Agile meaning → able to move quickly & easily.

PAGE NO.:
DATE: / /

Extreme Programming & Agile Processes:

Agile Processes are based on common principles, such as

1) working s/w is a key measure of progress in a project.
2) The s/w should be developed & delivered rapidly in small increments.
3) Late changes in the requirements should be entertained.
4) Face-to-face communication is preferred over documentation.
5) continuous feedback & involvement of customer is necessary for developing good-quality s/w.
6) Simple design which evolves & improves with time is a better approach than doing an elaborate design
7) The delivery dates are decided by empowered teams individuals.

Extreme Programming (XP) is one of the best know Agile process.

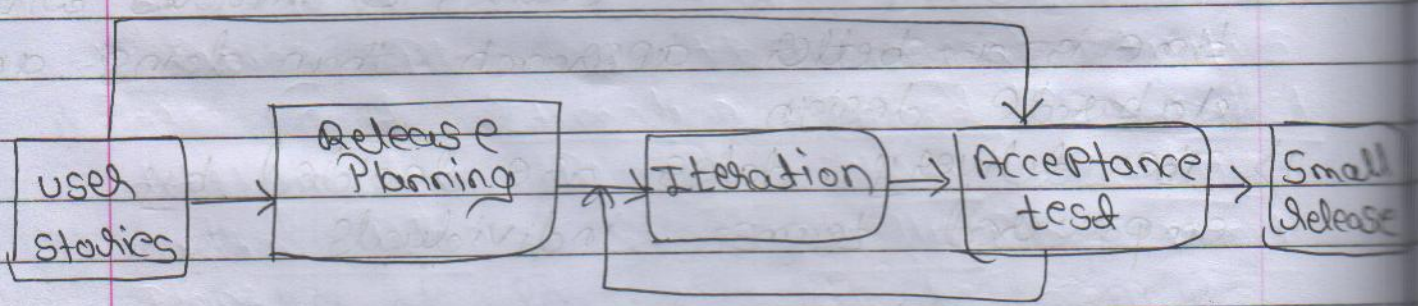
⇒ An extreme programming Project starts with user stories which are short descriptions of what scenarios the customers & users would like the system to support. Each story is written on a separate card.

⇒ The empowered development team estimates how long it will take to implement a user story. Using these estimates and stories

Release Planning is done that defines which stories are to be built in which system release. & dates of these releases.

⇒ Development is done in iterations, each iteration lasts not more than a few weeks. It starts with iteration Planning in which the stories to be implemented in this iteration are selected.

⇒ Acceptance Tests are built from the stories, which are used to test the s/w before the release. If Bugs (errors) are found during test, then repeat the implementation & retest the iteration. This overall Process is shown in fig-



Using Process Models in a Project:-

We want the Project to be done at low cost and less cycle-time and high-quality s/w. Another important task is selecting a suitable Process model

PAGE NO.:
DATE: / /

The following table shows the Situations in which particular Process model can be used.

Type of the Project	Suggested Model.
(1) ⇒ If a small Project is to be implemented. → If the requirements of the Project are well understood. ⇒ Existing manual system has to be automated.	waterfall model.
(2) ⇒ when the Project requirements are not clear. → The system will be new/beginner. ⇒ The GUI of the Project is very important.	Prototype model.
(3) ⇒ The Project is to be developed in very short span of time ⇒ The risk of long running Project is not affordable. ⇒ Requirements are not known & will be known only with time	Iterative model.
(4) Delivery of the Project is expected within a short period of time.	Time boxing model.

PAGE NO.:
DATE: / /

Unit 2

Project Management Process:-

⇒ Project Management is a process in order to meet the cost & quality objectives.

⇒ various activities that are carried out under this Project Management Process are

1. Project Planning
2. " Monitoring & controlling
3. Termination analysis.

Project Planning:-

⇒ The goal of this Phase is to develop a Plan for s/w development.

⇒ During Planning the major activities are cost estimation, Schedule & milestone determination, Project staffing, Quality control Plans & controlling and monitoring Plans.

⇒ This activity forms a basis for monitoring & controlling.

Project Monitoring and control:-

⇒ This activity is the longest in term of duration

⇒ Most of the development process is occupied by Project Monitoring and control

⇒ Most of the activities in this Phase revolve around managing the factors like

cost, Schedule & Quality.

