Date:

State Transition Diagram

- Data Modelling and Functional modelling exhibited in behavioural modelling.
- Behaviour → Human
  - ↳ System
  - → Reactive (Runtime)
  - → Observable in a given time.
- Attitude
  - → Permanent
  - → depicted through behaviour
  - → develops throughout life.

- System Behaviour
  - → Case Studies
  - → Fixed (until changed by us)
- Human Behaviour
  - → Changes during runtime

- AI increases options of behaviour.

Elements:

- 4 elements of STD.

① State

- Observed at a particular time
- Verbal phrase
- E.g lecturing, selecting, generating invoice.
- Represented by box.

② State Transition

- Shows direction of state from source to destination
- ⟶

③ Event

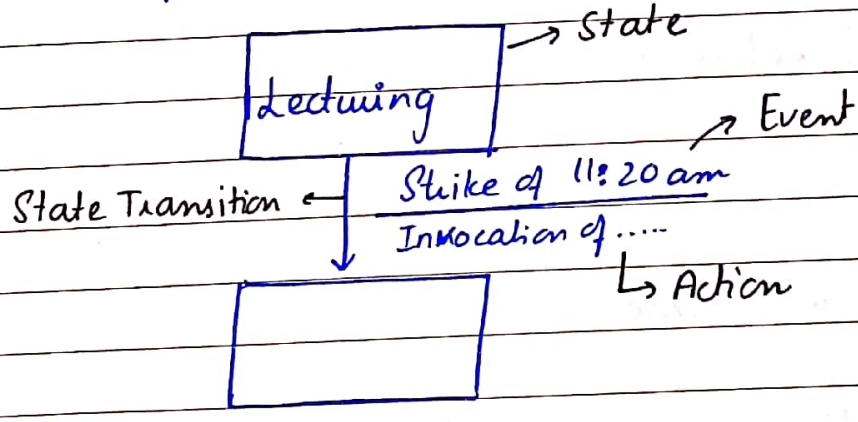
- Occurence that initiates state transition.
- Call of Function
- E.g striking of 11:20, leaving of teacher.

Date: _____

- Information production and consumption are main events.
- E.g output devices : keyboard, mouse, etc
- written on top of straight line
- Name of event → Noun Phrase.

④ **Action:**

- Consequences of making a transition
- E.g packing bags after teacher leaves.
- Once completed then second state starts
- written as:
  - → Invocation of
  - → call of
  - → Calling of
- Actions are processes in DFD



- First time interaction of external entities with system → Start.
- After structured analysis → document generated → called SRS.

SRS:

- Functional and non-functional requirements stated

Date: _____

- Completed and Signed.
- This document then serves as basis for all works onwards.
- Formally change in requirements can occur (the whole process is repeated)

Design:

→ Focus on finding answer of HOW? → In analysis focus is on finding answer of what?
→ Implementation of SRS.

① Data Design

- ERD and its documentation (Data Modelling) is input.
- Data Structures → hold vessels that hold data during runtime.
  → Trees, Stack, Arrays, graph, queue, list, hash.
  → Business logic layer b/w data base and your software
  → Data Structure temporarily hold data needed from data base.

=> Identification and implementation of appropriate data structure according to the requirements is called data design.

② Architectural Design

=> Placement of things according to the requirements

- 4 things are to be evaluated where to place :
  1. Components or modules (Process / Function)
  2. Inputs
  3. Outputs
  4. Decision making

Date:

- On higher level decision making > process
- Decision making is max at top level of hierarchy and processing is max at lower level of hierarchy.
- Architecture is also called program structure, controlled hierarchy, call and ~~ret~~ return architecture.

③ Component Level Design
- Detail of architecture
- Logic is transformed to technical language
- Main input is taken from mini specification

④ User interface Design
- Decide number of external entities
- Controls provided.

Fundamental Concepts
- Problems of Size and Complexity are solved by these concepts.
- ~~Abstrati~~ Abstraction and refinement → divide and rule.

