Preparatory Guide

For

ISTQB Advanced Test Manager (CTAL – TM)

Certification Exam

By

Sandeep Virk
ISTQB Advanced Test Manager
(CTAL – TM)

This E-Book contains Important bulleted points for every chapter of ISTQB Advanced Test Manager Syllabus. These points shall remain extremely handy for your last minute review of your preparation for the certification exam

After using this E-book, please don't forget to thank the Author of this E-book at e-mail ID virk_sandy@hotmail.com

www.softwaretestinggenius.com

CH-1 Basic Aspects of Software Testing

1) Sequential (waterfall model, V-model and W-model)
   - Requirements are complete and furnished early
   - Preventive in nature
   - Defect detection starts late
   - Manage requirements

2) Incremental (evolutionary and Agile methods)
   - Requirements are not complete
   - Not preventive in nature
   - Defect detection starts early
   - Identify and prioritize key quality risk areas

ISSUES

Sequential (V-model, waterfall)
- Schedule compression pressure is always there
- Pressurized development
- Test team involved late

Incremental (Agile- SCRUM) → also called as evolutionary
- Regression of N+1 increment is always complex
- Bugs planning failure
- Lack of testing culture in agile world

Iterative (RAD; Spiral)
- Flexibility in TC and design changes of system is too often
- Experimental model of early testing and not confidence building
- Dealing with unknowns

1) Systems of Systems: set of collaborating components interconnected to achieve a common purpose.
- Less reliable than individual systems.
- Multiple levels of integration and version management
- Long duration of project
- Formal transfer of information among project members is always hard
- Requirements for regression tests at system of systems level
- Maintenance testing due to upgrade, etc

2) Safety Critical Systems: can result in catastrophic or critical consequences.
- Proof of adequate testing
- Various regulations and standards apply for audit control
- Traceability to regulatory requirements and means of compliance
- Rigorous approach to development and testing
- High degree/level of documentation (depth and breadth of documentation)

Metrics & Measurement → Metrics means numbers and measurement means trends and graphs

1. Definition
2. Tracking: Use of automated tools
3. Reporting

1.5 Ethics (short cut - PCP JM PCS)

PUBLIC; CLIENT AND EMPLOYER; PRODUCT; JUDGMENT; MANAGEMENT; PROFESSION -

COLLEAGUES; SELF.

CH-2 Testing Processes

Test Process Models → Approximation and abstractions

- Fundamental test process model
- TMM: Practical Software Testing – Test Maturity Model
- CTP: Critical Testing Processes
- Systematic Test and Evaluation Process (STEP)

Fundamental test process model

1) Test Planning (Test plan) and control

Test Planning

a. Implements test strategy

b. Sets framework for deriving TC, Tconditions an Test procedures from test basis

Test control

a. Is an ongoing activity.
b. Compares actual progress against the plan
c. Must respond to changes in mission, strategies, testing objective

Metrics:

Risk and test coverage
Defect information
Planned versus actual hours to develop test ware and execute test cases

2) Test Analysis & Design

a. Identification of Test Conditions by analysis of Test basis
   - Granularity of the test basis
   - Product risks addressed
   - Requirements for reporting and traceability

b. Creation of Test Cases by elaborating identified test conditions
   - Should be repeatable, verifiable and traceable back to requirements.
   - Should have preconditions, Test data requirements, the expected results and post conditions

Metrics: (Requirements are covered by Test conditions, which are further covered by Test cases
- Percentage of requirements covered by test conditions
- Percentage of test conditions covered by test cases
- Number of defects found during test analysis and design

3) Test Implementation & Execution (Test cases to Test procedures + Test data+ Test environment + Test execution schedule

Test Implementation: Ensure all tasks to start test execution re handled
- Test procedures are ready
- Who will do what? (Test roles)
- Entry criteria check before execution
- Prioritization of Test procedures
- Test environment is ready

Challenges
- Sources of test data
- To use those source to obtain test data (data privacy issue + tools test data generation)
- Test environments (in case production like requirements of test data)

Test Execution: Starts when test objects is delivered and entry criteria met

False positive: Failed TC when behaviour was correct
False negative: Passed TC when behaviour was incorrect

Metrics

- Percentage of test environments configured
- Percentage of test data records loaded
- Percentage of test conditions and cases executed
- Percentage of test cases automated

Test design techniques

a) Black box: EP, BVA, State transition, Cause effect Graphing, Syntax testing

b) White box: State testing, brach.decison testing, data flow, LCSAJ.

4) Evaluating Exit Criteria and Reporting→. Includes Test summary report (TSR)

- Bug convergence chart: Bugs find rate vs. backlog
- Test progress chart: Planned vs. actual hours achieved
- Test fulfilment chart: completed TC
- Test coverage chart: covered test basis vs. relative bug found.