⇒ The another important activity in this Phase is to monitor the risk.

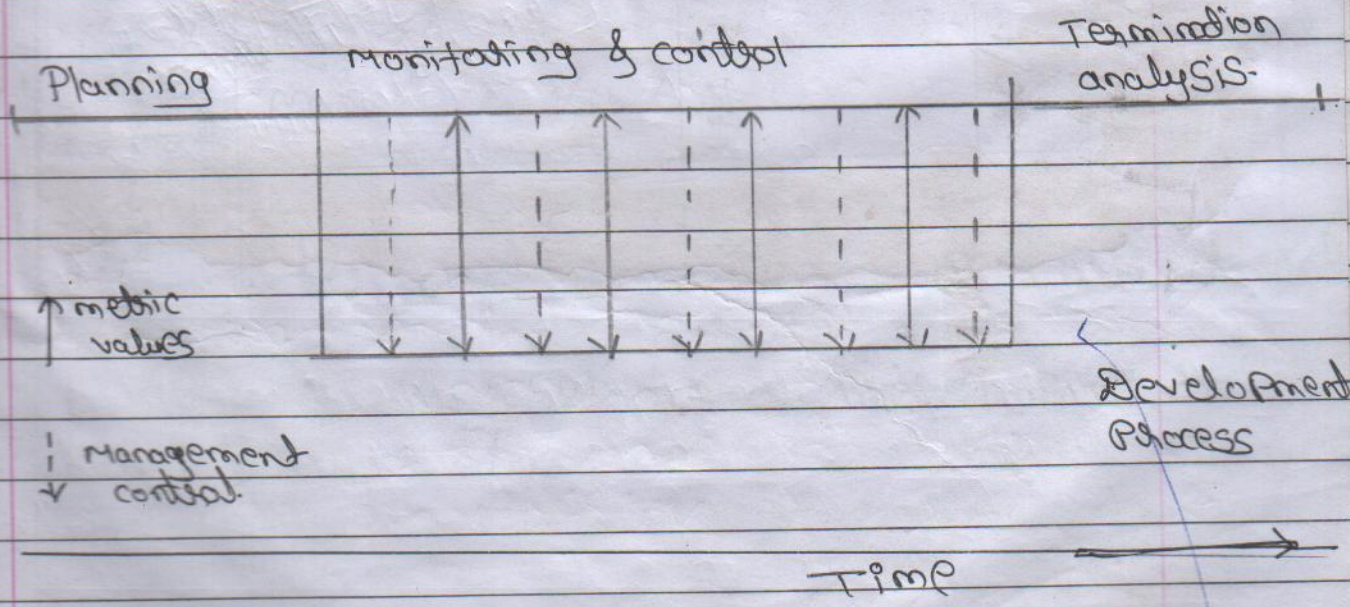
Termination Analysis :-

⇒ This is the last Phase of management Process.

⇒ when development gets over this Phase of management Process begins.

⇒ The basic reason for performing termination analysis is to Provide information about the development Process and learn from the ~~Process~~ Project in order to improve the Process.

The temporal relationship between management Process and development Process is as shown in the fig. This relationship shows that Planning is done before development begins. and termination analysis is done after the development is over.



:- Assignment Questions :-

PAGE NO.:
DATE : / /

1. Define software Engineering. Differentiate between process and project
2. Discuss the main differences between student software and industrial software.
3. Explain cost, schedule and quality.
4. Define scale and change in a software problem.
5. Define different attributes of software quality.
6. Explain software process and software project.
7. Classify the software process components.
8. Explain software development process model. List the various software development process models.
9. Explain project management process.
10. Explain Extreme programming.
11. Explain Agile processes.
12. Explain the temporal relationship between development and management process.
13. Discuss the software process and software project.
14. Explain components of software processes.
15. Explain Waterfall model with a neat diagram.
16. Explain Time boxing model with a neat diagram.
17. Explain Prototyping model.
18. Explain iterative development model.
19. Explain Rational Unified Process(RUP) model.
20. Explain Extreme programming and Agile processes.
21. Explain the phases of project management process.

UP IT - II 25 marks.

