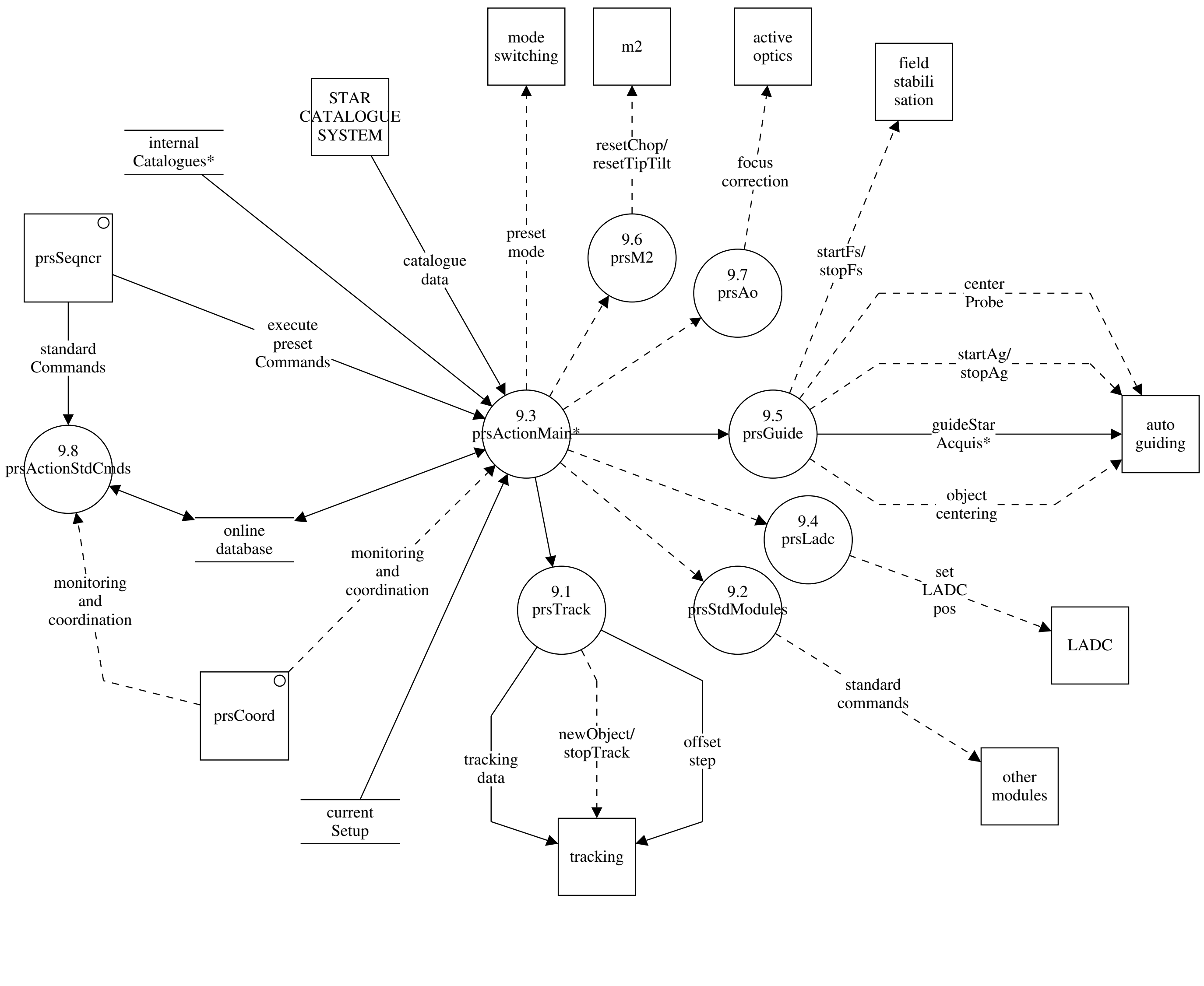
DFD for TCS Mode Switching



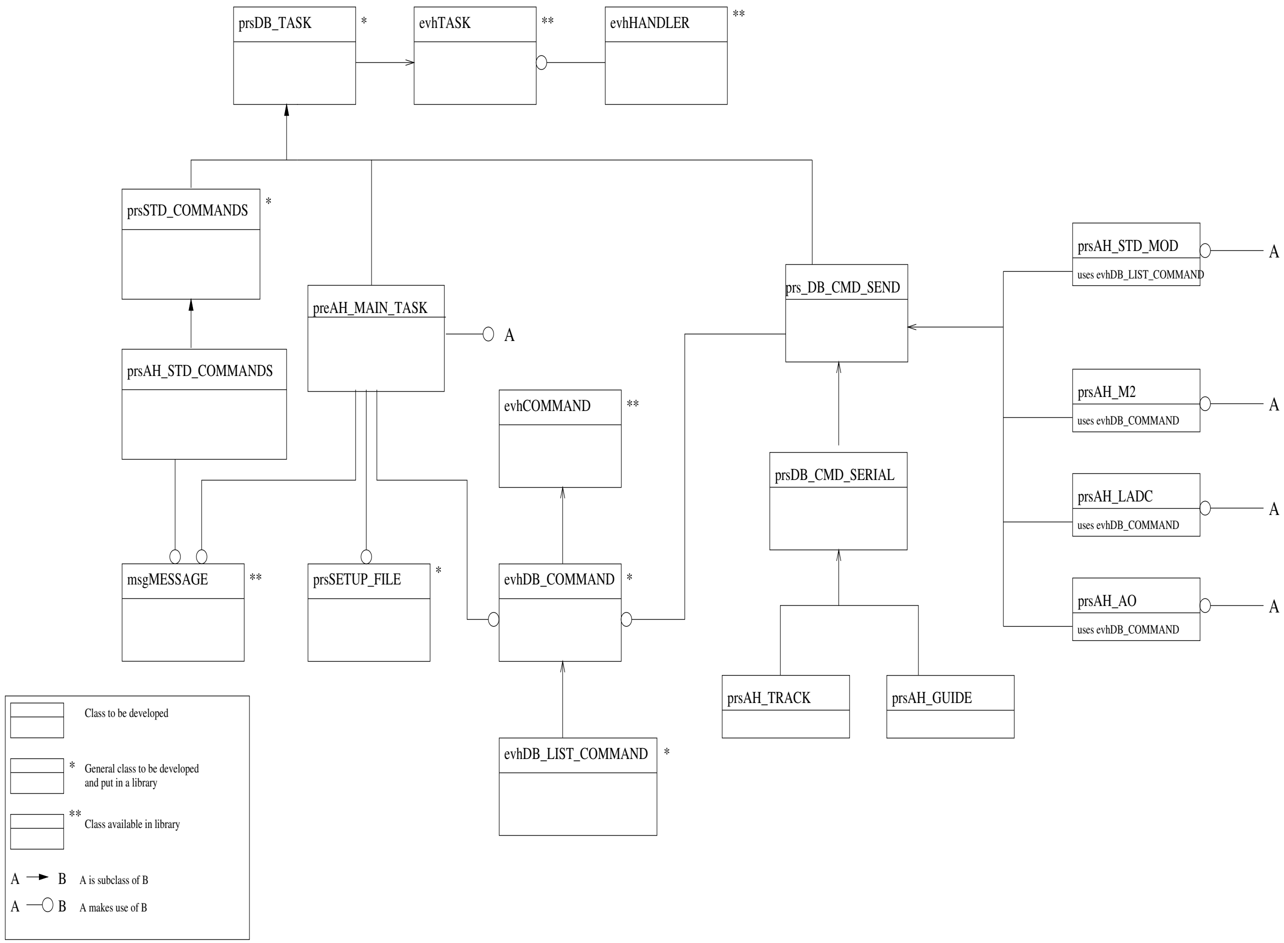
DFD for TCS Preset (0 lev)



DFD TCS Preset Action Main



CLD TCS Preset Action Main



Sample code

```
/************************************************************************
*   NAME
*     evhSample_1
*
*   SYNOPSIS
*
*     evhSample_1
*
*   DESCRIPTION
*
*   This program is a small example of usage of the Evh Toolkit.
*   It can receive the commands implemented in the base class evhTASK
*   plus the new command "SAMPLE".
*   The callback for the "SAMPLE" command just send back a reply to the
*   caller.
*
*   The program demonstrates the following aspects:
*
*    - Main program with TASK object instance and entering the main loop
*    - Inheritance of a new TASK class
*    - Addition of a new command and simple usage of evhMSG_TYPE_KEY
*      and evhOBJ_CALLBACK classes
*    - Usage of EasyCCS class msgMESSAGE to receive messages and send replies
*
*-----------------------------------------------------------------------
*/

#define _POSIX_SOURCE 1
#include "vltPort.h"

#include "ECCS.h"
#include "evhHANDLER.h"
#include "evhTASK.h"
```

```
/*****************************************/
/* Definition for the APPLICATION class   */
/*****************************************/
class APPLICATION: public evhTASK
{
  public:
    APPLICATION();

    // Definition for the callback to be used receiving the
    // "SAMPLE" command
    evhCB_COMPL_STAT  SampleCB(msgMESSAGE &msg, void *udata);

}; /* end class APPLICATION */


/* Body of the constructor method.              */
