UNIT - II

Requirements Analysis and specificat

1. S/w Requirements:

S/w req are classified as functional or non functional requirements

→ Functional req - These are stmts of services the sys should provide how the sys should react to particular i/p's and how the sys should behave in parti situations. The fn-al req may also explicitly state what the sys should not do.

→ non-fnal req:

The fns are offered by sys. They include timing constraints, constraints on developement process and constraints imposed by stds - non-fn-al req often apply as a whole, rather than individual sys features or services.

Functional requirements:

The fn-al req for a sys describe what the sys should do. These req depends on the type of s/w being developed, the expected users of the s/w and general

→ when expressed as user req fn.al req are usually described in abstract way. that can be understood by sys users. more specific fn.al req describes the sys fn its input and o/p, exceptions etc in detail.

Eg: The fn.al req for metal health care patient mgmt sys (MHC-PMS) used to maintain info about patients receiving treatment for mental health problems are

① A user shall be able to search the appointment lists for all clinics

② sys should generate each day For each clinic a list of patients who are expected to attend appointments that day.

③ Then fn.al user req define specific facilities to be provided by the sys.

fn.al req should be both complete and consistent.

Non-functional Requirements:

→ The requirements that are not directly concerned with the specific services delivered by the sys to its user.

→ It relates to emergent sys properties such as reliability response time and store occupancy

→ Also they may define constraints on the sys imple such as capabilities of I/O devices or the data representations used in interfaces with other systems.

→ Non-fn al requirements such as performance, security, or availability, usually specify or constrain characteristics of the system as a whole.

→ failing to meet non-fnal req means that the whole sys is unusable.

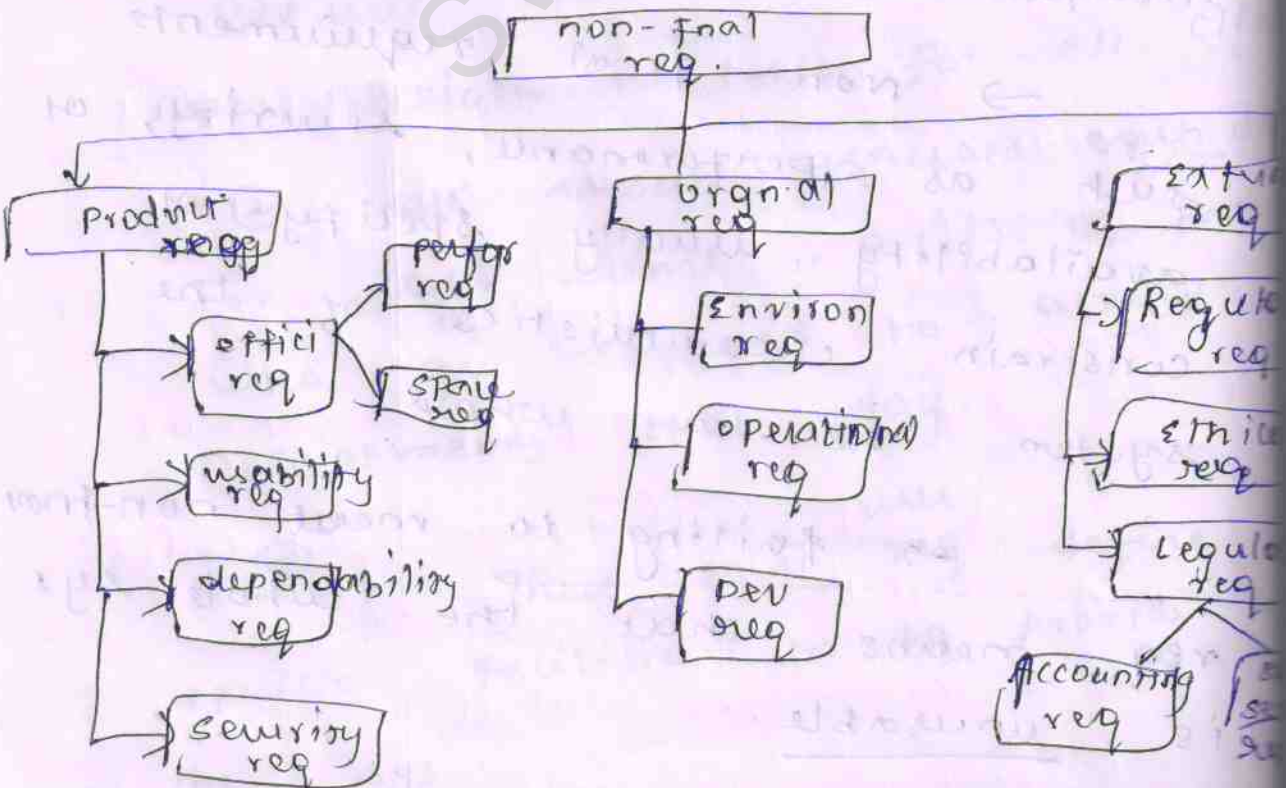
For eg: The aircraft sys does not meet its reliability req it will not be certified as safe