S/w Requirements Analysis and Specifications:

PAGE NO.:
DATE: / /

S/w Requirement Specification (SRS) :-

The goal of requirement activity is to Produce the SRS that describes what the Proposed S/w should do without describing how the S/w will do it.

Value of Good SRS:

client :- who needs the S/w system.
developers :- who builds the S/w system.
end user :- The complete system is used by end users.

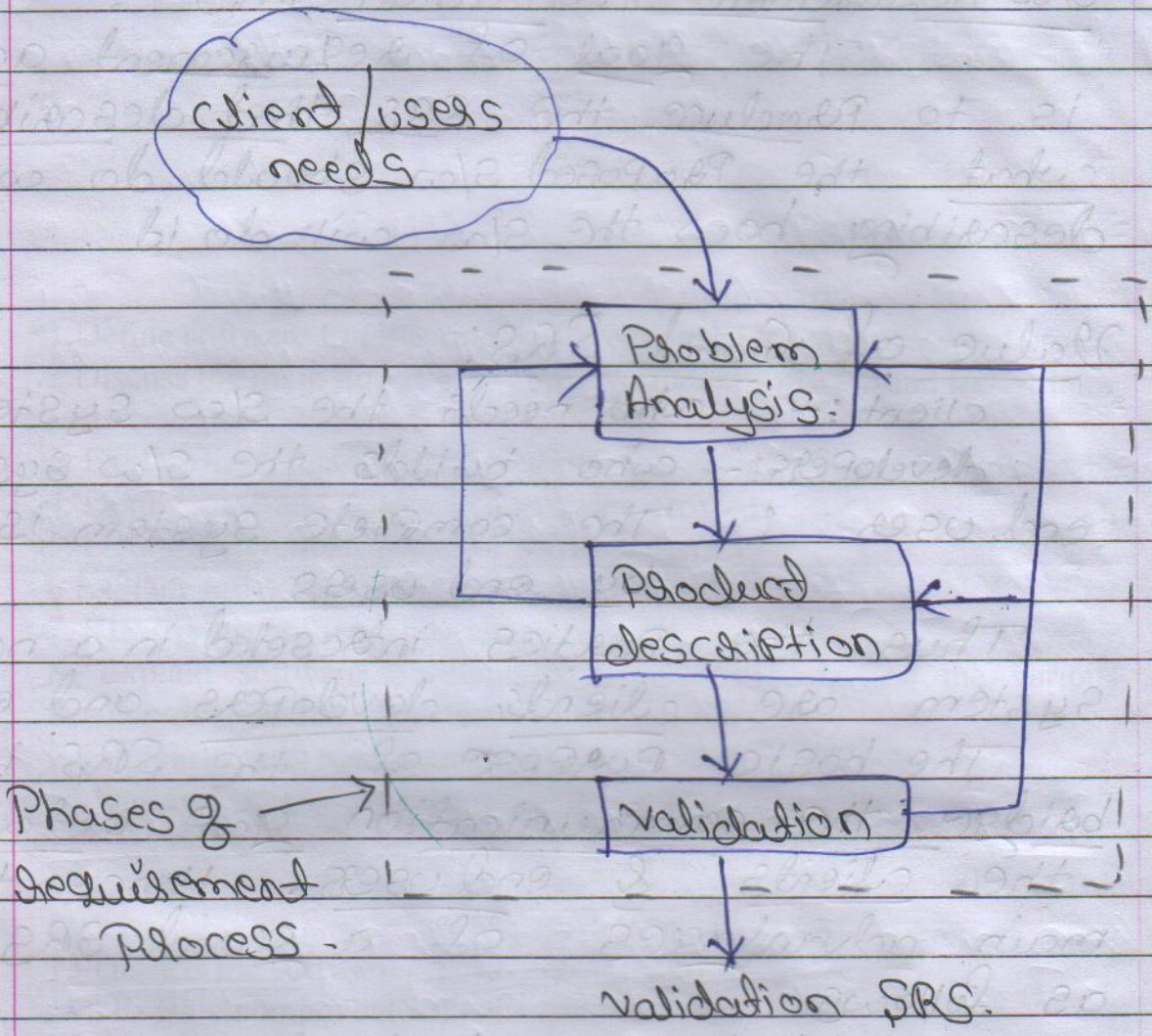
Thus the parties interested in a new system are clients, developers and end users

The basic Purpose of the SRS is to bridge the communication gap between the clients & end users. Hence the main advantages of a good SRS are as follows.

