THE HATLEY/PIRBHAI MODEL

INFORMATION vs REAL-TIME SYSTEMS

Information Systems:

– Analysed & designed on basis of static data models.
– Emphasis on data storage, update and access.
– Timing issues (although important) are not critical.

Real-Time Systems:

– Analysed & designed with an emphasis on time related change in conditions and processes.
– Data storage is usually not encyclopedic and therefore does not require rigorous modelling.
THE HATLEY/PIRBHAI MODEL

REAL-TIME (EMBEDDED) SYSTEMS

DEFINITION:
• Any information processing activity or system which must respond to externally generated input stimuli within a finite & specified period of time (Young - 1982)

HARD REAL-TIME SYSTEMS:
• Are critically time dependent.
• Missed deadlines may be disastrous.

SOFT REAL-TIME SYSTEMS:
• Are time dependent.
• Missed deadlines not usually disastrous.

See Burns & Wellings (Ref 26)

ASPECTS FOR CONSIDERATION

• Size and Complexity
  Management, Staffing, Development Techniques, Estimating, Scheduling, Maintenance, Metrics etc.

• Algorithms
  Manipulation of real numbers. Mathematical modelling of system components that are under control (eg. PID).

• Reliability and Safety
  Exceptional conditions of operation. Risk of failure predictions. Provision of facilities to guarantee reliability.

• Concurrency
  Recognition of separate system elements that must work or be controlled in parallel.

• Time Dependency
  Separation of those system inputs that cause the generation of certain system outputs within specific time constraints.

• Interaction with other systems
THE HATLEY/PIRBHAI MODEL

S/W ELEMENTS OF REAL-TIME SYSTEMS

Real-Time Structured Analysis - Hat/Pir

DEMARCO MODEL REVISITED

• Assumes ideal technology:
  Instantaneous responses.
  Steady state operation.
  Time independent.

• De-emphasises control view:
  Decisions described at PSpec level only.
  Ignores process sequencing/activation.
  Little support for describing states.

• Emphasises data view:
  All processing is on data.
THE HATLEY/PIRBHAI MODEL

HATLEY/PIRBHAI MODEL

• Hatley/Pirbhai (Requirements) Model = DeMarco (Data Processing) Model + Control Model

• Control Model is used to specify:
  – Requirements that exhibit finite state machine behaviour.
  – What behaviour the Data Process Model must exhibit when the system is under the influence of particular external or internal conditions or operating modes.
  – Separate operational modes.
  – High level decisions that affect operational modes.

THE HATLEY/PIRBHAI MODEL

Requirements Specifications are constructed using:-

– Data Flow Diagrams & Control Flow Diagrams
  Showing data processes, data inputs and outputs.
  Showing control processing, control inputs and outputs.

– Requirements Dictionary
  Containing definitions of data inputs, outputs, stores and intermediate data plus definitions of control inputs, outputs, stores.

– Structured Language
  As for DeMarco Model.

– Control Specifications
  FSMs which map control inputs to control outputs &/or show control of data processes according to the control inputs.
Processes should be named with a short action clause summarising what is to be done to the input (data) in order to produce the output (data).

- Dataflows indicate the content and direction of flow of information (or materials) to and from processes, stores and terminators.
- Treat them as pipelines along which single or groups of data/material items of known content and nature can flow.
- Their names reflect their content - nouns or adjectives.
- They do not contain or represent dynamic behaviour - no verbal names.

- Stores represent dataflows or control flows that are frozen for an indeterminate time.
- The (data or control) information/materials they represent can be accessed at any time and in any order - non-destructively.
- Nouns and/or adjectives should be used - sometimes plural.
- Event stores (Ward/Mellor) queue incoming events - reads are destructive.

Terminators represent things that are external to the system, but which are important because they provide &/or receive system input and output.

- Control Specifications (CSpecs) are used to indicate finite state machine behaviour in the form of:
  - STATE TRANSITION DIAGRAMS
  - STATE EVENT MATRICES
  - DECISION TABLES
  - PROCESS ACTIVATION TABLES

- Control flows indicate the composition and direction of flow of control information to and from CSpecs, control stores and terminators.
- Treat them as pipelines along which single or groups of control items of known composition flow.
- Their names reflect their content - nouns or adjectives.
- They do not contain or represent dynamic behaviour - no verbal names.
- Event flows (Ward/Mellor) have similar meaning to control flows but are also used as prompts to enable/disable processes and they do not contain grouped elements.