The imple of these req may be diffused thr' the sys. There are two reasons for thi

(i) non-fnal req may affect overall archi of a sys rather than the indivi component.

(2) A single non-fn·al req, such as security req, may gen a no·of related fn-al req, that define new sys services that are required.

Classification of non-fnal req:



1. Product req

The req specify the behaviours of the sys.

→ It include performance req on how fast the sys must execute and how much memory it requires.

→ reliability req that set out the acceptable failure rates, security req & usability req.

2. Organisational Requirements.

These req are broad sys req derived from policies and procedures of customer's and developer's orgn.

→ operation process req define how the sys will be used.

→ dev. proces req that specify the programming lang.

→ envi req that specify the operation environ. of the sys.

External Requirements:

This broad heading covers an req that are derived from factors external to the sys and its dev procss.

→ regulatory req that set out what must be done for the sys to be approved for use by a reg

→ legislative req that must follow the law & Emical req.

The non fnal sys properties are in below table.

property	measures.
Speed	processed trans/sec, user/event, run
size	M bytes, no of Rom chips.
Ease of use	Training time, no of help fra
Reliability	mean time to failure prob of unavailability, Rate of failure occurrance availabi
Robustness	Time to restart aft failure % of events causing failure prob of data corruption
Portability	% of target dependent stmt.

S/w Requirements Document: (SRS)

→ The s/w req doc (SRS) is an official stmt of what the sys developers should implement.

→ It should include both user & system req. some times, they are integrated into a single description.