1) It establishes the basis for agreement between the client & the supplier on what the S/w Product will do.
2. It Provides a reference for validation of the final Product.
3. High Quality SRS reduces the development cost.
4. A High-Quality SRS is a Prerequisite to high-Quality S/w.

PAGE NO.:
DATE: / /

Requirement Process:-



The requirement process is the sequence of activities that need to be performed in the requirements phase. This process produces a high quality document containing the SRS. There are 3 main tasks that are:

- → Requirement analysis.
- → Requirement specification.
- → Requirement validation.

PAGE NO.:
DATE: / /

Requirement Specification :-

The final output of Requirement Process is the SRS (S/w Requirement specification).

The o/p of the Problem Analysis is submitted as I/P to the requirement specification / Product description.

SRS also specifies performance constraints design constraints, standards compliance and recovery which are not included in the models that are created in requirement analysis Phase.

Desirable characteristics of an SRS :-

To properly satisfy the basic goal, an SRS should have certain characteristics which are listed below:

1) correct
2) complete.
3) unambiguous
4) verifiable ✓
5) consistent.
6) Stability.
7) modifiable.

PAGE NO.:
DATE: / /

Correct :- The SRS must be correct, which means all the requirements are correctly mentioned, for eg: while developing a word processing s/w, if there is a requirement for spell check facility & if the s/w can't find the spelling errors from the document, then that means requirement is incorrect.

Complete :- The SRS is said to be complete only in the following situations

1) when SRS consists of all the requirement related to functionality, performance, attributes & design constraints.
2) when the label & corresponding references are mentioned for all the figures, diagrams & tables in the SRS.

Unambiguous :- SRS is said to be unambiguous if and only if every requirement stated has one and only one interpretation.

Consistent :- If there are no conflicts in the specified requirements then the SRS is said to be consistent. Three types of conflicts that may occur in a SRS:
1) Logical / temporal conflict
2) characteristics conflict
3) Two different description about the same real world object.

5 Stability :- The SRS must contain all the essential requirement. Each requirement must be clear.

Verifiable :- The SRS must be written in such a manner that the requirements that are specified within it must be satisfied by the S/w.

Modifiable :- Writing SRS is an iterative process. Even after specifying the requirements they can be modified later on if there is a change in user requirements.

Components of an SRS

The following are the some basic issues that SRS must address.

1) Functionality
2) Performance
3) Design constraints
4) External Interfaces.

Functionality :-

* The function requirements specify what should be o/p for the given I/p. This requirement describes the relationship between the I/p & o/p.

PAGE NO.:
DATE : / /

* The detailed description of all the data I/P & their source, the units of measure & the range of valid I/P must be specified.
* Similarly all the operations that can be performed on the I/P must be specified.
* The behaviour the system under abnormal conditions must be specified.

Performance :-
* There are two types of Performance constraints requirements ① static and ② dynamic.
* The static requirements are those requirements that do not specify the constraints on executing characteristics of the system.
eg :- The total number of users that can operate the system.
The capacity constraint is specified by static requirement.

* The dynamic requirements specify the constraints on the execution behaviour of the system. For ex:- responstime, Through

Design constraints :-
The various categories of design constraints are
1) Hardware limitations :- S/w System that is to

be developed has to work on the existing H/w.

2) Reliability & fault Tolerance:- Requirement about system behaviour on occurrence of faults should be specified. under recovery requirements on occurrence of failure what the system should do. this should be mentioned.

3) Security:- It place restrictions on the use of certain commands.

External Interfaces:-

This I/F requirement should specify the I/F with other S/w system.

Structure of Requirements Documents:-

S/w designers use IEEE Standards as the basis for the entire S/w specification. The general structure of SRS is as shown in fig-2.

The introduction section contains the purpose, scope, overview, etc. of the requirements document.

The next section gives an overall perspective of the system how it fits into the larger system.

Product perspective shows the relations of the product to other products.

1. Introduction
    1.1 Purpose
    1.2 Scope
    1.3 Definitions, Acronyms, and Abbreviations
    1.4 References
    1.5 Overview
2. Overall Description
    2.1 Product Perspective
    2.2 Product Functions
    2.3 User Characteristics
    2.4 General Constraints
    2.5 Assumptions and Dependencies
3. Specific Requirements

Figure 2 General structure of an SRS.