See Hatley/Pirbhai (Ref 8) and Ward/Mellor (Ref 9)
Basic characteristics of CFDs:

- They are paired with Data Flow Diagrams
  - Have the same: Numbering, Levelling, Balancing, Parent/Child relationships.
- Show processes (same as DFDs) but not Data Flows.
  - Only concerned with elements that are affected by control.
  - Data processors are affected by high level control decisions but data flows are not.
- Show Control Flows and Stores
- Show Control Specifications (if any)
THE HATLEY/PIRBHAI MODEL

CONTROL FLOW DIAGRAM DECOMPOSITION

COMBINED DFD/CFD DECOMPOSITION
THE HATLEY/PIRBHAI MODEL

CONTROL FLOWS

- Represent pipelines that transport control information of known composition.
- Named according to content - use nouns or adjectives.
- Compositional elements can be primitive or collective.
- Primitive elements always consist of discrete values.

POSSIBLE SOURCES:

» External environment
» Control Specifications (CSpecs)
» Process Specifications - resulting from decisions made within PSpec about input data conditions.

POSSIBLE DESTINATIONS:

» External environment
» Control Specifications
» Not PSpecs - they only transform data inputs.
THE HATLEY/PIRBHAI MODEL

REQUIREMENTS DICTIONARY

- Contains definitions of both data flows & stores and control flows & stores.

- Notation used for all definitions is that of DeMarco.

- Primitive data flow definitions can be continuous or discrete in nature.

- Primitive control flow definitions must be discrete in nature.

Symbol | Meaning
--- | ---
= | composed of
This + That | This together with That
n( That )m | n to m iterations of That
[ This | That | Another | ... ] | select one of This or That or Another or ...
(This) | This is optional
" That " | literally the word That
* Note about this & that * | comment field and/or primitive element
This + That + Another + ... | key attribute of data entity (@ in TeamWork)
<That>This | That version of This (TeamWork only)
THE HATLEY/PIRBHAI MODEL

CONTROL FLOW DEFINITION EXAMPLES

Door_Position = [“Open” | “Closed”]
* The only door positions of interest *
---------
Rate: Event-driven by operator.

Fan_Setting = [“Off” | “Low” | “Medium” | “High”]
* Various fan rotation speed settings *
---------
Rate: As required.

Received = [“True” | “False”]
* Indicator for acknowledgment of receipt of certain signals *
---------
Versions: <Start_Date>, <Start_Time>, <Duration_Time>

CONTROL FLOW DEFINITION EXAMPLES

Primitive definition:

Windows_Command =
[“Main” | “Accessories” | “Windows_Applications”]

Decomposable definition:

Windows_Command =
[Main | Accessories | Windows_Applications]
Main =
[“File_Manager” | “Control_Panel” | “Print_Manager” |
“Clipboard” | “DOS_Prompt” | “Windows_Setup”]
Accessories =
[“Notepad” | “Write” | “Clock” | “Calendar” | “Terminal”]
CONTROL FLOW DEFINITION EXAMPLES

Grouped definition:

File_Management_Activity =
   Main + File_Manager + File_Selection + (File_Operation) + ...

• Primitive control flow definitions are common with most systems.

• Decomposable definitions are less common.

• Grouped definitions are relatively uncommon.

CONTROL SPECIFICATIONS

Absolutes of Real-Time Systems:

– Behave predictably.
– Maintain history of events or system conditions.
– Change behaviour (predictably) according to event history (past and present).

• These properties imply that real-time systems must deal with a finite number of events and produce a finite number of outcomes in order to behave predictably.
THE HATLEY/PIRBHAI MODEL

CONTROL SPECIFICATIONS

Represent various finite state machines as follows:

- State Transition Diagrams (STD)
- State Event Matrices (SEM)
- Decision Tables (DT)
- Process Activation Tables (PAT)

- Finite State Machines (FSMs) can be used only when a finite number of inputs having a finite set of values can lead to a finite number of outputs (or set of actions) having a finite set of values.