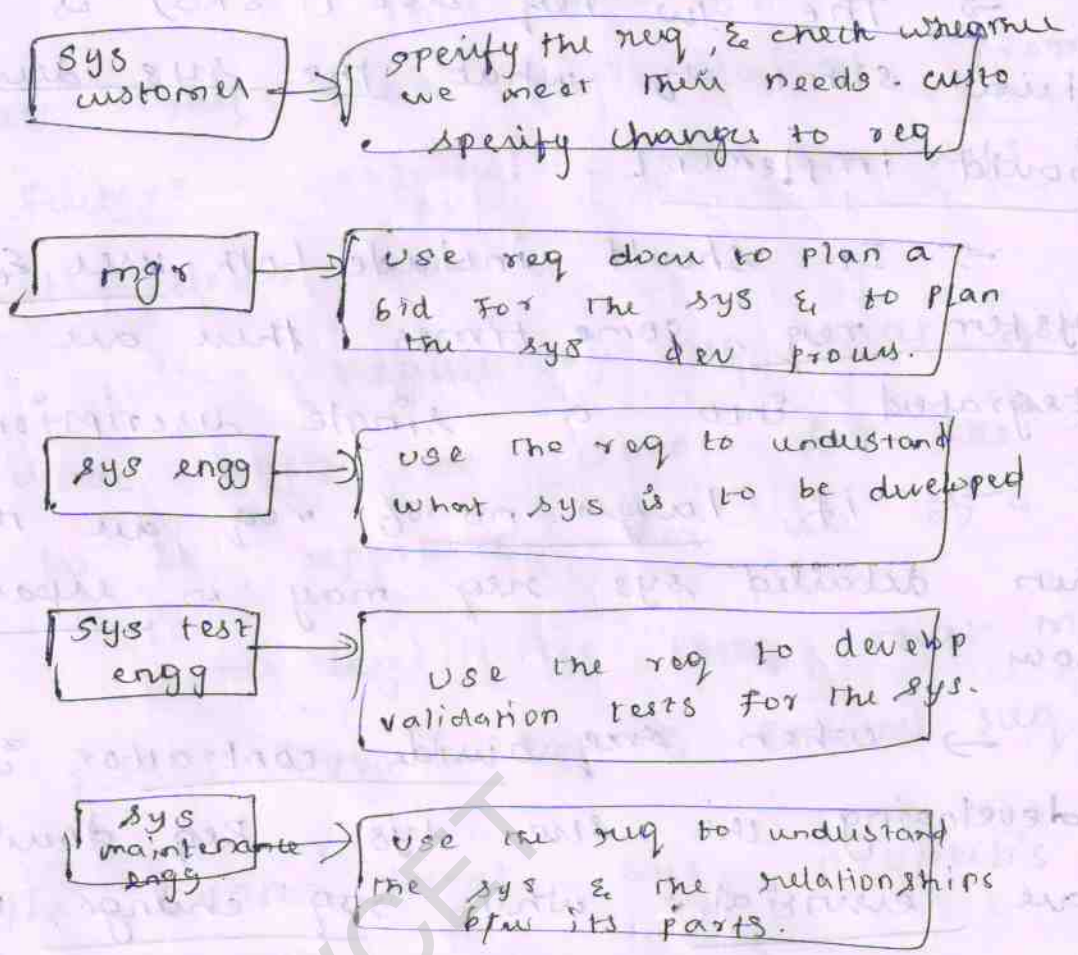
→ if larger no of req are there, then detailed sys req may in separate doc

→ when req outside contractor is developing the s/w sys, Req doc's are essential. when req change, the doc is out of data, & so the effort is largely wasted.

→ Extreme prog approaches collect the user req incrementally & write them on cards as user stories. Then user priorities req for imple in the next increment of the sys. This is suitable for business sys where req are not stable.

→ req doc have diverse set of users, ranging from senior mgmt of orgn, to the engineers responsible for