3. Detailed Requirements
    3.1 External Interface Requirements
        3.1.1 User Interfaces
        3.1.2 Hardware Interfaces
        3.1.3 Software Interfaces
        3.1.4 Communication Interfaces
    3.2. Functional Requirements
        3.2.1 Mode 1
            3.2.1.1 Functional Requirement 1.1
            :
            3.2.1.*n* Functional Requirement 1.*n*
        :
        3.2.m Mode *m*
            3.2.*m*.1 Functional Requirement *m*.1
            :
            3.2.*mn* Functional Requirement *mn*
    3.3 Performance Requirements
    3.4 Design Constraints
    3.5 Attributes
    3.6 Other Requirements

Figure 3 One organization for specific requirements.

A general abstract description of different functions to be performed by the Product is given.

One method to organize the specific requirements is to first specify the external I/F, followed by functional requirements, Performance requirements, design constraints. This structure is shown in the fig-3.

The external I/F requirements section specifies all the I/F of the s/w: to people, other s/w, H/w, & other systems.

The user I/F clearly a very important components they specify each human I/F the system plans to have, including screen formats, contents of menus, & command structure.

In H/w I/F, it describes I/F's to H/w devices.

Communication I/F: Describes N/w I/F.

Performance Requirements :- Specifies speed and memory requirements.

Functional requirement section, ~~the~~ specifies the required I/P, desired o/p & Processing requirement have to specified.

Functional Specification with use cases:

Use cases specify the functionality of a system by specifying the behaviour of the system. They can also be used effectively for analysis in later stages.

Basics

* use case is a set of scenarios tied together by some goal.
* use cases are textual descriptions.
* various terminologies used in use cases are

Terminology	Description
Actor	Actor is an entity that interacts with the system.
Primary Actor	It is a main actor which initiates the use case. If the goal of primary actor gets satisfied then the overall objective of the use case gets satisfied.
Scenario	It is a set of actions that are performed to satisfy the goal.
Main Success Scenario	It is a scenario in which all the steps of the scenario get satisfied.
Extension Scenario	If some steps of the main scenario do not get successful then the this exceptional behaviour can be modelled by extension scenario.

- The **main advantage** of use case is to focus on external behaviour and the specified functionalities of the system.

Example of use case.

Example 3.7.1 : *Design a use case for withdrawal of money from ATM system.*

Solution :

Use Case : Withdrawal of money from ATM

Primary Actor : Customer

Precondition : 1) Customer must have card and a PIN code.

2) ATM must have sufficient cash available

Main Success Scenario :

1. User inserts card and then types the valid PIN.
2. User makes an enquiry for balance and the system displays the balance.
3. System presents the menu to the customer for performing various operations.
4. Customer selects the menu for withdrawal of money and enters the desired amount.
5. If sufficient balance is present then the desired amount is returned to the customer.
6. Finally user closes the session and collects the amount, card and the statement.

Exception Scenarios :

1(a) The user enters the wrong PIN.

1(b) The system rejects the card if it is invalid or PIN is entered incorrectly for more than 3 times, then the system should display the warning message.

4(a) The customer enters the amount which exceeds the withdrawal limit, then the system should display the warning message.

4(b) The customer enters the amount that exceeds the amount of cash available in the ATM machine.

6(a) If the customer does not remove the money with in 15 seconds then the machine should display the message "Please remove your cash" and should beep.

Extensions :-

use cases are the textual description ~~of use case~~ in which the description about the Primary actor, precondition, main success scenario and

Exception scenario is given. Beside these use case can also specify the scope.

These are three ways to specify the scope:

1) Business use case.
2) System use case.
3) Component use case.

Example:- Assign a use case for withdrawal of money from ATM System.

[Refer the class-work].

Developing use cases:-

* use cases are used for requirement collection and problem analysis.
* In use cases additional steps can be added as the problem analysis proceeds.
* Hence use cases can be presented using different levels of abstraction.