5) Test Closure Activities

- All planned TC were executed + defects were closed
- Opportunities to reuse test work products + handover to maintenance
- Lessons learnt document (project retrospective)
- Archiving test work products (safe guarding info)

Metrics

Percentage of test cases run during test execution (coverage) Percentage of test cases checked into re-usable test case repository Ratio of test cases automated: to be automated

Percentage of test cases identified as regression tests

Percentage of outstanding defect reports closed off (e.g. deferred, no further action, change request, etc.)

Percentage of work products identified and archived.

CH-3 Test Management

Risk based testing and FMEA

Product or quality risk: Directly related to test object

Project or planning risk: Related to management or control for project

Risk= Likelihood + impact, where:

- Likelihood comes from technical considerations
- Impact comes from business consideration

Risk guides testing in 3 ways:

a. Test effort, test techniques, test activities sequencing + repair of bugs → Product risk
b. Test planning and management to mitigate and contingency → Project risk
c. Represent test results and project stats in terms of residual risks

Characteristics and benefits of RBT

a. Matches level of testing effort to level of risk
b. Matches order of testing to level of risk.
c. Maintain traceability from testing to risk and defects to risks
d. May report test results in terms of residual risk
e. May compress testing if require in risk order during execution.
f. Includes stakeholders for identifying risk covering gaps in doc.
g. Can blend reactive strategise to detect missing risk during implementation and execution phase?

Risk management activities:

a) Identification: Finding project and product risks
   i. Expert interviews
   ii. Independent assessments
   iii. Use of risk templates
   iv. Project retrospectives
   v. Risk workshops and brainstorming
   vi. Checklists
   vii. Calling on past experience

b) Analysis: Assessing level of risk (impact vs. likelihood)

c) Mitigation (or control):
   a. Mitigation: take preventive measures to reduce the likelihood and/or the impact of a risk.
   b. Contingency: have plan(s) to reduce the impact if the risk becomes an actuality.
   c. Transference: get another party to accept the consequences of a risk (outsourcing).
   d. Ignore or accept the risk and its consequences.

Two types of risk measurements

Quantitatively: Precise info of likelihood and impact (e.g. 90%...insurance sector)

Qualitatively: High, medium, low, etc

Project risk mitigation

- Early preparation of test ware
- Tough entry criteria
- Early review
- Monitoring project progress

Product risk mitigation

- Early review
- Priority of test cases (Depth/Breadth first)
- Monitoring project progress

Two industry standard

a) ISO/ IEC=61508
   a. Build quality from beginning and not at the end
   b. Reduce intolerable risks
   c. Tester independence to avoid biasing
   d. Properly document for audit.

b) Do-178B or ED-1213 (Aviation related)

Risk identification techniques

a) Formal: more costly

Requirements review- What system should do but quality risk identification' & assessment = what system might do that it should not.

b) Informal: Easy, light weight, flexible

Disadvantage': Wrong participants can give wrong results

Quality risk techniques

a) ISO 9126 standard (FRUEMP)
Functionality; Reliability; Usability; Efficiency; Maintainability; Portability; compliance

b) Cost for exposure: Likelihood of loss + average cost of loss

c) Hazard analysis: Try to understand hazards that create risks for our systems

Used for medical or embedded projects

Depth first: TC execute in strict risk order

Breadth first: Cover every risk at least once

FMEA → For safety critical system

(FCE → Failure modes, Causes, Effects)

- Iterative approach
- Select a function and find its possible failure modes
- Define causes of those failures and their effect on users
- Disadvantages:
  - Documentation heavy
  - Lengthy output
  - Heavy maintenance
  - Hard to learn
- Usefulness
  - Used for Critical project (Avionics, nuclear)
  - Mandatory or regulatory requirement apply
  - Risks of project delay is unacceptable
  - Complex and safety critical projects

Example:

- Inappropriate credit card access → Failure mode (FM)
- CC Data not encrypted → Cause of FM
- Fraud charges → Effect of FM

Three facts about RBT

a) Not all tests should find bugs…some should built confidence

b) Preventive approach to testing

c) Risk items and risk level evolve as the project goes on and needs to be realigned.--> Ongoing process

Challenges of RBT

- Building agreement and consensus on the level of risk
- Building agreement and consensus on the priority of risk
- Making people take rational decisions about risks
- Ongoing time investment
- Challenge in adapting to change.