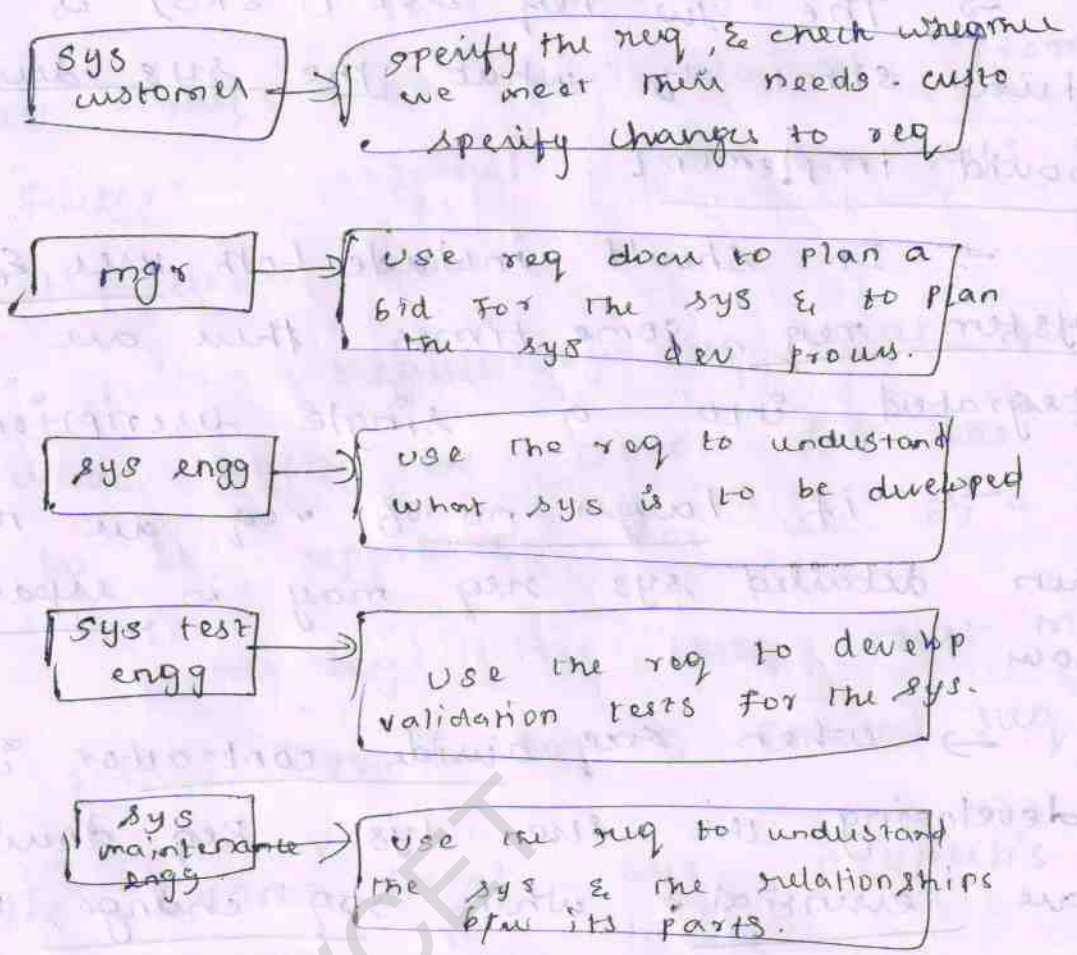
Fig users of req docum



IEEE std for req document.

Chapter	description
Prefau	expected readership of docu & describe its versions history & chang
Intro	need for the sys. briefly describes the sys's fn. & explain how its works.
Glossary	technical terms used in the docu.
user req defn.	define service provider for the user. it describe non-fn al req.
sys. archi	high level overview of the sys

Fig areas of req docum



IEEE std for req document.

Chapter	description
Prefau	expected readership of docu & describe its versions history & chang
Intro	need for the sys. briefly describes the sys's fn. & explain how its works.
Glossary	technical terms used in the docu.
area req defn.	define servic provider for the user. it describe non-fn al req.
sys. archi	high level overview of the sys

system req — it describe fn al & non-fn al
Spcifi — req in more detail. ;;

System models — include graphical sys models showing the relationship b/w sys comp. & the sys and its environ.

System evolution — fundamental assumption on which the sys is based and any anticipated changes due to b/w evolution, changing user needs & so on

Appendices — detailed, specific info, that is related to appln developed

Index — index of diagram, table, fn etc.

Requirement Engineering Process:

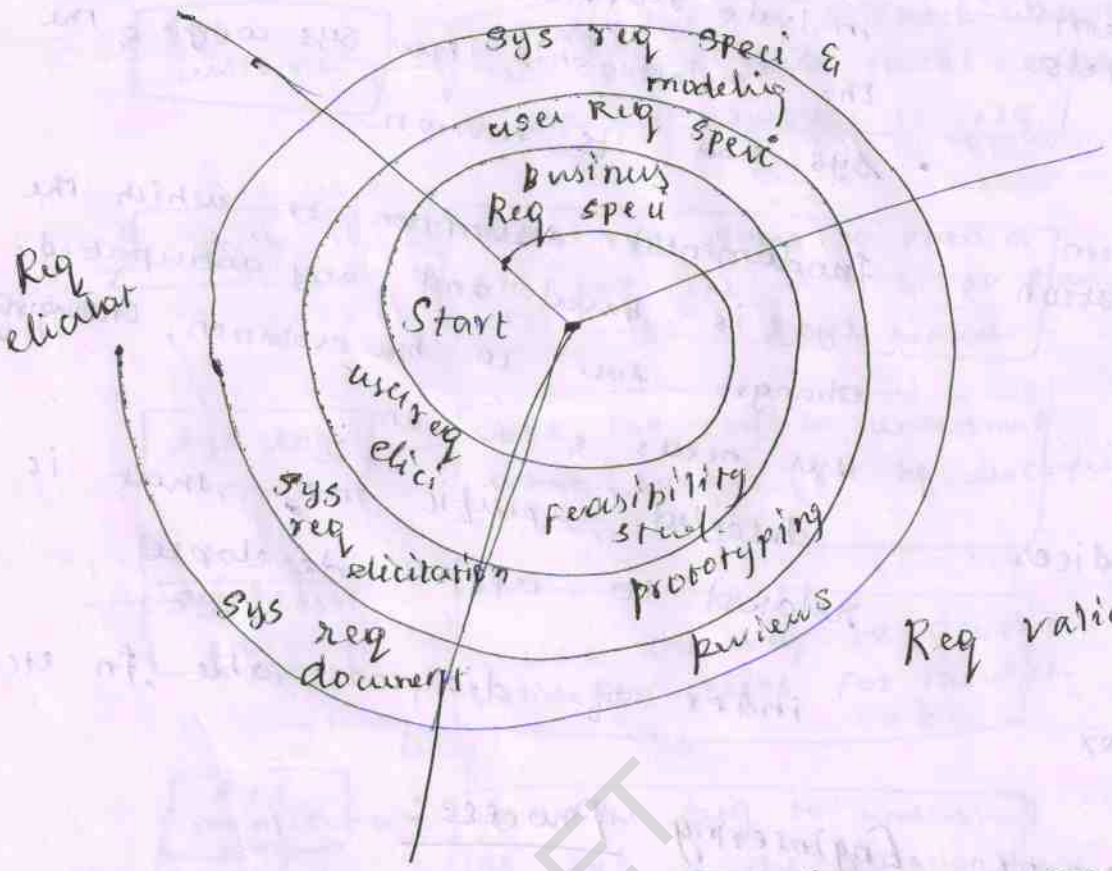
Req engg process may include 4 high-level activities

(i) business (feasibility study)
(ii) discovering req (elicitation & analysis)
(iii) converting these req into some std (specifications)
(iv) checking that the req actually define the sys that the customer wants. (validation)

The req engg is an iterative process in which the activities are interleaved.

The activities are organized as an iterative process around a spiral, with

Req Spec



→ Early in the process, most of the time spent is high-level busin and non-fnal req, user req for sys Later in process, most effort will be devoted to eliciting and understandi the detailed sys req.

Feasibility studies:

A feasibility study is a sho focused study that should take plac early in the req eng process. It should ans 3 questions.

(i) does the sys contribute to the overall obj of the orgn.

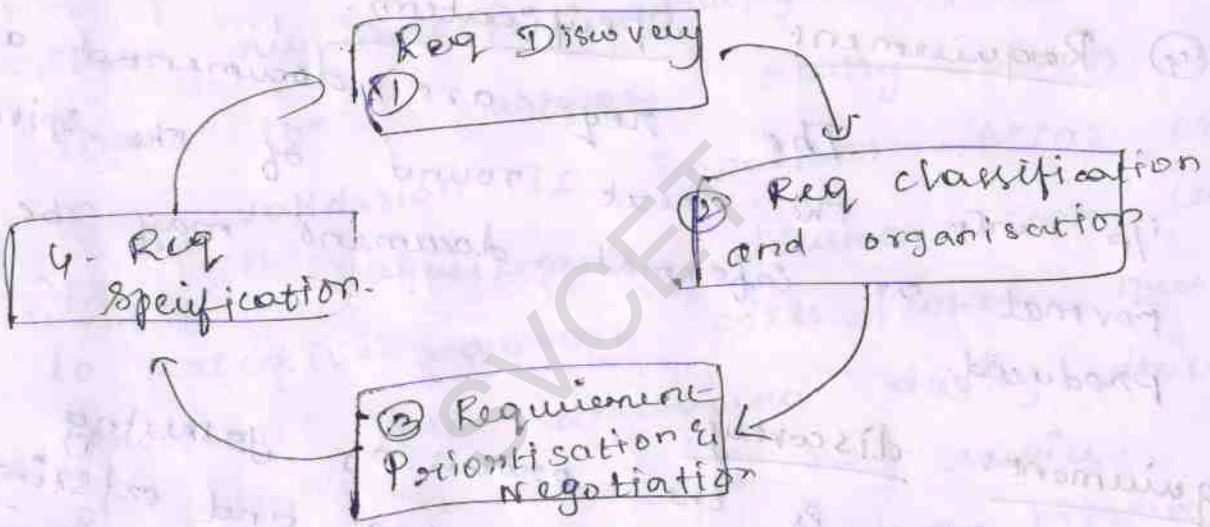
(ii) can sys be implemented using current tec within schedule & budget