Architectual Design

Architecture is also called
- → Program Structure
- → Control Hierarchy
- → Call and Return
- → Architectual Style

Transform flow:
Pure Processing
input → process → output

Date:

Transactional Flow:
- Decision making flow — action paths
- Input is evaluated → it has paths → one path is selected on basis of evaluation
- Evaluater → Transactional Centre / ~~Deis~~ Decision Maker / Dispatcher
- Transform flows start from action paths.

Example: String Conversion Level 1 DFD
- 4 transformational flows
- 1, 3, 4, 5 process → transformational flow.
- 1, 2 process → transactional flow

Functional independency
- Dependency
- Association ✓
  - ↳ arrangment but no dependency

Situation → dependent → proactive measures → decrease or remove dependency.

Two types

① Coupling
→ How much dependency module have? / How much work is done by parts ~~team~~ of module?

② Cohesion
→ How much work module is doing itself?

=> We need to maintain high cohesion and low coupling
- We have to decrease dependency.
- Module → functionality should be independent.

Date:

- Design principles can be violated to achieve high cohesion and low coupling.

Component Level Design

- An option → Flowchart.
- Closest activity to coding

① Sequence State.
② Conditional Statem.
③ Repititional Statem.

- Main input → Mini Specification
  ① Identifying mini specification
  ② Transform mini specification to programming constucts

- MS Visio
  ↳ Tool for diagrams

↳ also called PDL → Program Design Language

* • Technically pseudocode and algorithm are dif.
  ↳ Programming lang and keywords involved
  ↳ plane logic
  ↳ no syntax of any Programming lag.

⇒ PDL → Mixture of any language and plane English

- P.L can itself be used to design Component level
- Decision table → Not an option rather a support to develop component level design.
- 2^n → no. of combination n → no. of conditions
  Cases against each combination
- Decision table → tells total no. of combination
  ↳ and g cases against each combination

Condition	Combination of Conditions
-	-
Actions	Combination of actions
-	-

Date:

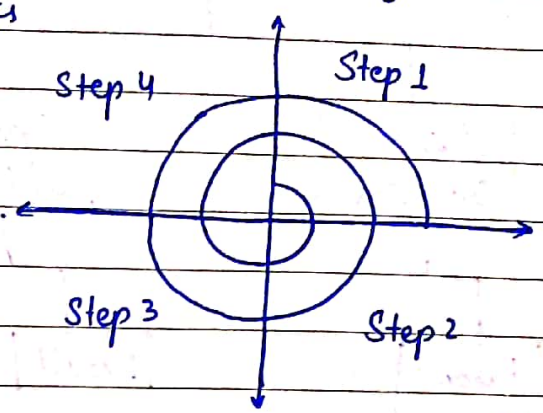
User Interface Design Most important

- Called HCI → Human Computer Interaction
- Aestheticism needed
- Identify user and their related tasks

① Site map
② Story board

Process: (to design User Interface)

Step 1: User, task and environmental analysis



→ Identification of users (external entities)
↳ First visible at context level dfd.

→ Task analysis
- tasks of users

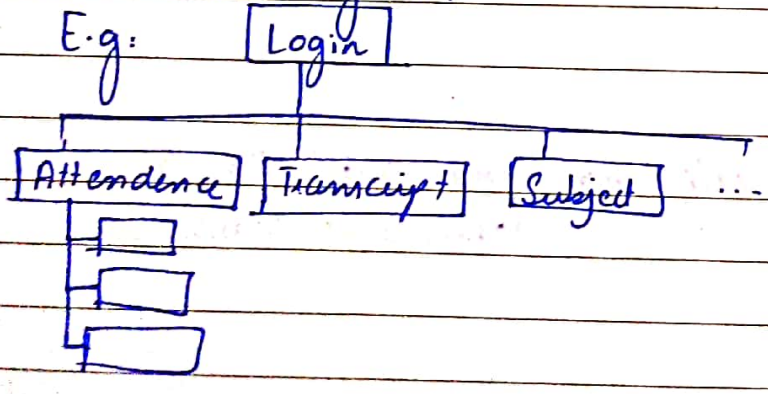
→ Environment analysis
- Environment given by STD.
- Consider and mention environmental factors.

