Hall-D Farm Manager

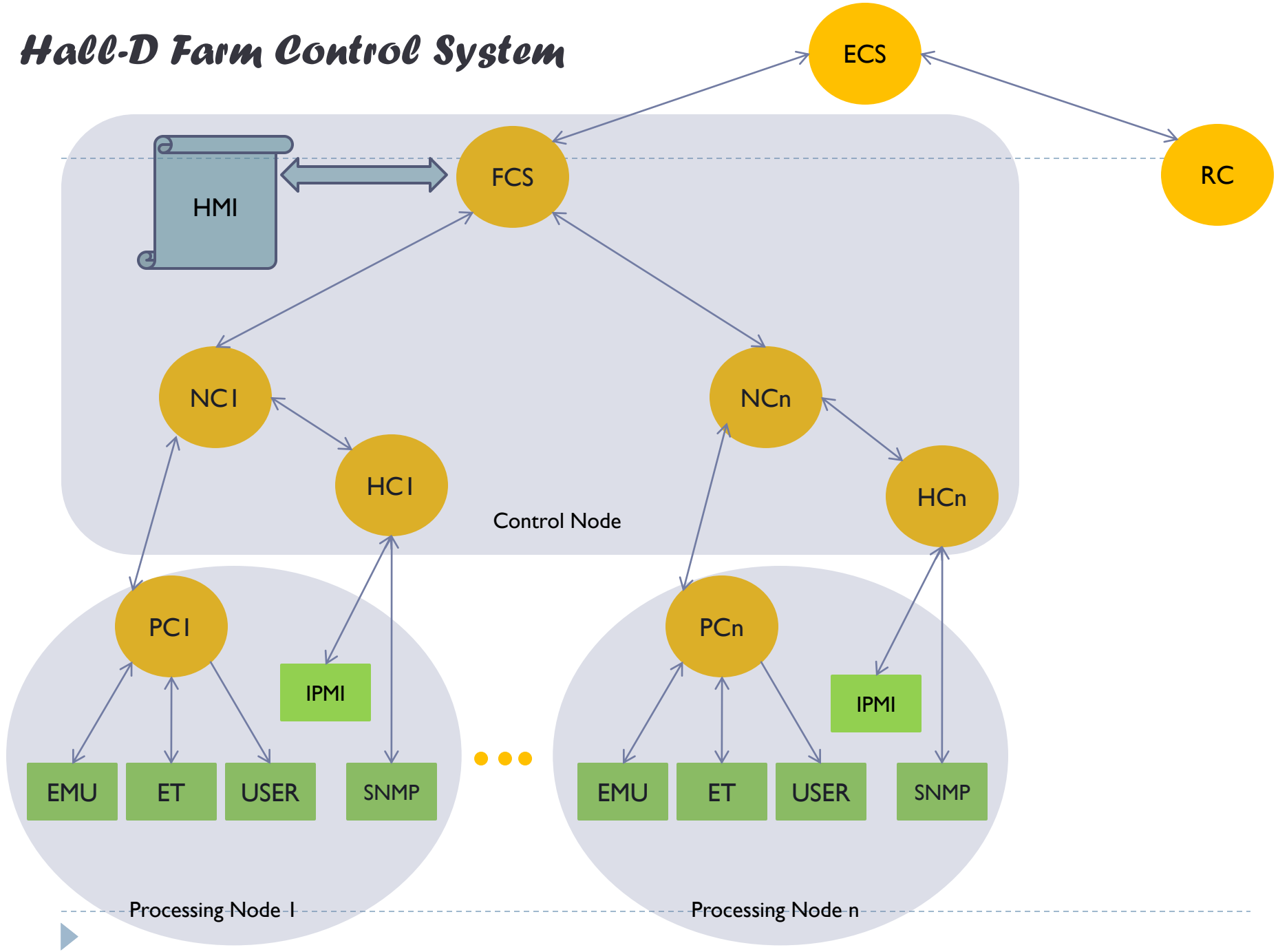
Progress Report

V. Gyurjyan for DAQ group

Project requirements

- Develop conceptual design before the end of the fiscal year.
- No concrete implementation is required.