CONTROL SPECIFICATIONS

CSpecs are best used for systems that exhibit:

- significant numbers of states or modes of operation.
- significant complexity in terms of elaborating decisions that affect or determine system configuration or modes of operation.
- significant numbers of inputs (events) that have no effective content and merely serve as triggers/signals for actions that affect operational modes (or states) of parts of the system.
- high level management control of large portions of the system.
THE HATLEY/PIRBHAI MODEL

CONTROL SPECIFICATIONS

The FSMs within CSpecs are best used to:

• elaborate system states in terms of externally observable behaviour - with STDs.
• specify states in terms of combinations of active and inactive processes - with STDs and PATs.
• specify sequences of high level actions that change operational modes - with STDs and PATs
• specify results of particular combinations of control input - with DTs and PATs
• map combinations of input control signals into an output signal - with DTs.

TYPES OF FSM USED WITHIN CSPECS

Two Categories:

COMBINATIONAL

– Outputs are uniquely determined by current inputs.
– Keeps no information of past events (no memory).
– Include Decision Tables & Process Activation Tables.

SEQUENTIAL

– Outputs determined using both current and previous inputs.
– Maintains information on specific system conditions (has memory).
– Include State Transition Diagrams & SEMs.
THE HATLEY/PIRBHAI MODEL

COMBINATIONS OF FSM TYPES IN CSPECS

HATLEY/PIRBHAI & SEQUENTIAL FSMs

Basic Characteristics:-

1 Use a hybrid form of Mealy and Moore models in an attempt to take advantage of both models.
   – Basically use Mealy model.
   – Additionally use Moore when it is (more) convenient.

2 Define states as being the mode or condition of system.

3 Output of SFSM includes activation of processes.

4 Actions associated with a transition continue in effect until the next transition.
Actions include activation of processes on associated DFD.

Events are represented with input Control Flow values.

When Process-1 is activated, it will remain active (in State-2) until Event-3 causes a transition from State-2.

Actions are represented with output Control Flow values or indicated Process Activations.

The State Event Matrix is an alternative form for an STD.

For Mealy form of SFSM cells of the matrix that correspond to a transition will contain the resulting action together with the name of the destination state.
The value of Event Decision is determined by the combination of the input values of Event-1, Event-2 and Event-3.

<table>
<thead>
<tr>
<th>Event-1</th>
<th>Event-2</th>
<th>Event-3</th>
<th>Event Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete Value-1</td>
<td>Discrete Value-2</td>
<td>Discrete Value-3</td>
<td>Decision 1</td>
</tr>
<tr>
<td>Discrete Value-2</td>
<td>Discrete Value-3</td>
<td>Discrete Value-1</td>
<td>Decision 2</td>
</tr>
<tr>
<td>Discrete Value-1</td>
<td>Discrete Value-1</td>
<td>Discrete Value-2</td>
<td>Decision 2</td>
</tr>
</tbody>
</table>

Event decisions are represented with output Control Flow values.

The integer values indicate (de)activation of Process-1 and/or Process-2 and are determined by the combination of the input values of Event-1, Event-2 and Event-3.

<table>
<thead>
<tr>
<th>Event-1</th>
<th>Event-2</th>
<th>Event-3</th>
<th>Process 1</th>
<th>Process 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete Value-1</td>
<td>Discrete Value-2</td>
<td>Discrete Value-3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Discrete Value-2</td>
<td>Discrete Value-3</td>
<td>Discrete Value-1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Discrete Value-1</td>
<td>Discrete Value-2</td>
<td>Discrete Value-2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Control Actions are represented with indications of process activation in a table.

Only those processes being controlled are shown in the PAT.
THE HATLEY/PIRBHAI MODEL

CONTROL SPECIFICATIONS AS STDs

Combined DFD0/CFD0 showing data & control processing

- Accept Timing Data
- Calculate Average Temperature
- Accept Setpoint Temperature
- Date and Time
- Timing Data Received
- Door Position
- Time To Stop
- Time To Start
- Setpoint Temperature
- Average Temperature
- Temperature
- Temperature Tolerances
- Overall Setpoint Range
- Temperature Statistics
- Count Down To Time Out
- Oven States
- Processes under control

Software Improvements
©Copyright 1997 Software Improvements
THE HATLEY/PIRBHAI MODEL

CONTROL SPECIFICATIONS CONTAINING STDs

Pure Hatley/Pirbhai Std

Combined Mealy & Moore STD

Software Improvements

©Copyright 1997 Software Improvements
Use a DT when combinations of specific events result in the establishment of particular important conditions of which the system needs to be aware.