Levels of Abstraction:-

1) Actors and Goals
2) Main Success Scenario
3) Failure condition.
4) Failure Handling

PAGE NO.:
DATE: / /

Actor and Goals :- A list of actor-goal is prepared in which the actors for each corresponding goal is specified. These use cases are later on reviewed in order to ensure the completeness of functionality

Main Success Scenario :-
using the main success scenario the system behaviour for each use case is specified. The review of main success scenario ensures ~~weather~~ whether the goals of stake holders are satisfied or not.

Failure condition :- The behaviour of the system other than the steps mentioned in main success scenario represents the failure condition

Failure handling :- When the designer determines behaviour of the system on occurrence of failure condition new actor must be defined in order to handle the failure conditions.

Steps involved in developing use cases are as follows

Step 1: Identify the actors and their goals and get an agreement with the concerned stakeholders as to the goals.

Step 2: understand and specify the main success scenario for each use case, giving more details about the main functions of the system.

Step 3: when the main success scenario for a use case is agreed upon the main steps in executing are specified, then the failure conditions can be examined.

Step 4: Finally, specify what should be done for these failure conditions.

Other Approaches for Analysis:

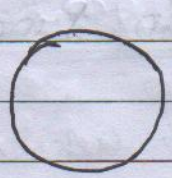
There are two commonly used approaches for analysis and design of the System.

1) DFD's [Data Flow Diagrams]
2) ER diagrams [Entity Relationship]

DFD:-

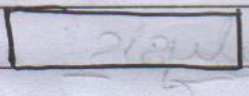
A DFD shows the flow of data through the system. It views the system as a function that transforms the I/P into desired o/p.

. The symbols that are used in DFD's are

 ⇒ Process

——— ⇒ Data Store / files / Database.

⟶ ⇒ Flow of data (may be I/P data or O/P data).

 ⇒ External entity [source or sink]

* ⇒ Multiple data flows by a Process. (AND relationship)

+ ⇒ OR relationship.

f^n : Data Dictionary :-

It can be defined as an organized collection of all the data elements of the system with precise and rigorous definitions. So that user and system analyst will have a common understanding of I/P, O/P, and intermediate calculations.

PAGE NO.:
DATE: / /

Example for DFD: that Pays workers is Show in fig 4 (next Page).

⇒ In this DFD there is one basic I/P data flow, the weekly timesheet, which originates from the source workers

⇒ The basic o/p is the Paycheck, the sink for which is also the workers.

⇒ First the employee's record is retrieved, using Employee ID.
⇒ From the employee record, the rate of Payment, & overtime are obtained.
⇒ These rates and the regular & overtime hours are used to compute the Pay.
⇒ After the total Pay is determined, taxes are deducted.
⇒ The amount of tax deducted is recorded in the employee & company records.
⇒ Finally, the Paycheck is issued for the next Pay.

⇒ The amount is Paid & also recored in company records.