Planned testing → 60-70%

Unplanned testing (30-40%) includes:

- Confirmation testing
- Reactive testing (exploratory testing)

01 Test management documentation

a) Test policy: Organization philosophy towards testing and Quality assurance (**WHY**)

- Why of testing for an organization
- Objective that organization w ants to achieve
- Company management establishes or approves this document
- Short HL document that including definition of testing, fundamental Test process, Quality targets etc

b) Test strategy: Organizations general methods of testing (including. Project and Product risks) (**HOW @org' level**)

- How of testing for an organization
- Not project dependent but approach can be tailored.
- Should be consistent with Test policy
- Dividing testing into levels
- High level testing activities i.e. E&E criteria of each level
- Independent of any project but lays out approach to testing for most projects

Two types:

Preventive (V-model):

Reactive (Exploratory): Test design after system has been produced or responding to system as it is presented to test team.

Types of strategies

- **Analytical**: RBT preventive I nature
- **Model based**: operational profits
- **Methodical**: as per check list of quality char'.
- **Process or standard compliant**: Follow process or standards from a standards committee.
- **Dynamic or heuristic**: Based on bug taxonomies or S/W attach
  - Making educated guesses
  - Min structure; max flexibility
  - Focus on finding bugs rather than building confidence
- **Consultative: user determines what to test**
  - Serves as hands and eyes of another group i.e. outsourcing O/P: validation
- **Regression**: functional test automation

Risk of using templates

- May turn people brain off
- People assume one size fits all

c) Master Test plan or Test approach: (**HOW @Project/Product level**)

- How of testing for a single project but spanning multiple levels.

- Deviation from Test plan and Test strategy should be explained.
- Application of Test Strategy for a particular project.
- Includes particular levels and relationship between them.
- Should be consistent with Test policy and Test strategy document.
- Mentions items to be tested and not to be tested
- Mentions quality attributes to be tested and not to be tested
- Test schedule and budget + SARS

Level Test plan (or phase test plan): Particular test activities within each test level including expanding the Master test plan for this specific test level.

- How of testing for a level.
- Schedule, tasks and milestone details not necessarily covered in Master test plan.

---

02 Templates

a) IEEE829: Test plan Template + T management documentation
b) Others: IEEE 829: TDS Test design spec: Collection of Test cases or Tconditions at high-level (also called test suite)
c) Others: IEEE 829: TCS Test case spec: details of a TC
d) Others: IEEE 829: TPS Test design spec: How to run/execute one or more TC (Test steps).
e) Others: IEEE 829: T-ITS (Test items transmittal report: Items being delivered for testing (release notes)

PS: Testing is downstream: Chaos and disorder in other areas of project can leak into the testing effort.

Test logs are kept to track events leading to deviation from test plan.

TEST ESTIMATIONS

Creation of approx target for

- Cost
- Completion dates

Includes all 5 activities of fundamental test process

Factors influencing the cost effort and duration of Test activities

- Required level of quality of system → Delaying factors
- Size of project → Delaying factors
- Historical data → Delaying factors
- Process factors → accuracy of project estimates
- Material factors → Test automation of tools Test environment and Test data.
- People factors → managers, Test Leader, Senior management commitment, skills in team, etc

03 Test estimation techniques

Bottom up: estimate each task individually an then drive the project estimate from those estimates

Top down: Estimate test effort all at once (i.e. 20% of whole project estimates)

- Intuition and guesses
- Past experience
- Work-breakdown-structures (WBS)
- Team estimation sessions (e.g., Wide Band Delphi)
- Three point estimate
- Test Point Analysis (TPA)
- Company standards and norms
- Percentages of the overall project effort or staffing levels (e.g., tester-developer ratios)
- Organizational history and metrics
- Industry averages and predictive models such as test points, function points, lines of code, estimated developer effort, or other project parameters

WBS: Decompose Test project into activities, tasks, subtask etc
TPA: User for system and acceptance testing
Part of T-map
# of Test points of a project is calculated base on (STP)
- Size of project
- Test strategy
- Productivity level of participants (skills of Test team)

Negotiations and reduced test scope

- Use risks to reduce Test scope --> preserve depth; reduce breadth approach
- Reduce extent of testing while maintaining some amount of coverage --> preserve breadth; reduce depth
- Postpone automation or tools development
- Outsource some part of project or testing
- Reduce project scope (if no budget and delay schedule is possible.