Hall-D Farm Control System



Node Control Agents: states

- public class PCAgent extends ACAgent {
  private int monPeriod = 1000; // milli sec.
  private String stateBooted = "**booted**";
  private String stateConfigured = "**configured**";
  private String stateOff = "**off**";
  private String stateOn = "**on**";
  private String stateWarning = "**warning**";
  private String stateWarning = "**error**";

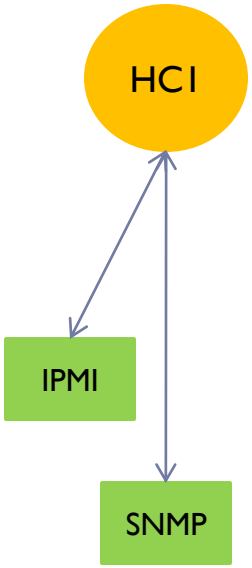
  private ProcessDrv pDrive;
  .....
- public class HCAgent extends ACAgent {

  private int monPeriod = 1000; // milli sec.
  private String stateBooted = "**booted**";
  private String stateConfigured = "**configured**";
  private String stateOff = "**off**";
  private String stateOn = "**on**";
  private String stateWarning = "**warning**";
  private String stateWarning = "**error**";

  private IpmiDrv iDrive;
  .....

Hardware Control Agent

- IPMI (Intelligent Platform management Interface) support
  - Technology considered a de-facto standard for computer system management.
  - Hardware chip known as BMC (board management controller) implements the core of IPMI.
  - OS kernel independent.
  - R/W access to sensors
  - Access to system event log
  - Configuring hardware watchdogs
  - Etc.
- SNMP (Simple Network Management Protocol) support
  - Requires hardware specific MIBs
  - User selected set of OIDs to control/monitor hardware
  - Supports SNMP API: set, get, getnext, trap



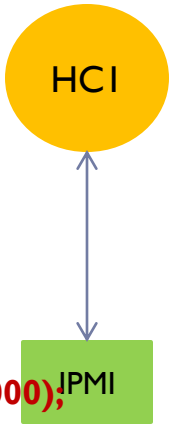
Hardware Control Agent: Initialization

```
@Override
public boolean cL_setup(AComponent aComponent) {
    if(me.getName().equals(AConstants.udf)){
        me = aComponent;
    }
    me.setState(stateBooted);
    // ask platform to report the password necessary to
    // open an ipmi connection to the hardware. Assumed
    // that the user name used for ipmi communication is root.
    ArrayList<cMsgPayloadItem> al = new ArrayList<cMsgPayloadItem>();
    try {
        al.add(new cMsgPayloadItem(AConstants.NODE,myName));
        al.add(new cMsgPayloadItem(AConstants.USERNAME,"root"));
```

```
cMsgMessage secM =
p2pSend(myConfig.getPlatformName(),AConstants.PlatformControlGetSecretKey,al,1000);
cMsgMessage pasM =
p2pSend(myConfig.getPlatformName(),AConstants.PlatformControlGetPassword,al,1000);
```

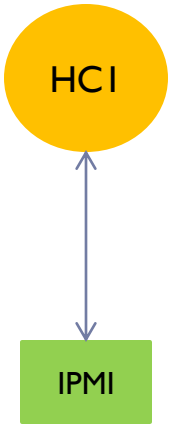
```
        if(secM!=null && pasM!=null &&
                secM.getByteArray()!=null && secM.getByteArray().length>0 &&
                pasM.getByteArray()!=null && pasM.getByteArray().length>0){
```

```
            iDrive.init(myName,"root",secM.getByteArray(),pasM.getByteArray());

            ……
```



Hardware Control Agent: payload

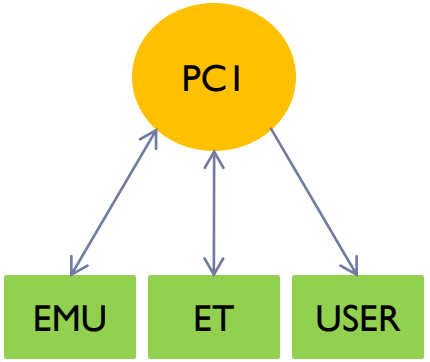
- mb.t_amb
- mb.v_bat
- mb.v_+3v3stby
- mb.v_+3v3
- mb.v_+5v
- mb.v_+12v
- mb.v_-12v
- mb.v_+2v5core
- mb.v_+1v8core
- mb.v_+1v2core
- fp.t_amb
- db.t_amb
- io.t_amb
- p0.t_core
- p0.v_+1v5
- p0.v_+2v5core
- p0.v_+1v25core