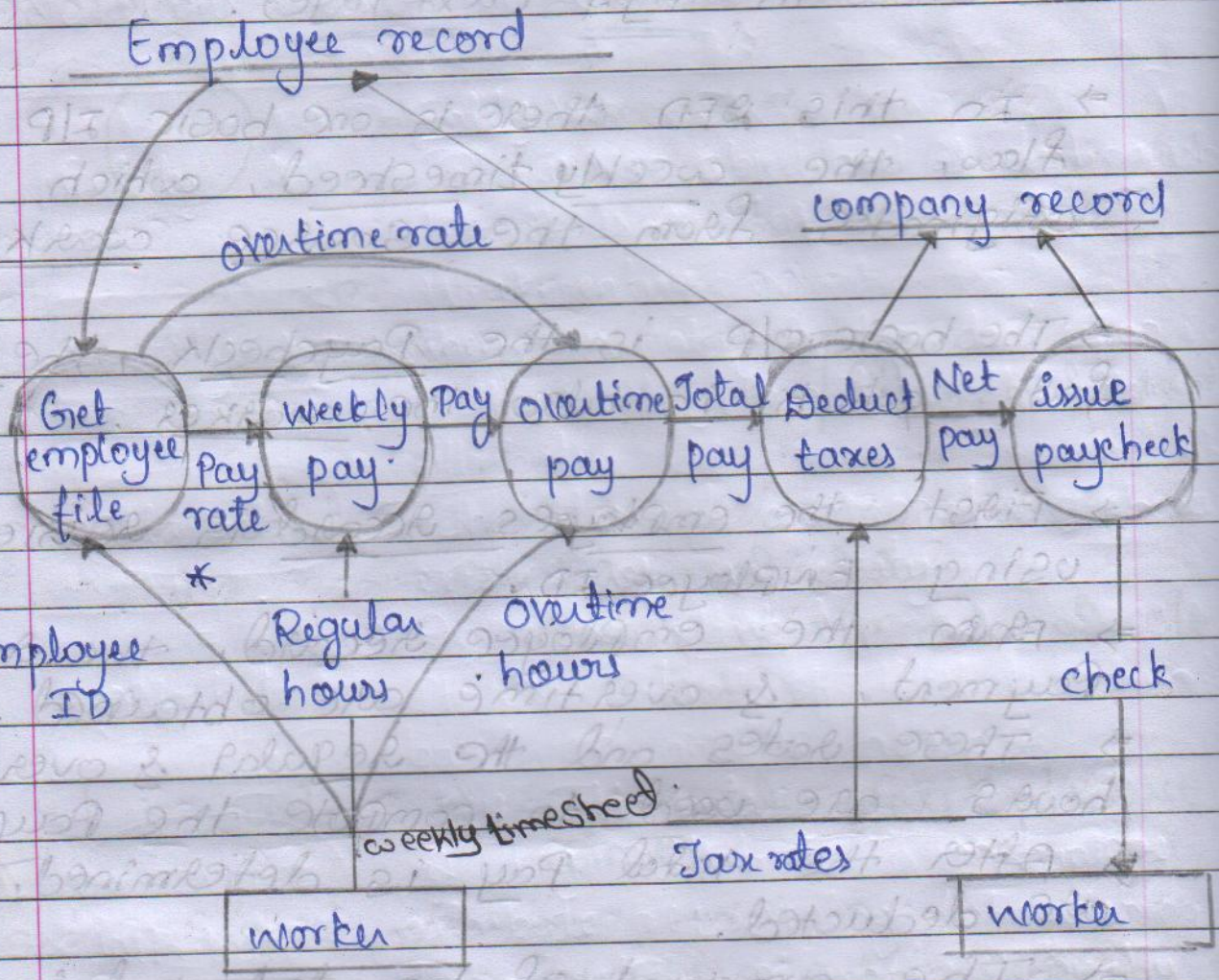


fig 4: DFD of a system that pays workers.

Entity Relationship diagrams:- (ERD)

⇒ An ERD can be used to model the data in the system and how the data items relate to each other.

⇒ It is often used by database designers to represent the structure of the database

PAGE NO.:
DATE: / /

=> It is usefull tool for analyzing s/w systems which employe database.