(iii) can the sys be integrated with other sys that are used.

(5)

Requirement Elicitation and Analysis: (REA)

→ REA may involve a variety of different kinds of people in an orgn. A system stake holders is anyone who must have direct or indirect influ on the sys req. Stake holder includes end user & engineers, developers.



REA process activities are.

① Requirement Discovery:

It's a process of interacting with stake holders of sys to discover their requirements. Domain req from stakeholders and documentation are also discovered during this activity.

② Requirement classification and organisation

It takes unstructured collection of req groups related req, & organizes them into clusters. Common of

grouping requirements as use a mod
of sys architecture to identify sub
systems and to associate req with
each sub-systems.

3. Requirements Prioritization and Negoti

when multiple stake holders are
involved, req will conflict. This activit
is concerned with prioritizing req an
finding and resolving req conflicts th
negotiation.

④ Requirement Specification:

The req are documented
i/p into the next round of the sy
formal or informal document may b
produced.

Requirements discovery:

It is the process of gathering
info about the required sys and exist
sys and distilling the user and sys.
from this info.

interact with stakeholders thr' inte
and observation and use scenario & prot
to help stakeholders understand what
sys will be like.

Interviewing:

interview may be of two types.
1. closed interviews

Scenarios:

People usually find it easier to relate to real-life examples than abstract descriptions.

Use cases:

Use cases are a req discovery technique that we'll first intro in the Objectory method.

Requirement Validation:

It is the process of checking that requirements actually define the sys that the customer really wants.

Req validation is important because errors in a requirements document can lead to extensive rework costs when these problems are discovered during development or after the sys is in service.

The cost of fixing a req problem by making a sys change is usually much greater than repairing design or coding errors.

Different types of checks

(i) validity checks:

A user may think that a sys is needed to perform certain fns. The thought and analysis may identify additional or diff fns that are required

(ii) Consistency checks:

Req in the document should no conflict. That is there should not contradictory constraints of the same sy

(ii) Completeness checks.

The req doc should include req that define all fns and the constraints intended by the sys use

(iv) Realism checks:

Using knowledge of existing tech. the req should be checked to en that they can actually be implement

(v) Verifiability:

To reduce the potential for c b/w customer, sys req should alwa be written so that they can verifia This means that we should be abl write a set of tests that can dem that the delievered sys meets eac specified requirement

Requirement Management:

The req for large sw are always changing. one reason This is that these sys are usually developed to addr the prob that cannot be completed defined. As a the s/w bound to be in

Once a sys has been installed and is regularly used, new req inevitably emerge. There are several reasons why change is inevitable.

(1) The business and technical environments of the system always changes after install new h/w may be introduced. It may be necessary to i/f the system with other sys,

(2) people who pay for a sys and users of that sys are rarely the same people.

(3) Large sys usually have a diverse community with many users have different req & priorities that may be conflicting

Req mgmt is the process of understanding and controlling changes to sys req.

Requirements mgmt planning:

Planning is a essential stage in the Req mgmt process.

① Requirement Identification:

Each req must be uniquely identified so that it can be cross-referenced with other req and used in traceability assessments.

(ii) A change mgmt process:

This is the set of activities that assess the impact and cost of changes.

(iii) Traceability policies:

These policies define the relationships b/w each req & b/w the req & the sys design that should be recorded. The traceability policy should also define how the records should be maintained.

