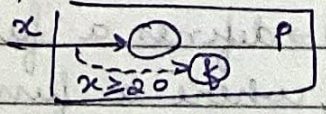
Week-8

4 Requirements

- SDLC Life Cycle
- product req. → describe the properties of the product or a sys. Product can be a software too.
- process req. → describe the activities to be done by the org. involved in the product dev.
- diff. types of req → "X" RS (X → Business, User, etc.)
- Business req. → specific characteristics from pov of user. StRS (Stakeholder).
- User req. → specifies what the user expects the software to be able to do. URS. A mutually agreed contract details what the software must do from the pov of a user.
- Functional req. (FRS) → defines a func^n of a sys., or its component, where a func^n is described as a specifica^n of behaviour between inputs & outputs. It describes specific / particular results / behaviour of the sys. Can be given as use cases. Design document caters to functional req. "What"
- Non-func. req. → specifies criteria that can be used to judge the opera^n of a sys., rather than specific behaviour. "How". Specified as diff. quality parameters.
- Regulatory req. → Regula^n is the mgmt. of systems / software as per a set of rules & regula^{ns} specific to several diff. factors including business, etc.
- Black box testing deals with req. & spans most of the req. Treats code as a black box. Test cases are designed purely based on req. to be tested, inputs & outputs.

L2 Functional Testing

- A prog. P is viewed as a func^n transforming inputs to outputs. Given inputs x_i, P computes outputs y_i s.t. $y_i = P(x_i)$
- Impt steps →
  1. precisely identify the domain of each I & O.
  2. select values from the data domain of each var. having impt. properties.
  3. consider combina^n of spl. values from diff. input domains to design test cases.
  4. consider I. values s.t. the prog. under test prod. spl. values from the domains of the O. var.
- Sometimes, we need to know minimal content.



- Equivalence Class Partitioning → If input domain is too large, it is partitioned into finite no. of sub-domains (equivalence class) for selecting test inputs. 1 input from each sub-domain.
- BVA (Boundary Value Analysis) → it selects test inputs near the boundary of a data domain so that the data both within & outside eqv class are selected. Once eqv class partitioning partitions the inputs, boundary values are chosen on & around the boundaries of the partitions to generate test input for BVA.
- Decision Tables → It handles multiple inputs by considering diff. combina^{ns} of equivalence class. Conditions & effects in table format.
- Random Testing → Test inputs are selected randomly from the input domain.

↳ input space partitioning

L3 ISP-1

- Given a set S, a partition of S is a set $\{S_1 \cdots S_n\}$ of subsets s.t., they are pair-wise disjoint & the union of all sets is set S.
- The set that is split into partitions while testing is the input domain. Each partition represents 1 characteristic of the input domain. Also, there is an underlying equivalence relaⁿ btwn. that influences the partitions.
- Partitions must ensure - completeness and disjoint
- Steps in input domain modelling —
  (i) identificaⁿ of testable funcᵐˢ.
  (ii) identify the params that affect testable funcⁿ
  (iii) identify characteristic & partitions.
  (iv) get the test inputs.
- ISP - 2 broad approaches —
  1. Interface-based → considers each param in isolaⁿ
     Adv. → easy to identify, easy to model input domain
     Disadv. → incomplete, analysing in isolaⁿ misses combinaᵐˢ
  2. Functionality based → identifies overall functionality of the sys. under test
     Adv. → better results, testing starts early
     Disadv. → can't identify charac., hence, testing difficult
- Choosing Partitions → key step of ISP, balance btwn. no. and their effectiveness.
- Identify Values Strategy → valid values, sub-partitions, boundaries (BVA), invalid values, balance, missing partitions, overlapping partitions.

L4 ISP-2

- ACoC (All Combinations Coverage) – Take all combinations of each partition, of each block

Eg. 3 partitions [A, B], [1, 2, 3], [x, y]

Total tests = $\prod_{i=1}^{n} B_i$ B_i → is no. of block in each partitions
n → no. of partitions

$2 \times 3 \times 2 = 12$ tests.

- Each Choice Cov. (ECC) – 1 value from each block for each charac.

$\max_{i=1}^{n} B_i$ $\max(2, 3, 2) = 3$

weak criterion

- Pair Wise Cov. (PWC) – do pair-wise

$\leq (\max_{i=1}^{n} B_i)^2$ $\max(2, 3, 2) = (3)^2 = \leq 9$

- T-Wise Cov. (TWC) – extension of PWC

$(\max_{i=1}^{n} B_i)^t$ → t characteristics

- Base Choice Cov. (BCC) – take a base choice from each block.

Eg. (A, ①) (1, 2, 3) (x, y)
A, 1, x

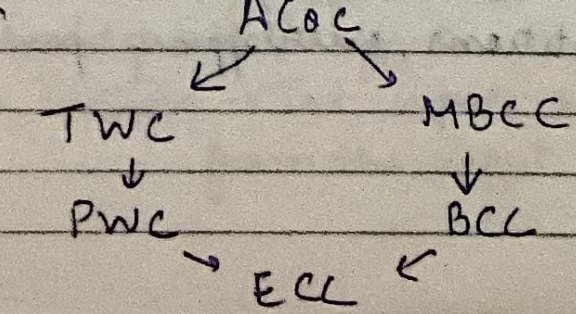
$1 + \sum_{i=1}^{n} (B_i - 1)$

$1 + (1 + 2 + 1) = 1 + 4 = 5$

- Multiple Base Choice Cov. (MBCC) – extension of BC

$M + \sum_{i=1}^{n} (M \times (B_i - m_i))$

- Subsumption



LS ISP Ex.

- Constraints are rela^{w} btw. partitions from diff. charac. 2 kinds —
  1. A block from 1 characteristic can't be combined with a block from another charac.
  2. must be combined.
- Constraints are handled better with BCC & MBCC criteria. The base cases can be altered to handle infeasible constraints.