---

04 Test planning → benefits (PDTCC)

- Should be done in close cooperation with development
- Detection and management of project risks (e.g. due to bad assumption)
- Detection and management of product risks (missing performance and reliability info)
- Detection of problems in project plan
- Opportunity to increase staff, budget and effort
- Identification of critical component and accordingly ensure early delivery of those components.
- Should be done in close cooperation with development

---

05 Test progress monitoring and control is done on:

- Product risks → measurable and quantifiable. Metrics below:

- Number of remaining risks (including. type and level of risks)
- Number of risks mitigated (including. type and level of risks)

- Defects → measurable and quantifiable. Metrics below:
  - Cumulative number reported (identified) versus cumulative number resolved (disposed) Mean time between failure or failure arrival rate (MTBF)
  - Breakdown of the number of defects associated with: particular test items or components; root causes, sources; test releases; phase introduced, detected or removed; and, in some cases, owner
  - Trends in the lag time from defect reporting to resolution
- Tests → measurable and quantifiable. Metrics below:
  - Total number planned, specified (developed), run, passed, failed, blocked, and skipped
  - Regression and confirmation test status
  - Hours of testing planned per day versus actual hours achieved
- Coverage → measurable and quantifiable. Metrics below:
- Confidence → Subjective. Metrics below:
  - Requirements and design element coverage
  - Risk coverage
  - Environment / configuration coverage

Benefits of metrics

- Increment resources if budget flexibility
- Extend testing end date of schedule flexibility
- If neither, try to change scope of testing or of project (See negotiations topic above)
- Ease up exit criteria or relax test exit criteria to declare victory.

---

06 Business value of testing in 2 ways

- Qualitative values: improved reputation for quality + reducing risk of loss and smooth releasing
- Quantitative values: finding defects (prevent or fix) prior to release)

Quantitative value can be measured by cost of quality:

- Cost of prevention→ dev training
- Cost of detection → expense for test planning, design and implementation)
- Cost of internal failure → because bugs were detected + development fixed bugs.
- Cost of external failure → expense as bugs were not detected; business lost

---

07 Distributed - outsourced – In sourced

Distributed: multiple locations

Outsourced: not collocated and not fellow employee of rest of the project team

In sourced: collated but not fellow employee

Challenges and Issues

- Clear channel of communication
- well defined expectation

- need for alignment of methodologies
- division of work across multiple location
- develop and maintain trust
- cultural and languages are different + diff business culture too
- Issue: outsource resource is focussed and not looking for more work on your billing

Don't underestimate

- Power of travelling
- Compounding difficult of managing multiple time zones
- More and different project risks

SPICE: (alike CMMi)

- S/W process improvement and capability determination
- Framework for assessing S/w processes model

---

08 Test management Issues

a) Issue for exploratory testing (reactive and experience based test techniques)

1) SBTM (Session based Test management → Strategies has following 3 points)

- Sessions et-up
- Test design and execution
- Defect investigation and reporting

Session agenda

What worked, what didn't worked, bugs found and were it interesting? Or **PROOF**

Past: What happened during the session? What did you do, and what did you see?
Results: What was achieved during the session? What bugs did you find? What worked?
Outlook: What still needs to be done? What subsequent test sessions should we try?
Obstacles: What got in the way of good testing? What couldn't you do that you wanted to?
Feelings: How do you feel about this test session?

b) Systems of system issues

- Test management is more complex (diff location; diff org) with diff life cycle models
- Configuration, change and release management must be formally defined
- Management of Test environment and Test data
- Cost of simulators
- Dependencies among different parts

c) Safety critical systems

- Industry specific standard apply normally
- Demonstrate compliance (requirements traceability, Test coverage) due to liability issues

- Audits may be required for organisation structure compliance
- Life cycle are sequential in nature and pre-defined
- Non-functional attributes for critical system **RAMS**
    - Reliability
    - Availability
    - Maintainability
    - Safety and security

d) Others – Non- functional issues

The following factors can influence the planning and execution of non-functional tests:

SRR-OCD

- Stakeholder requirements
- Required tooling: for performance, efficiency and security tests
- Required hardware: Prod like environment or dedicated labs
- Organizational factors
- Communications
- Data security: encryptions

Ch 4- Test Techniques

1) Specification-based (Behaviour-based or black-box)→ Test analysts & Technical test analysts

Deriving Tconditions or TC's based on test basis doc' and without reference to its internal structure.

- Equivalent Partitioning
- Boundary value analysis (BVA)
- Decision table testing and Cause-effect graphing
- State transition testing
- Classification tree method, orthogonal arrays and all pairs tables
- Use case testing

2) Structure-based (or white-box) → Technical test analysts

- Tests derived systematically from the structure.

Information about how the software is constructed is used to derive the test cases

- Statement testing
- Decision testing
- Branch testing
- Condition testing
- Multiple condition testing
- Condition determination testing
- LCSAJ (loop testing)
- Path testing.

3) Defect based → Test analysts and Technical test analysts

Type of defect found is used as the basis for test design.

Taxonomies (categories & lists of potential defects)

4) Experienced-based techniques → Test analysts and Technical test analysts

Used when

- no or poor specification documentation
- Insufficient time

Utilize testers' skills and intuition, along with their experience with similar applications.