- p1.t_core
- p1.v_+1v5
- p1.v_+2v5core
- p1.v_+1v25core
- ft0.fm0.f0.speed
- ft0.fm1.f0.speed
- ft0.fm2.f0.speed
- ft1.fm0.f0.speed
- ft1.fm1.f0.speed
- ft1.fm2.f0.speed
- ft0.fm0.f1.speed
- ft0.fm1.f1.speed
- ft0.fm2.f1.speed
- ft1.fm0.f1.speed
- ft1.fm1.f1.speed
- ft1.fm2.f1.speed

Process Control Agent

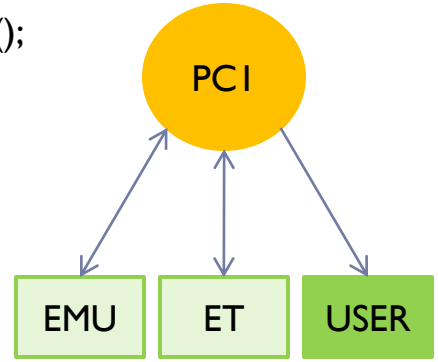
- Runs on the node
- CODA specific processes controls
  - ET
  - EMU
- User specific processes control
  - Unix process control (no intra-process communication necessary, i.e. no cMsg)



Process Control Agent: Initialization to control user processes

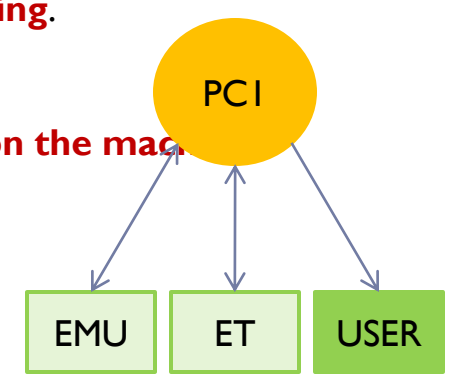
```
private void config(AComponent cmp){
    if(me.getName().equals(cmp.getName())){

        // read configuration file provided within the cool
        // options concept and fill the processes local hashMap
        if(cmp.getOption()!=null && !cmp.getOption().getConfigFile().equals(AConstants.udf)){
            String fileContent =
getProcessManager().getConfigFileContent(cmp.getOption().getConfigFile(),cmp);
            if(fileContent!=null){
                processes = new ConfigStringParser(fileContent).getProcesses();
                me.setState(stateConfigured);
```

```
<process>
    <processName></processName>
    <execString></execString>
    <isCritical></isCritical>
</process>
```



Process Control Agent: Payload

```
// owner of the process
me.addMonitoredData(pName+"_usr",new cMsgPayloadItem(pName+"_usr",pst.nextToken()));
//process ID of the process
        me.addMonitoredData(pName+"_pid",new cMsgPayloadItem(pName+"_pid",pst.nextToken()));
//CPU time used divided by the time the process has been running.
        me.addMonitoredData(pName+"_%cpu",new
cMsgPayloadItem(pName+"_%cpu",pst.nextToken()));
//ratio of the process resident set size to the physical memory on the mac
        me.addMonitoredData(pName+"_%mem",new
cMsgPayloadItem(pName+"_%mem",pst.nextToken()));
//virtual memory usage of entire process
        me.addMonitoredData(pName+"_vmu",new
cMsgPayloadItem(pName+"_vmu",pst.nextToken()));
//resident set size, the non-swapped physical memory that a task has used
        me.addMonitoredData(pName+"_rms",new
cMsgPayloadItem(pName+"_rms",pst.nextToken()));
        pst.nextToken();// controlling tty (terminal)
//multi-character process state
        me.addMonitoredData(pName+"_stat",new
cMsgPayloadItem(pName+"_stat",pst.nextToken()));
//starting time or date of the process
        me.addMonitoredData(pName+"_stt",new cMsgPayloadItem(pName+"_stt",pst.nextToken()));
//cumulative CPU time
        me.addMonitoredData(pName+"_time",new
cMsgPayloadItem(pName+"_time",pst.nextToken()));
//command with all its arguments
        String tmpS;
        StringBuffer tmpSb = new StringBuffer();
        while((tmpS = pst.nextToken())!=null){
```



Current Status

- Project Hpc-1.0
  - Developed IPMI, SNMP and UnixProcess driver classes
  - Prototyped and tested IPMI and SNMP based controls using sun fire 4100 DAQ server (megrez)
- Project HallDFnc-1.0
  - Conceptual design is completed
  - Process control agent is designed and programmed
  - Hardware control agent is designed and programmed
  - Supervisor agents design in progress
  - State machine design in progress
    - Node sub-control system state machine definition
    - Farm control system state machine definition
    - Synchronization between farm control system and run control system SMs