Step 2: Interface design

2 things

① Site Map
→ is like architecture
→ Hierarchy of all pages of a website given.

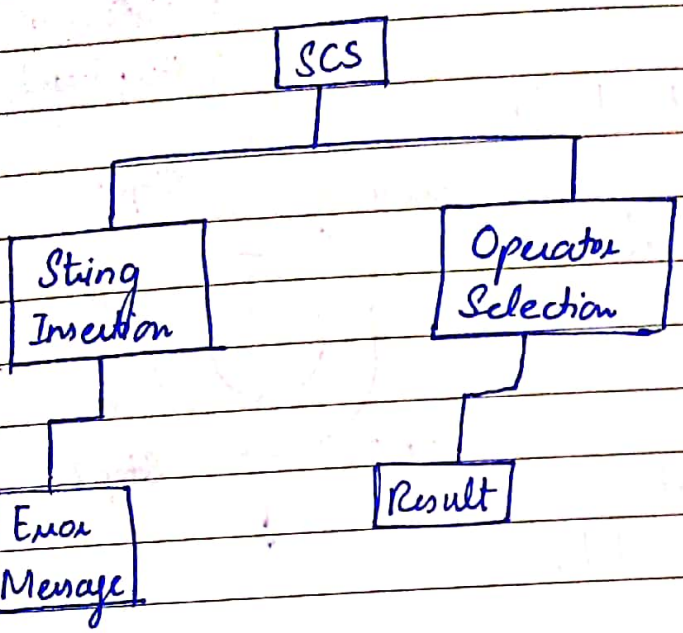
E.g:



② Story board
→ is like Component level Design.
→ detailes of all screens are decided
→ Functionalities (Controles)
- Button - Textbox
- Radio button - Check box
- Drop down list - dialogue box
- Label - Banner (search)

Date: _____

Site map of String Conversion System.

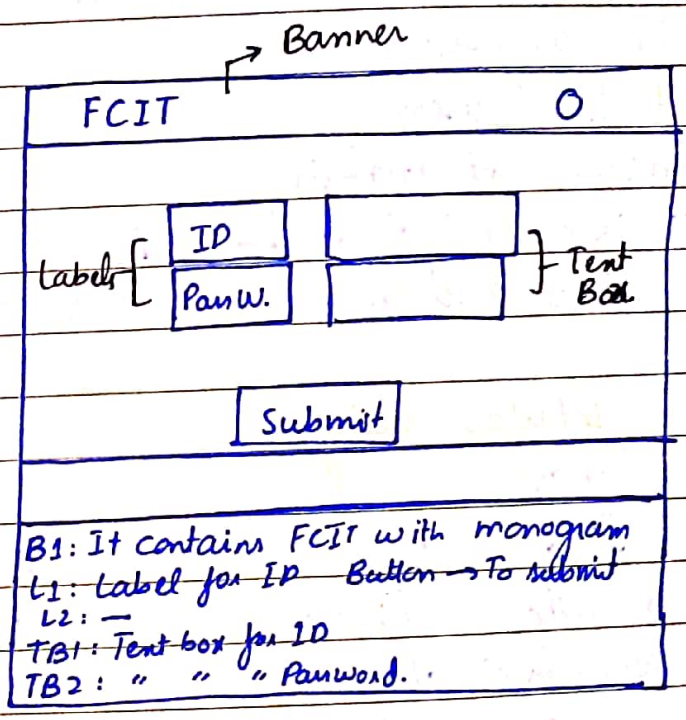


=> Identification and hierarchy of screens.

Story Board of CMS

- 2 parts.

- For implementation
  - → Technical Skills
  - → Aesthetic sense needed



- Validation → Testing
- Verification
  - → Whatever you are doing is right.
  - → Testing product or result
  - → Testing process

=> You will have to maintain validation and verification to achieve product
↳ should be consistent.

