**IIIT_H/ Spring 2015**

**Internals of Middleware Systems**

# Project: Design Outline

(First version due by- Saturday, this week)

Use same template for group design doc, an dthe three team design documents.

---

1. Intro to the Project (1-2 parah)
Functional overview of Team's module (1-2 parah. & Block diagram)

**2. Test cases (will clarify what your module will do. Build on your operations and usecases listed in requirements doc)**

2.1Test cases- used to test the team's module

- List the cases- simulate what the other modules/users can do with your module
- At least 10 test scenarios. Give name, and a brief 3-5 line description. Inputs. Outputs. Invocation mechanism. Expected behavior. Other external considerations (say, when to start a container or how many to start and such- as needed by the test)
- This shd be based on the usecases listed in the Team's reqt document

2.2Overall project test cases (relevant to the module)

- Integration testing
- How & what to test, after integrating with other modules
- This shd be based on overall use-cases listed in Group reqts doc

**3. Solution design considerations**

(Group level- to be discussed by all three teams, and captured in all three team-design documents).

3.1 Design big picture

- one diagram listing all components in the solutions and interactions therein

3.2 Environment to be used

- what (OS, App server set al)
- where (in which module or for what function)
- nature of interactions (if any)
- more..

3.3 Technologies to be used

- what (Node.JS,. ++)
- why
- how

3.4 Approach for Device gateways

3.5 Devices type/interfaces and information structure

3.6 Devices registry & repository

3.7 Approach to get device information (event streams)

3.8 Device info proxy server

3.9 Communication overview

3.10 Filter Server

3.11 Logic server

3.12 Interactions between modules

3.13 Wire and file formats

3.14 User's view (end UI on mobile)

**5. User's view of system**

- What is an application- for each type.
- What are the files (config, et al)
- How are analytics rules defined
- Structure of the files
- How is it deployed/setup. And where (on which module/process/location et al)
- How to start & stop
- How to access/invoke
- More..?

User interactions

- How will the user use this system
- what & how to configure
    - (processes, config files, et al)
- What are How to deploy

**6. Key Data structures**

Definition & other details of deice types
Solution level data structures
Wire formats
Persisted data (registry, rules, et al)
APIs- interaction data objects
App Logic definitions
User's UI (mobile)

**7. Interactions & Interfaces**

APIs
(Should include more detail than what was in Team's reqt document)
- User interactions
- Module to Module interactions
- File/Wire format definitions
  # list the nature of interaction (between what & what, and why)
  # describe the form & content (say what form of communication, protocols if any, standards in use if any, structure of data to be exchanged
- Inter-module interface APIs

8. Persistence

(Should include more detail than what was in Team's reqt document)

- config info- what needs to be saved
- transient state (services location & status , et al)
- more?

9. The three modules; that the three team will work on

Internal design overview (additional details like above, for each model)

**Low Level Design (for each of the 3 modules)**

1. Lifecycle of the module
   - How will the module/its components be started, monitored and stopped (if applicable).
   - How will any user's application components need be made available/deployed/setup on this module.
   - Any other considerations (based on the module's functionality)

2. List the sub modules

A block diagram
Describe the interactions among sub-modules

3. Brief overview of each sub module
   - For each sub module:
     #(parts of the whole solution
     #give a meaningful name for each part)
   - Describe the functionality of the module (7-10, bullet list)
   - A brief block diagram of the internals (constituents)
   - List the interactions (with which other module, external entities, file or network, comm. Framework et al)
   - APIs & Classes .. and

4. Interactions between sub modules
   - Interactions among the sub-modules
   - APIs & .. such
   - More..? (describe as relevant)

5. Interactions between other modules

(describe as relevant)

**6. Other design considerations**

(describe as relevant)