**IDLE WITH DOOR CLOSED**

- **Door Position = Open**
  - **Light Setting = Off**
  - **Fan Setting = Off**
  - **Lock Setting = Unlocked**
  - **Enable "Count Down To Time Out"**

- **Door Position = Closed**
  - **Light Setting = Off**
  - **Fan Setting = Off**
  - **Lock Setting = Unlocked**
  - **Enable "Count Down To Time Out"**

**IDLE WITH DOOR OPEN**

- **Door Position = Open**
  - **Light Setting = On**
  - **Fan Setting = High**
  - **Lock Setting = Unlocked**
  - **Enable "Count Down To Time Out"**

**READYWAITING**

- **Door Position = Open**
  - **Light Setting = On**
  - **Fan Setting = High**
  - **Enable "Count Down To Time Out"**

**READYIMMEDIATE**

- **Door Position = Open**
  - **Light Setting = On**
  - **Fan Setting = High**
  - **Enable "Count Down To Time Out"**

**RUNNING**

- **Door Position = Open**
  - **Light Setting = On**
  - **Fan Setting = High**
  - **Enable "Count Down To Time Out"**

**CONTROL SPECS CONTAINING DTs AND STDs**

- **Minimum Inputs Available**
  - **True**
  - **False**

- **Start Date**
  - **Received**
  - **Not Received**

- **Start Time**
  - **Received**
  - **Not Received**

- **Duration Time**
  - **Received**
  - **Not Received**

- **Setpoint Temperature**
  - **Received**
  - **Not Received**

**REAL-TIME STRUCTURED ANALYSIS - HAT/PIRBHAI MODEL**

**Software Improvements**

©Copyright 1997 Software Improvements
THE HATLEY/PIRBHAI MODEL

CONTROL SPECS CONTAINING DTs AND STDs

CSpec bars can be duplicated for diagrammatic convenience. Each duplicated CSpec bar represents the same SHEET.

---

CONTROL SPECS CONTAINING DTs AND STDs

SHEET-1 (s1) Oven State:

- IDLE WITH DOOR CLOSED
- READY_WAITING
- RUNNING

SHEET-0 (s0) Enough_Inputs

- "True", "Don't Care", "False"
- "True", "False", "Don't Care"
- "False", "True", "Don't Care"

Software Improvements
©Copyright 1997 Software Improvements
The DT part of a combined PAT and DT.

DTs should be used whenever complicated decisions about control input combinations need to be made in order to produce possible combinations of control outputs.
THE HATLEY/PIRBHAI MODEL

CONTROL SPECS CONTAINING PATs AND DTs

The PAT part of a combined PAT and DT.

```
<table>
<thead>
<tr>
<th>Door, Position</th>
<th>&lt;Start_Date&gt; Received</th>
<th>&lt;Start_Time&gt; Received</th>
<th>&lt;Duration&gt; Received</th>
<th>&lt;Temperature&gt; Received</th>
<th>Start</th>
<th>Stop</th>
<th>TIME</th>
<th>TIME</th>
<th>PROCESS 1</th>
<th>PROCESS 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Open&quot;</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>&quot;Closed&quot;</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>&quot;Open&quot;</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>&quot;Closed&quot;</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
```

PATs can be used whenever complicated decisions about control input combinations need to be made in order to produce possible combinations of control actions.

CONTROL SPECS CONTAINING PATs AND DTs

This PAT contains the transition information of the previous STD. Each row of this PAT could be viewed as representing a state - perhaps?

```
<table>
<thead>
<tr>
<th>Door, Position</th>
<th>&lt;Start_Date&gt; Received</th>
<th>&lt;Start_Time&gt; Received</th>
<th>&lt;Duration&gt; Received</th>
<th>&lt;Temperature&gt; Received</th>
<th>Start</th>
<th>Stop</th>
<th>TIME</th>
<th>TIME</th>
<th>PROCESS 1</th>
<th>PROCESS 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Open&quot;</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>&quot;Closed&quot;</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>&quot;Open&quot;</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>&quot;Closed&quot;</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
```

PATs are best used when it is necessary to view all of the possible process activations for a given CFD.