(iv) tool support req mgmt involves the processing of large amt of info about the req.

Req mgmt needs automated support tool. we need tool support for

① Requirement storage:

The req should be maintained in a secure, managed data store that is accessible to every one involved in the req engg process.

② change mgmt:

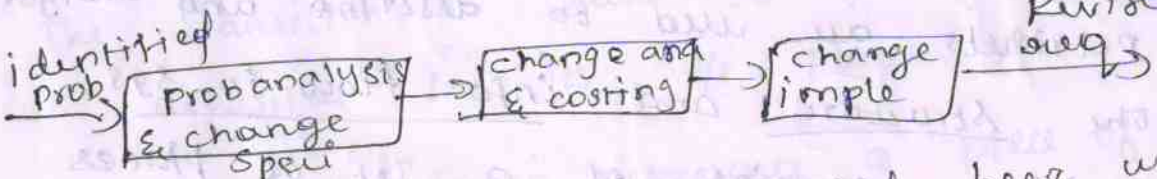
The process of Change mgmt is simplified if active tool support is available.

③ Traceability mgmt:

For tool support for trace allows related req be discovered

Requirement change mgmt

Req change mgmt should be applied to all proposed changes to a sys req, after the req doc has been approved.



change mgmt is essential bcz we need to decide if the benefits of imple new req are justified by the costs of imple.

3 principal stages to a change mgmt process:

(1) problem analysis and change specification

The problem or change proposal is analysed to check that it is valid.

(2) change analysis and costing:

The cost of making the change is estimated both in terms of modification to the req doc.

(3) change Implementation:

when ever a new req has to be urgently implemented we should not change the sys and modify the req doc as it leads to req spec and sys imple getting out of step

PETRINETS

A petrinets is a form of finite state machine useful in modeling concurrency and asynchronous comm. petrinets are used to describe and analyse the *structure* and *info flow in sys.* petrinet consist of a set of places P, set of transitions T, an input function I, o/p fn O. place represent storage for i/p or for o/p. Transitions represent activities that transform i/p into o/p. An i/p mapping I maps a transition to its i/p places. An o/p mapping O maps a transition to its o/p places.

Notation for Petrinets:

- • - Token (information)
- ○ - place
- ⊙ - place marked with one token
- | or ▯ - transition
- → - connection b/w place and transitions

whenever the activity represented by transition t occcus, this activity is hidden from view. whenever the transition completes the transformation of its i/p an event occurs. The transition is said to fire and its o/p is deposited in its o/p places.

→ A token represents a piece of info either to be processed by one or more transitions or info resulting from the firing of one or more transitions. The placement of tokens in a petrinet is called a marking

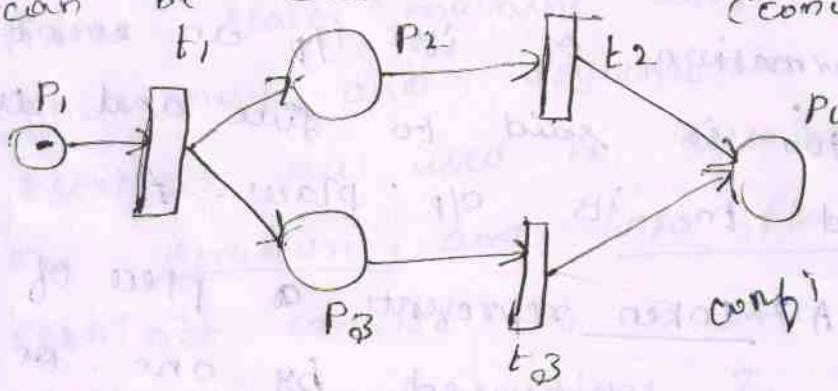
Petrinets are governed by transition firing rules:

(i) A transition is enabled if each of it's i/p places has the required no. of tokens, one for each arc leading from a place to a transition.

(ii) A trans can only fire if it is enabled.