/* It is called whenever an instance is created */
APPLICATION::APPLICATION()
{
    /* This variable is used to define the event:    */
    /* a message of type msgTYPE_COMMAND, with       */
    /* Command="SAMPLE", no regard for the CommandId */
    evhMSG_TYPE_KEY key(msgTYPE_COMMAND,"SAMPLE");

    /* This variable is used to define the callback:     */
    /* It is a method of this object, no user parameters */
    evhOBJ_CALLBACK cb(this,(evhCB_METHOD)SampleCB);

    /* Then the callback is attached to the event handler */
    evhHandler->AddCallback(key,cb);
}

/* Body for the callback */
evhCB_COMPL_STAT  APPLICATION::SampleCB(msgMESSAGE &msg, void *udata)
{
    /* Just print the message buffer on stdout */
    printf("Received a SAMPLE command: %s\n", msg.Buffer());
```

```
    /* Send back the reply to the command */
    msg.LastReply(ccsTRUE);
    msg.Buffer("Command received!");
    msg.SendReply();

    /* The callback must remain installed for another command */
    return evhCB_NO_DELETE;
}



/***********************************/
/* main procedure                  */
/***********************************/

int main(int argc, char *argv[])
{
    ccsInit(argv[0]);

    /* Create the instance of APPLICATION */
    /* The call to the macro USE() prevents the GCC compiler     */
    /* from issuing a warning: it assumes that app is not used ! */
    APPLICATION app; USE(app);

    printf("Entering the main loop and waiting for commands!\n");

    /* enter the main loop */
    evhHandler->MainLoop();

    ccsExit();
}
```