PATs should not be used in lieu of STDs because they tend to hide important state information.

Using a COMBINATIONAL FSM to represent state information creates ambiguity. Combinational FSMs should be used only in situations when past history is unimportant.

Real-Time Structured Analysis - Hat/Pir

MSE - ANU
Process (de)activations resulting from control actions within an STD can be made more obvious on the corresponding CFD by inclusion of a PAT.

The disadvantage is that some superficial control flows have to be created to connect the STD with the PAT.

Alternative to showing (de)activations of processes within STD is to provide action decisions (output control flows) that enter a PAT.
THE HATLEY/PIRBHAI MODEL

REQUIREMENTS MODEL SUMMARY

PROCESS MODEL

DATA FLOW DIAGRAMS

PROCESS SPECIFICATIONS

CONTROL FLOW DIAGRAMS

CONTROL SPECIFICATIONS

CONTROL MODEL

Data Input

Data Output

Control Input derived from Data Conditions

Control Input

Process Activations derived from Control Decisions

Control Output

Real-Time Structured Analysis - Hat/Pir
MSE - ANU

Adapted from Hatley/Pirbhai-1987 (Ref 8)

TIMING SPECIFICATIONS

System and Software Requirements Specifications:

– are the result of analysis.
– are fully documented statements of what a system or software product must achieve.
– should not contain descriptions of system structure or how to achieve particular requirements.
– become the starting point for design.

Timing Specifications therefore:

– are determined for overall (external) system behaviour.
– are not provided for particular (internal) processes since no design structure has yet been determined.
THE HATLEY/PIRBHAI MODEL

TIMING SPECIFICATIONS

There are two types of Timing Specification.

Timing Specifications - Repetition Rate

Output signal(s) that are required to be recomputed at a particular frequency.
THE HATLEY/PIRBHAI MODEL

TIMING SPECIFICATIONS - REPETITION RATE

REPETITION RATE:

– The required recomputation rate of signal outputs to the external environment.

– Is included with the description of the signal in the Requirements Dictionary.

• Example (from previous slide)

<table>
<thead>
<tr>
<th>Time_Remaining</th>
<th>&quot;The amount of time that remains before a run ends&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Units</td>
<td>Hours:Minutes:Seconds</td>
</tr>
<tr>
<td>Range</td>
<td>0:0:0 to 12:0:0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>1 second</td>
</tr>
<tr>
<td>Rate</td>
<td>Once per minute</td>
</tr>
</tbody>
</table>

TIMING SPECIFICATIONS - I>O RESPONSE TIME

INPUT-TO-OUTPUT RESPONSE TIME:

– A specification of allowable timing ranges (within which the system must respond) for each input event and the resulting output response(s).

– Can be included with each of the dictionary components that contribute to the various sets of event-responses.

– Is better to tabularise.
THE HATLEY/PIRBHAI MODEL

TIMING SPECIFICATIONS - I>O RESPONSE TIME

Input signals and corresponding output responses, some of which are time critical.

Table of Specified Input-to-Output Response Times

<table>
<thead>
<tr>
<th>EXTERNAL INPUT SIGNAL</th>
<th>EVENT</th>
<th>EXTERNAL OUTPUT SIGNAL</th>
<th>EVENT</th>
<th>RESPONSE TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature</td>
<td>Temperature value is read</td>
<td>Heat_Setting</td>
<td>Heat_Setting value is issued</td>
<td>&lt; 1 second</td>
</tr>
<tr>
<td>Temperature outside tolerance</td>
<td>Temperature_Overrun</td>
<td>Temperature_Overrun value is issued</td>
<td></td>
<td>&lt; 0.5 second</td>
</tr>
<tr>
<td>Door_Position</td>
<td>Door is closed</td>
<td>Fan_Setting, Light_Setting</td>
<td>Fan and Light are turned off.</td>
<td>&lt; 50 milli secs</td>
</tr>
<tr>
<td>Timing_Data</td>
<td>Timing data is entered</td>
<td>&lt;Accepted&gt; Timing_Data</td>
<td>Timing_Data value is accepted</td>
<td>Not critical.</td>
</tr>
</tbody>
</table>
GUIDELINES FOR USING HATLEY/PIRBHAI

There are no strict step-by-step procedures. However, the following guidelines are useful:-

• Construct an Event/Action list to more easily:
  – Identify processes that transform input data (or material) flows into output data flows.
  – Isolate those inputs that directly control the internal operation of the system.
  – Isolate those data inputs that are used to make decisions about controlling parts of the system.
  – Identify various operating modes together with the inputs that cause operational mode changes.