Date:

Step 3: Implementation
Step 4: Interface validation.
- If validation report not zero against iteration starts.
- Report of testing is made

Objected Oriented Analysis & Design

Analysis:
① Use Case Modelling
- High level Use case
- Analysis level Use Case
- Use Case Description (DOC)

② Domain Model

Design:
① System Sequence Diagram
② Sequence diagram
③ Final → Design Class Diagram (DCD)

- Dif of Structured and Object oriented approach
=> No protection rights in Structured approach
=> Simultaneous control of data and function in oop.
=> Size and time increase in Structured approach

Use Case Modelling

- Start with list of requirements
- Step 1: Collecting Requirements.
- Way of Showing Req Requirements → Use Case Modelling.
↳ Resembles process

=> Goal in the system of the Actor (User)
- Goal should be SMART
→ Specific → Measureable
→ Achievable → Realistic
→ Time Bound

Date:

- Goal must be complete, independent.
- First stage of Use Case Modelling is a Complete Goal
- Short term, mid term, long term goals.
- Goal → main
  - ↳ sub

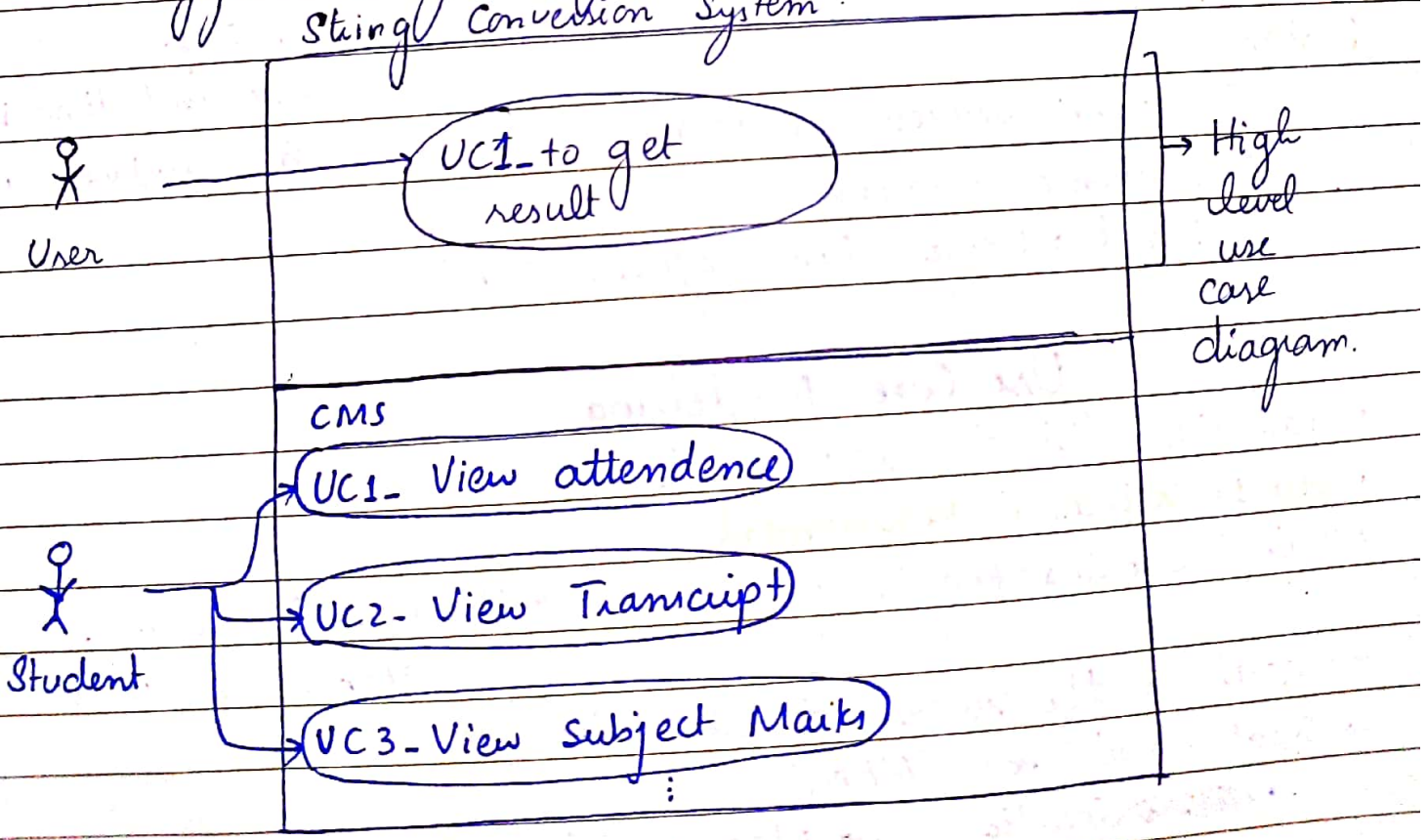
③ Scenario
④ System

Example: String Conversion System

~~① High~~
~~② Analysis~~

(- High level usecase diagram)

- 1 actor
- Identify main goals first.



Date: _____

- Types of actors
  1. Primary actor
  2. Secondary actor
  3. Offstage actor

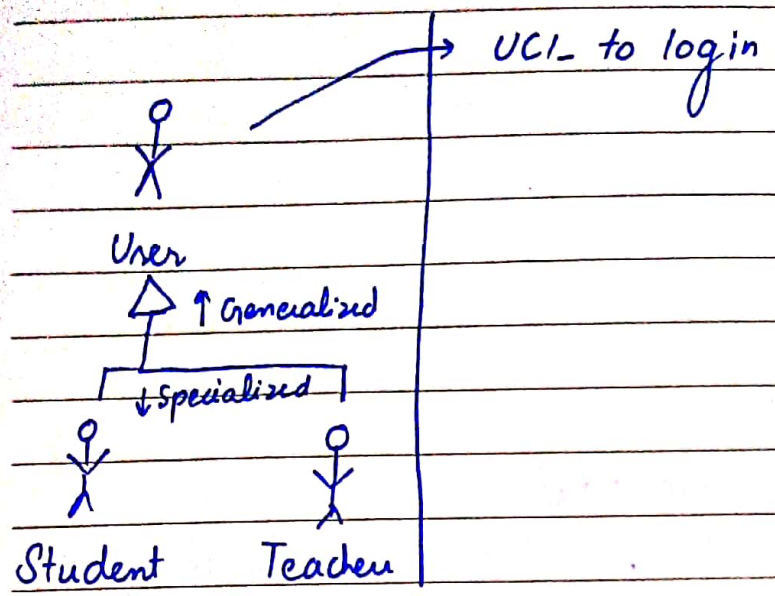
1. Primary actor
   - Primary goal
   - Interact with system with some goal
   - E.g student, teacher, state officer, degree coordinators, etc

2. Secondary actor
   - Support primary actor in achieving their goals.
   - E.g in viewing transcipt for student teacher is the secondary actor

3. Offstage actor
   - No work in system
   - Either gives or takes some information.
   - E.g Parents in our education.
     Role of Punjab University with this department.