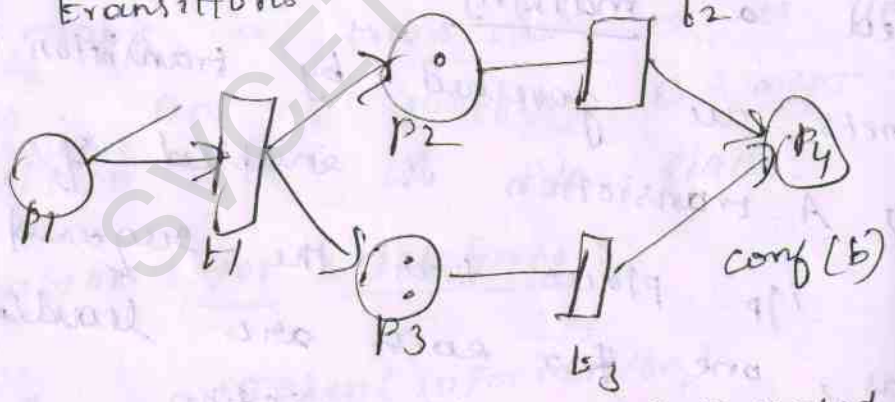
(iii) whenever a tran t fires, each of token that enabled t is removed and transition t places one or more tokens in each of its o/p places, one for each arc leading from t to an o/p place.

The significance of the fire firing rule is that more than one transition can be enabled at the same time (concurrent proces)



config (a)

In second config (after transition t_1 fires) transitions t_2 and t_3 are enabled (The activities represented by these transitions overlap in time)



config (b)

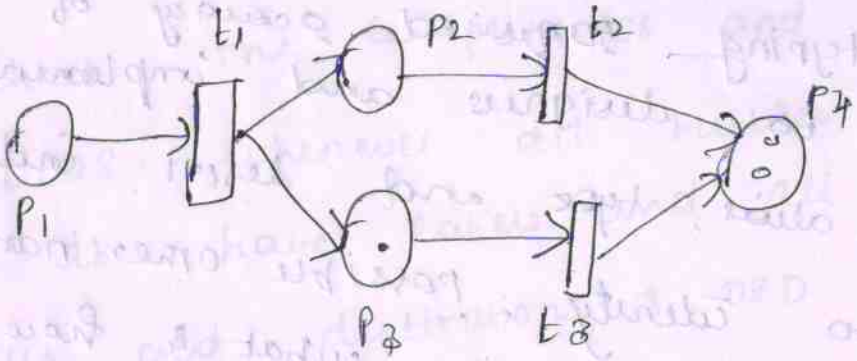
The behaviours represented by changing the petrinet result from successive firings of transition which can be explained by writing out some of the applns of the i/p I and o/p O mapping.

config (a) : $I(t_1) = \{P_1\}$ with t_1 enabled

config (b) : $O(t_1) = \{P_2, P_3\}$, $I(t_2) = \{P_2\}$

$I(t_3) = \{P_3\}$ t_2 & t_3 enabled

Confi (c) : O (t2) = O (t3) = { P4 }. (6)



conf (c)

Data Dictionaries:

A data dic Stores info about data items found in a DFD.

A Summary of typical info used to construct a data dic is given below

Info stored	Explanation
Name	identifies data item
Alias	Identifies other names, abbr, used to identify a data item
DS (type)	Type of data (eg; int, char)
Description	indicates how (why) a data item is used
Duration (begin)	Life span of data (when created)
Accuracy	High, medium, low accuracy
Range of values	Allowable values of data item

A data dic supplies info ~~such~~ about data typing required accuracy of data useful to designers and implementers name, alias, type and descri indicating how to identify. possible other names type of data and what & how the data are used.

duration, accuracy and range values specify ~~the~~ life span, all permissible data values of data items respectively. Data flow specify processes, that generate or receive data.

when data are derived from a real time sys, data items can have timing constraints specifying the period of time before the data become out-of-date. for eg. traffic flow changes continuously, depending on time so traffic flow data indicating current and near time flows needs to be refreshed, replacing old with new data.

A data dictionary can be used to check the completeness and consistency of DFDS. whenevel all bubbles, arrows and db have labels and all arrows have sources and destinations, a DFD is considered compute.