- Error guessing → without formal test documentation
- Checklist-based
- Exploratory → with formal test documentation
- Attacks – attempting to force specific failures to occur.

5) Static Analysis

Testing without actually executing the software under test (may relate to code or system architecture)

Static Analysis of Code

Control Flow Analysis
Data Flow Analysis
Compliance to Coding Standards
Generate code metrics

Static Analysis of Architecture

Static Analysis of a Web Site
Call Graphs

6) Dynamic analysis

Analyze an application while it is running.

- Detecting Memory Leaks: memory is allocated but subsequently not released (symptoms – steady worsening of response time)
- Detecting Wild Pointers: Pointers "lost" the object to which they should be pointing to.
- Analysis of Performance: Also called as "performance profiling".

Ch-5: Testing of Software Characteristics

Quality attributes for domain testing → **WHAT** it does (ASU – FSA)

- Accuracy: application's adherence to the specified or implied requirements
- Suitability: evaluating and validating the appropriateness of set of functions for its intended tasks
- Interoperability: can function correctly in all intended target environments
- Functional security: Penetration testing -prevent unauthorized access; accidental or deliberate
- Usability: why users might have difficulty using the proposed software system
- Accessibility: accessibility to those with particular requirements or restrictions in its use.

Quality attributes for technical testing→ **HOW** the product works (TREMP)

- Technical security: Concerns unauthorized access
- Reliability: monitor a statistical measure of software maturity over time and compare this to a desired reliability goal
- Efficiency:
  - **Performance Testing**: ability to respond within a specified time and under specified conditions
  - **Load Testing**: ability of a system to handle increasing levels of anticipated realistic loads
  - **Stress Testing**: ability of a system to handle peak loads at or beyond maximum capacity.
  - **Scalability Testing**: ability of a system to meet future efficiency requirements
- Maintainability: Ease, with which software can be analyzed, changed and tested.
- Portability: Ease with which software can be transferred into its intended environment.

Ch-6: Reviews

- Can be done on all types of documentation
- Type of static testing
- Objective is Detection of defect + building confidence
- Reviews are not equal to static analysis as static analysis uses tools!
- Reviews are done on documentation, before he code has been written, whereas static analysis is done on code without execution and done after code has been written)

Outputs of reviews:

- Unchanged or minor change → no follow-up
- Changed but no further review required
- Extensively changed and a further review is necessary

Review Types

1) Informal reviews
   a. Cost effective and very flexible.
2) Walkthrough:
   a. Author is moderator
3) Technical review
   a. also called as Peer review)
4) Inspection: (Formal + requirementsuires moderator + E/E criteria)
   a. Management may not attend
5) Management review
   a. Carried out by management or stakeholders
   b. Procedures includes assessment of project risks
   c. Outcome and decision are documented
6) Audit – very formal
   a. To demonstrate conformance to some sort of applicable standards or contractual obligations
   b. collect evidence of compliance
   c. Outcome includes observations, recommendations, corrective actions

Reviews as work products

1) Contractual reviews
   a. Associated with a contract milestone
   b. Type could be Management review
2) Requirements reviews
   a. Walkthrough or technical review or inspection
3) Design reviews
   a. Type could be Technical review or Inspection
4) Acceptance
   a. Type could be Management review or audit

Performing a review

1) Planning
2) Kickoff
3) Individual preparation
4) Review meeting

6) follow-up

Introducing review:

- Secure management support
- Educate managers
- Put structure in place (Document review including metrics)
- Train on review techniques and procedures
- Obtain participant support
- Do some pilot reviews
- Demonstrate the benefit:
- Apply reviews to all (or at least the most important) documents

Major benefits

- Reducing or avoiding costs
- Saving elapsed time by finding and fixing defect early

Success factors

a) Technical

- Record costs versus benefits achieved
- Review early drafts to identify patterns
- Organization specific checklist of common defects
- Focus on content and not format

b) Organizational

- Ensure management allows adequate time on review activities and fixing defects found
- Don't use these metrics for appraisal
- Right mix of pp are present in review
- Training in reviews

c) People

- Find defects in a blame free environment
- Comments are objective and not subjective

Ch-7: Incident Management

Incident: Unexpected occurrence that requires further investigation OR
It is recognition of a failure caused by a defect.
May or may not result in the generation of a defect report.

Defect: - Actual problem that has been determined and requires resolution by changing the work item.
- Can be detected thru static testing

Failure: - Deviation from expected result.
- Can only be detected thru dynamic testing.