• Use Event/Action list to separate data signals from control signals.
  – Data signals usually have some content in the form of a range of non-discrete values.
  – Control signals will always have discrete values and tend to be used to trigger some action.
    Activate, enable, engage, execute, stop/start, trigger, toggle.
  – Some signals may have discrete values but not be part of the control model - they are then data signals.
### GUIDELINES FOR USING HATLEY/PIRBHAI

- Use Event/Action list to establish processes first.
  - Because it is preferable to identify what is to be controlled before isolating control-type activities.
    - Establishing what is to be controlled before thinking too much about control issues ensures less confusion and faster progress toward a good set of requirements specifications.
  - Because the control model is dependent upon the process model in most systems.
  - By applying DeMarco approach to construct the essential functional abstractions and decompositions.
    - This can be done by transforming the event/action list into a set of event/action diagrams before composing a set of DFDs.

- Keep control issues at a high-level.
  - Because it is important to determine system operating modes (states).
  - Because low-level control issues tend to be more strongly related to implementation issues.
    - Overuse of Hatley/Pirbhai Model at lower levels of functional decomposition can obscure specifications with implementation decisions.
  - Because it will aid architectural decisions during the design phase.
THE HATLEY/PIRBHAI MODEL

HATLEY/PIRBHAI & SYSTEM DEVELOPMENT

SOFTWARE DEVELOPMENT

SOFTWARE REQUIREMENTS ANALYSIS

HARDWARE DEVELOPMENT

SYSTEM REQUIREMENTS COLLECTION

SYSTEM REQUIREMENTS ANALYSIS

SYSTEM DESIGN

SOFTWARE DEVELOPMENT

SOFTWARE REQUIREMENTS ANALYSIS

PRELIMINARY DESIGN

DETAILED DESIGN

CODING AND UNIT TESTING

SOFTWARE INTEGRATION TESTING

SYSTEM INTEGRATION AND TESTING

PRELIMINARY DESIGN

DETAILED DESIGN

FABRICATION

HWCI TESTING

SYSTEM REQUIREMENTS ANALYSIS

HARDWARE DEVELOPMENT

SOFTWARE DEVELOPMENT

SYSTEM DESIGN

SOFTWARE REQUIREMENTS ANALYSIS

PRELIMINARY DESIGN

DETAILED DESIGN

CODING AND UNIT TESTING

SOFTWARE INTEGRATION TESTING

SYSTEM INTEGRATION AND TESTING

PRELIMINARY DESIGN

DETAILED DESIGN

FABRICATION

HWCI TESTING

SYSTEM REQUIREMENTS ANALYSIS

HARDWARE DEVELOPMENT

SOFTWARE DEVELOPMENT

SYSTEM DESIGN

SOFTWARE REQUIREMENTS ANALYSIS

PRELIMINARY DESIGN

DETAILED DESIGN

CODING AND UNIT TESTING

SOFTWARE INTEGRATION TESTING

SYSTEM INTEGRATION AND TESTING

PRELIMINARY DESIGN

DETAILED DESIGN

FABRICATION

HWCI TESTING

SYSTEM REQUIREMENTS ANALYSIS

HARDWARE DEVELOPMENT

SOFTWARE DEVELOPMENT

SYSTEM DESIGN

SOFTWARE REQUIREMENTS ANALYSIS

PRELIMINARY DESIGN

DETAILED DESIGN

CODING AND UNIT TESTING

SOFTWARE INTEGRATION TESTING

SYSTEM INTEGRATION AND TESTING

PRELIMINARY DESIGN

DETAILED DESIGN

FABRICATION

HWCI TESTING

REAL-TIME STRUCTURED ANALYSIS - HAT/Pir

MSE - ANU