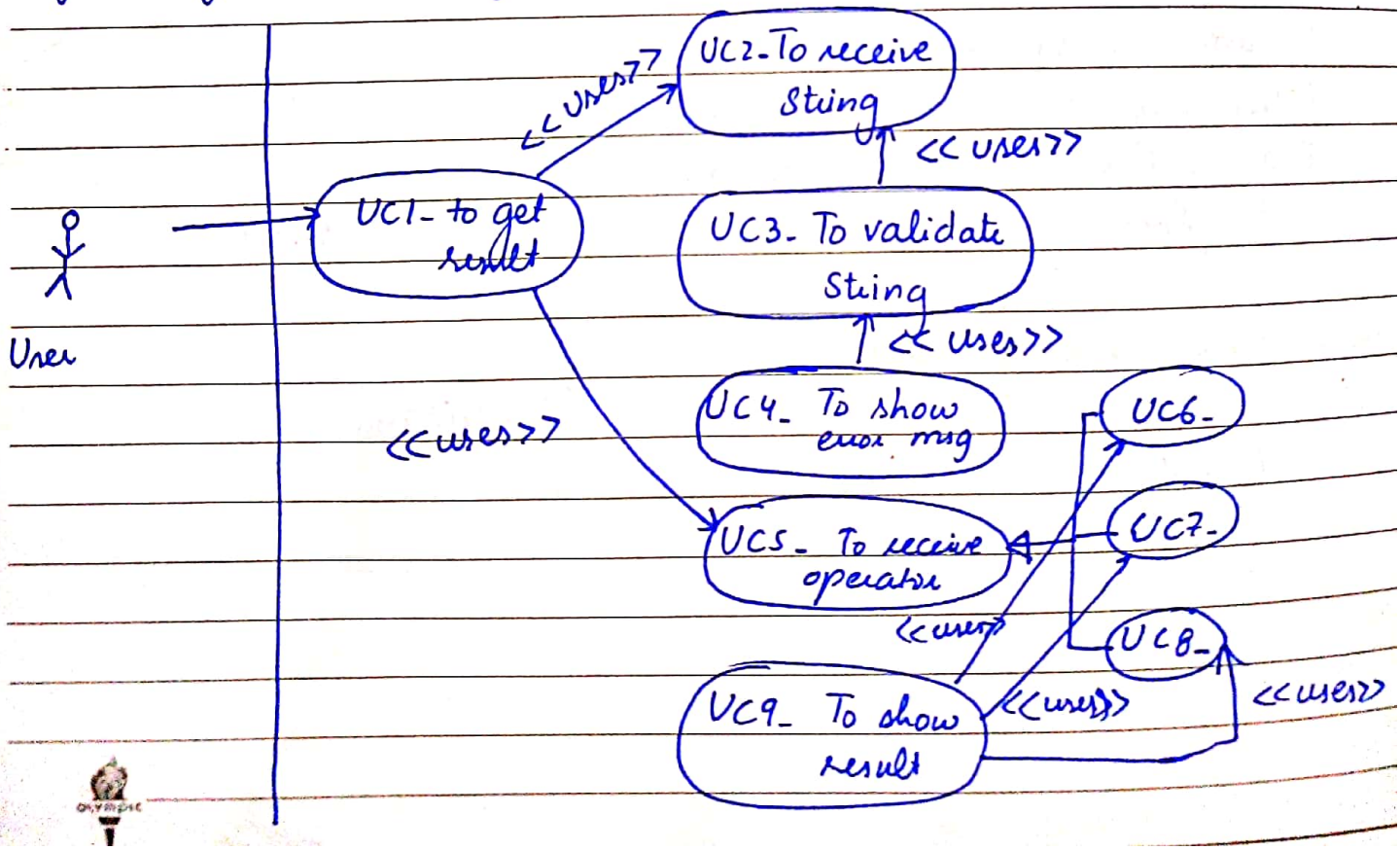
Relationship among actors:
- Generalization - Specialization (Inheritance)
  Parent Child

Date:



- Analysis level Usecase diagram
  - Keeping high level diagram under consideration.
  - Extend every high level taken
  - Make list of subgoals to help.
  - Processes of level 1 dfd are sub goals

(E.g) String Conversion System.



Date: _____

- Relationship b/w Usecases 2
  1. Uses / Include (Mandatory) <<Uses>> ⟶ or <<include>> ⟶
  2. Extend / Generalization - Specialization (Optional)

- From user ⟶ direct relation.

Description of Usecase

- Brief
- Casual
- Fully dressed

] - 3 formats.

- 1 paragraph explanation ⟶ Brief
- Multiple paragraph ⟶ casual
- Proper heads ⟶ Fully dressed

Fully dressed.

1. Preconditions
   (User must have valid user ID and password)
2. Post Condition / Success Guarantee.
   (User login)
3. Main Sucess Scenario
4. Extensions (Alternative flow)
   (Error msg or asked to enter ID and password again
5. Special Requirements (optional)
   (Multiple language facility)
6. Technology and data variation list
   (E.g any dispensor machine, card reader)
7. Open issues
   - laws, rules