These various notations used in ER Diagram

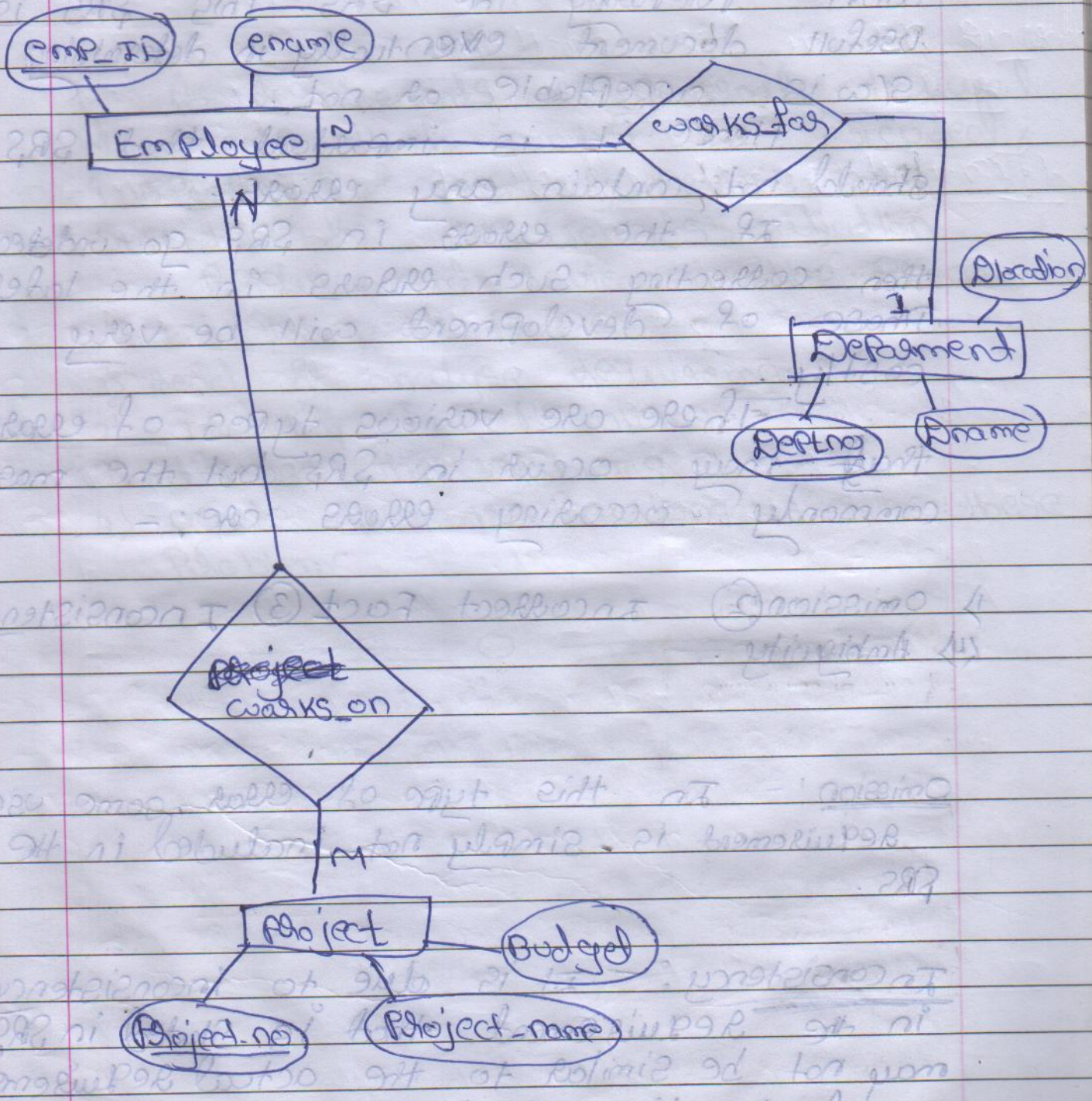
[P.T.O to refer the Notations]

PAGE NO.:
DATE: / /

* Notations used in ER diagram.

Entity	**Entity** It is an object and is distinguishable it is similar to record.
Entity	**Weak entity** When this entity is dependant upon some another entity then it is called week entity
Attribute	**Attribute** The attribute are properties or characteristics of an entity.
Attribute	**Derived attribute** It is a kind of attribute which is based on another attribute.
Attribute	**Key attribute** A key attribute is an unique attribute representing distinguishing characteristic of entity. Typically primary key of records is a key attribute.
Attribute	**Multivalued attribute.** A multivalued attribute have more than one value.
relation-ship	**Relationship** When two entities shares some information then it is denoted by relationship.

Let us draw an ER diagram for Employee, Project & Department.



PAGE NO.:
DATE: / /

Validation

The Process of S/w development begins with preparing the SRS. This SRS is useful document eventually in determining S/w is acceptable or not.

Hence it is important that SRS should not contain any errors.

If the errors in SRS go undetected then correcting such errors in the later Phase of development will be very costly.

There are various types of errors that may occur in SRS but the most commonly occuring errors are:-

1) Omission (2) Incorrect Fact (3) Inconsistency
(4) Ambiguity.

Omission:- In this type of error, some user requirement is simply not included in the SRS.

Inconsistency:- It is due to inconsistency in the requirement that is stated in SRS may not be similar to the actual requirement stated by the client.

Incorrect:- This type of error occurs when some fact recorded in SRS is not correct.

PAGE NO.:
DATE: / /

Ambiguity:- If there are multiple meaning to the stated requirement then this type of error occurs.

As requirements are generally textual documents that cannot be excuted, inspection & Review are suitable for requirements validation.

Requirement review is the process in which group of people -

1) read & analyse requirements
2) Identify the problems.
3) discuss the problems.
4) agree upon the actions to solve these problems.