Defect Lifecycle IEEE 1044-1993) → RIAD- RCI

1) Recognition: recording data items when a potential defect (incident) is discovered.
2) Investigation: Investigated defects to find any related issues and propose solutions.
3) Action: resolve the defect + improving processes to prevent future similar defects
4) Disposition: either closed, deferred, merged or referred to another project

Each step further has:

i. Recording
ii. Classifying
iii. Identifying Impact

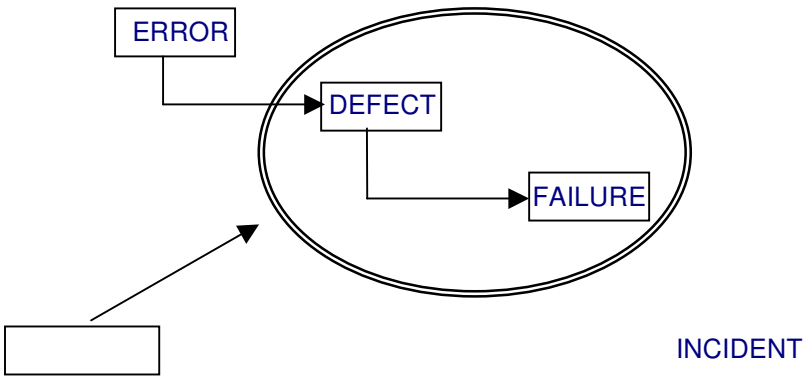
Communicating Incidents

1) Free from accusation
2) Defect prioritization via defect triage meetings (i.e. sorting via priority)
3) Document objective factual info only.

Metrics

1) defect density analysis,
2) found vs. fixed metrics
3) convergence metrics (open vs. closed)
4) Defect trends

e.g code



CH 8 - Standards & Test Improvement Process

- Standards removes guesswork
- Project cud be better funded and supported buy management
- Promotes preventive testing strategy
- Provides adaptable process checklists

General aspects on standards

1. **Standard sources →**
    - **a.** International
        - **i.** IEEE 829: S/W T documentation (Test plan, Test strategy,, etc)
        - **ii.** IEEE 1028: S/W reviews
        - **iii.** ISO 15504: SPICE = ~ CMMi
        - **iv.** ISO 12207: S/W Life cycle process
    - **b.** National
        - **i.** BS 7925-2 (EP, BVA, State transition, Stat, Branch, condition testing, etc)
    - **c.** Domain specific
        - **i.** RTCA DO-178B/ED 12B: Avionics systems
        - **ii.** ECSSL Space industry (including SFMECA, etc)
        - **iii.** Title 21 CF8 Part 820: Food and Drug admin (medical system)
    - **d.** Others
        - **i.** Tailored
        - **ii.** Adaptative
        - **iii.** In-house (common best practices)

2. **Sources Usefulness**: Focus on min defect introduction rather than fixing them via testing
3. **Conflicts**: standard may be conflicting info between them

Test Improvement Process

As testing improves s/w → s/w quality processes are selected and used to improve the process of s/w.
Provides guidelines and area of improvement

RIP models: TMM, TPI, CTP and STEP

TMM and TPI are used for bench mark comparisons (prescriptive models)
CTP and STEP are non - prescriptive models.

Intro of PI → relevant to s/w dev process as well as testing process

Demining improvement cycle PDCA: PLAN, DO, CHECK, ACT

Types of Process Improvement

1) Process reference models (or prescriptive; Staged TMM or continuous TPI)
    - Used as a framework when an assessment is done.
    - Evaluate an organization's capability compared with the model.
2. The content reference model (non-prescriptive).
    - Used to improve the process once the assessment is done.

Improving the Test Process

STEP, TMMi, TPI and CTP:

- All four of these test process assessment models allow an organization to determine where they stand in terms of their current test processes.
- Once an assessment is performed, TMMi and TPI provide a prescriptive roadmap for improving the test process.
- STEP and CTP: determine where a org' biggest process improvement ROI will come from

Generic Process steps for review: IMPROVE

Initiate
- Confirmation of the stakeholders, the GSOC (goals, scope, objectives & coverage of the PI is agreed.
- The choice of the process model upon which the improvements will be identified is also made during this activity.
- Success criteria should be defined and a method by which they will be measured throughout the improvement activity should be implemented.

Measure
- Performing the assessment

Prioritize and Plan
- The list of possible process improvements are put in order of priority (based upon ROI, risks, alignment to organizational strategy, measurable quantitative or qualitative benefits).
- Plan for the delivery of the improvements is developed and deployed.

Define and Redefine → Implement the PI's
- New processes are defined
- Existing processes are redefined

Operate → Deploy the PI's
- Once developed, the process improvements are deployed.
- This could include any training or mentoring required, piloting of processes and ultimately the full deployment of them.

Validate
- Benefits agreed are validated using metrics etc. (benefit realization)
- Success criteria for the process improvement activity have been met.

Evolve
- monitoring of the next level of maturity starts
- Decision to either start the improvement process again, or to stop the activity at this point.

