unit 4

User Application Analysis

→ The purpose of analysis is to understand the problem so that the correct design can be constructed.

Application interaction model

We can construct Application interaction model.

① Determine the system boundary
- → Scope of an application should known
- → Correct system boundary
- → you don't take human as part of system.
- → During Analysis we determine the purpose of system
- → During design you can change the internal implementation of the system as long as you maintain the external behavior.

② finding actors and use case.

→ After Boundary, we identify actors. (human, External device)

→ And we find the use-case, scenario, how diff. way the actor can interact.

③ finding initial and final event.

↳ Initial event is an occurence that triggers a chain of activity
↳ final event → last event.

④ preparing for normal scenario
It illustrate the major interaction, external ~~data~~ display format and info. Exchange.

⑤ Adding variation and EXception scenarios.
Consider omitted input/output, error cases

⑥ finding external events.
all inputs, decisions, interrupts and interaction to or from user or External device.

⑦ Preparing activity diagrams for complex use case

⑧ organizing actor and use cases.

⑨ Checking against the domain class model.

Application class Model

① Application class Model define :

① Application class Model define the application itself rather than real-world objects that the application acts on.

→ Most application classes are Computer-oriented and define the way that the users perceive the application.

- Specify the user interface
- Define boundary phase
- Determine controllers
- Check against the interaction model.

Application State Model.

Application State Model focuses on application classes and augments the domain state model.

- Determine Application classes with states
- Finding events
- Building state diagram
- Check against other state diagram
- check against the class model
- check against the interaction model.

Adding operation

- define operation.
  - Operation from class model
  - Operation from use cases.
  - Shopping list operation
  - Simplifying operations.

Overview of System design.

① Estimating Performance.

→ to determine the feasibility of a Software System. during planning phase.

① prediction and approximation of proposed System performance

② then. performance estimation process.

② Making a Reuse plan.

① with the help of existing things.

② By Means of producing reusable new things

* Libraries, * frameworks * pattern.

Libraries- Collection of classes.

1. Efficient
2. Generic
3. Consistence
4. Completeness
5. Coherence.
6. Extensibility.

framework. - a complete architecture of a Software System application.

↳ it expand by designer

1. Black box framework → Internal structure is hidden from end user.
2. White box framework. Internal structure known to end user

pattern. - Established and confirmed solution to a general problem.

→ They are already applied on ancient problem

Organising System into Subsystem.

Divide the system into no. of pieces for better understanding called subsystem.

- A subsystem is group of classes, association, operation, events and constraints.

- Each subsystem has well-structured and defined interface to the rest of the system.

Layers (horizontal)	&	Partitions (vertical).
* Diff. level of abstraction.		Identical and same level of abstraction.
* peer to peer relationship		* Client server relationship
* self governing		* dependent on other layer.
* client server relationship		* peer to peer
* dependent on other layer.		* self governing

Identifying

Inherent concurrency in a problem.

* Two object are inherently concurrent if they receive events at the same time and donot interact.

If the events not are synchronised the two object can not be on single thread of control

* with the help of State diagram several object involved in a scenario can be congregated into a thread of control.
* only a single object can active on thread of control.

Allocation of Subsystem to hardware

- Subsystem ~~to~~ are allocated to processors and tasks.
- Estimate hardware resource requirement to decide if some subsystem should be implemented in hardware rather then in software.
- Allocate tasks to processor and determine physical connectivity.

---

* Data storage Management
* file, data structure, database
* DBMS
* RDBMS
* OODBMS - domain app have wide variety of data.

Global resource handling.

physical units	Processors, tape drivers,
Space	Mouse button, workstation
Logical Names	object ID, filename,
Access to shared Data	Data base

Scanned with CamScanner

read of control.

Choosing Software control strategy

① External control.
↳ flow of event between object from outside.
1. ~~procedure~~ driven control
control exist with the Software program coding.
2. Event driven control
→ where measurement method is inherently
→ event based in nature
3. Concurrent control
transaction executed concurrently.

② Internal control
control ~~flow~~ flow ~~etc~~ comprised by a process

Handling Boundary Conditions
① System should initialize the parameters constants and variable.
② Termination — ~~unrestricted~~.
③ ~~fail~~ failure — unintended closure of System

Setting Trade off priorities.

→ designer should define the significance of the several criteria as a guide for creation & design tradeoffs

→ launches the priorities for constructing respective software system arrangements.

Selecting an Architectural style.

① Batch transformation.

* info. transformation is executed once on → complete input dataset.
* Breakdown the complete transformation into stages.
* Expansion of each phase or level.
* reorganise the ultimate pipeline to optimization

② Continuous transformation.

→ output depend on varying inputs.
→ An uninterrupted transformation updates system outputs frequently.

③ Interactive interface.

→ interaction among agents and the system itself

→ predefined classes should be used for communication and coordination amongst the external agents.

(4) Dynamic Simulation
- designing and modeling real world objects
→ video game

(5) Real Time System.
- interactive systems with tight time constraints on actions.

(6) Transaction Manager
* retrieval and storage of data
* It deal with several User who write and read data at same time.

Component diagram.

→ used represent the physical aspects of System.

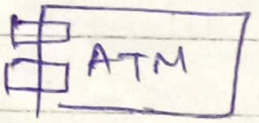
↙

tables, libraries, file, documents.

Components of Components diagram,

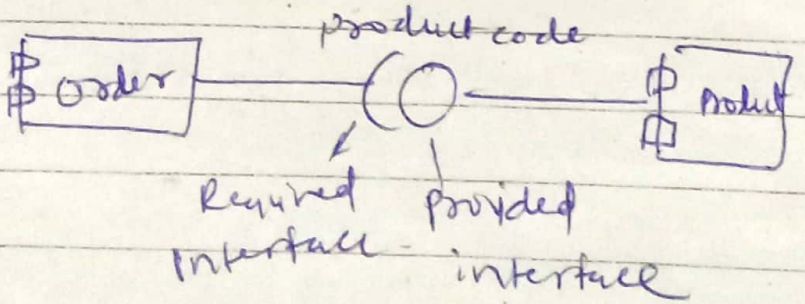
① Component.

↳ physical and expendable part of a System.



② Interface.

- Collection of operation.



~~Relation~~

Deployment diagram.

Deployment diagram trying represent visualize topology or organisation of physical component of system upon which system software is deploy for execution.

* Static deployment view of system.

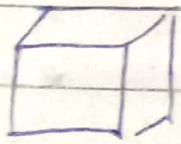
elements.

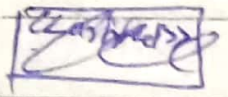
① Node →

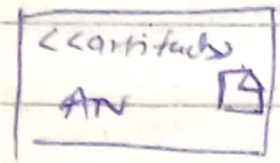
Artifacts ⇒ files, table, data.

deploy

large computational unit.







relationship

① Association ——

② dependency - - - - →