1) TMM --> IDIMO (Staged)

- Complement CMM (fills testing related gaps of CMM).
- Provides both a process r**eference model and a content reference** model.

Level 1: Initial: no formally documented or structured testing process.
Testing is same as debugging.

Level 2: Definition:
- setting testing policy and goals
- introducing a test planning process
- Implementing basic testing techniques and methods.

Level 3: Integration:
- testing process is integrated into the software development lifecycle
- Test training for staff.

Level 4: Management & Measurement:
- testing is thoroughly defined
- Peer review occurs.
- Testing process is capable of being effectively measured, managed, and adapted to specific projects.

Level 5: Optimization:
- testing is completely defined
- controlled in terms of effectiveness and efficiency via appropriately metrics
- data from the testing process can be used to help prevent defects, and the focus is on optimizing the established process

TMMi is successor of TMM --> complementary to the CMMI → i.e. Fills in the testing gaps let out by CMMi.

2) TPI --> ICEO - uses a continuous representation rather than the staged representation of TMM.

- TPI is primarily a process reference model (prescriptive)

- 2 processes apply to all cornerstones – (cornerstone concept is TMap concept).
- Key areas can be evaluated at a level between A to D, A being low
- The level achieved is assessed by evaluating checkpoints defined in the TPI model.
- A Test Maturity Matrix maps the levels for each key area to an overall test process maturity level (i.e. Controlled Efficient Optimizing)
- Initial : 0
- Controlled: 1-5 (test plan, Test strategy, Test levels, trainings for staff)
- Efficient: 6-10 (o= improve efficiency, automate testing)
- Optimization: 11-13

3) CTP

CTP is primarily a content reference model
CTP is flexible model and allows tailoring.

- The model identifies twelve critical testing processes.
- Assess existing test process. (Strong, weak), and provides prioritized recommendations for improvement based on organizational needs.
- Quantitative metrics commonly examined during a CTP assessment:
    - Defect detection percentage
    - Return on the testing investment
    - Requirements coverage and risk coverage
    - Test release overhead
    - Defect report rejection rate
- Qualitative factors commonly evaluate:
    - Test team role and effectiveness
    - Test plan utility
    - Test team skills in testing, domain knowledge, and technology
    - Defect report utility
    - Test result report utility
    - Change management utility and balance

4) STEP → Could be combined with TPI sometimes

STEP (Systematic Test and Evaluation Process), like CTP and unlike TMMi and TPI, does not require that improvements occur in a specific order.

- STEP is primarily a content reference model.
- Basic premises of the methodology includes 7 assumptions:
    - A requirements-based testing strategy
    - Testing starts at the beginning of the lifecycle
    - Tests are used as requirements and usage models
    - Testware design leads software design
    - Defects are detected earlier or prevented altogether
    - Defects are systematically analysed
    - Testers and developers work together
- Stresses “test then code" using a requirements-based Test strategy to ensure early creation of test cases validates the requirements specification prior to design and coding.
- The methodology identifies and focuses on improvement of three major phases of testing (**PAM**):
    - Planning

- Measurement → Test execution and Test reporting

- Quantitative metrics includes:
  - Test status over time
  - Test requirements or risk coverage
  - Defect trends including detection, severity, and clustering
  - Defect density
  - Defect removal effectiveness
  - Defect detection percentage
  - Defect introduction, detection, and removal phases
  - Cost of testing in terms of time, effort, and money
- Quantitative factors includes:
  - Defined test process utilization
  - Customer satisfaction

5) CMMI → IMDQO (initial, managed, defined, quantitatively managed, optimization)

Two approaches

- the staged representation
  - Five “levels of maturity”, each level building upon the process areas established in the previous levels.
  - ensure commonality with CMM
- continuous representation
  - Organization is allowed to concentrate its improvement efforts on its own primary areas of need without regard to predecessor levels.
  - More flexible

CH-9 Test Tool Concepts

- Test tools can greatly improve the efficiency and accuracy of the test effort
- Cost benefits and Risks of Test Tools and Automation

Initial costs

Knowledge acquisition (learning curve for tool)
- o Evaluation (tool comparisons) when appropriate
- o Integration with other tools
- o Consideration of initial costs for purchase, adaptation or development of tool

Recurring costs → Pilot projects often miss this cost
- o Cost of tool ownership (maintenance, licensing fees, support fees, sustaining knowledge levels)
- o Portability
- o Availability and dependencies (if lacking)
- o Continual cost evaluation
- o Quality improvement, to ensure optimal use of the selected tools

Additional Benefits of automation:

- Repeatable low maintenance automated execution over month n yrs.
- Predictable test execution time
- Regression testing and defect validation are faster and safer
- Coverage of certain test types which cannot be covered manually (e.g., performance and reliability)
- Useful for iterative/Incremental development (for regression).

Additional Risks:

- Unrealistic expectation for the tool
- underestimating the time cost and effort required to introduce the tool
- underestimating the time and effort needed to achieve a positive return on investment (ROI)
  - effort requirements to maintain the test
  - Over reliance on tool
  - Incomplete or incorrect manual testing is automated as is
  - The test ware is difficult to maintain (e.g. scripts)
  - Fail to monitor ROI

PS: Biggest cause to automation failure: Failure to think ahead in terms of how the **test can be maintained**.

Test automation strategies
- Automate for long term
- Built maintainable automated system
- only automate test that are automatable
- automate test that are error prone if done by human
- automate test for which there is a business case (meaning that needs to be repeated multiple time etc)

The Concept of Test Oracles

- used to determine expected results
- e.g. old system is being replaced by a new system with the same functionality

Test Tool Deployment

- Should be documented and tested regardless of whether it is purchased as-is, adapted or created in house
- Do a cost-benefit analysis that includes initial implementation as well as long-term maintenance.
- Redefine manual TC for automation
- Test tool training for max' benefit.

Developing Your Own Test Tool
- Should be documented in a way that they can be maintained by others.
- Review the purpose, aim, benefits and possible downside before spreading it across an organization.
- Expanded far beyond the initial use due to new requirements and become unmanageable.

Automation scripts: Allows testing of all possible combination of T/P which would be infeasible with manual testing.

Test Tools Categories

a) Test Management Tools → Traceability of test artefacts

- Used by TM, TA, TTA
- Data regarding the execution of concurrent test suites
- Various testing related Metrics

b) Test Execution Tools → mostly used by Test analysts and Technical Test analysts to automate regression tests.

- to reduce costs (effort or time),
- to run more tests,
- To make tests more repeatable.

C) Debugging & Troubleshooting Tools

- To narrow down the area where a fault occurs.
- Mainly used by Technical Test Analysts.

d) Fault Seeding & Fault Injection Tools → by Technical Test Analysts

PS: to create single or limited types of code faults in a systematic way.

FI: deliberately (re-)injecting a particular fault to check if

1) The software can cope with it (fault tolerance) or
2) To evaluate that a test in a test suite finds the deliberately inserted fault.

e) Simulation & Emulation Tools → by Test Analysts and Technical Test Analysts

Simulators: to support tests where code or other systems are unavailable

Emulators: software written to mimic the hardware

f) Static and Dynamic Analysis Tools → Technical Test Analysts use these tools.

Static analysis tools

Static Analysis tools report their findings in terms of warnings.
Technical Test Analysts use these tools.

Dynamic analysis tools

Dynamic Analysis tools provide run-time information on the state of the executing software.

Used to identify unassigned pointers, use and de-allocation of memory, to flag memory leaks

g) Keyword-Driven Test Automation → used by domain experts and Test Analysts.

- Efficient test case specifications as keywords are defined by domain experts.
- Easy to maintain TC.
- TC spec are independent of their implementation

h) Performance Testing Tools → used by Technical Test Analysts.

Two main facilities:

Load generation: simulating large numbers of multiple users ("virtual" users) with specific volumes of input data.
Measurement and analysis of system response to a given load

- Numbers of simulated users
- Number and type of transactions generated by the simulated users
- Response times to particular transaction requests made by the users Reports based on test logs, and graphs of load against response times.

i) Web Tools

CH 10 People Skills – Team Composition

Individual Skills

- tester's individual knowledge base--> Testing skills, tech and s/w skills, user, business n domain skills
- Knowledge of the software development process -->
- Specific software testing skills (analysis, risk analysis)
- Experience in project management (especially for Test manager)
- Interpersonal skills, such as giving and receiving criticism, influencing, and negotiating

Test Team Dynamics

- complement the skills and personality types that already exist within the test team
- Handling the interpersonal interactions with the other project team members.
- Cross-training within the team is required to maintain and build the team knowledge and increase flexibility.
- Each person should be given a defined role on the team.

Fitting Testing Within an Organization

- No independent testers: developer is testing his own code
- Testing is done by a different developer than the one who wrote the code
- Testing is done by a tester (or test team) being part of the development team
- Testers from non-development technical organization
- External test specialists perform testing on specific test targets.
- Testing is done by an organization external to the company

Outsourcing is one of the forms of using an external organization.
Outsourcing brings challenges, particularly when external to your country

- Cultural differences
- Supervision of outsource resources, Transfer of information, communication
- Skills sets, skill development and training
- Employee turnover Accurate cost estimation Quality

Motivation

- Recognition for the job accomplished
- Approval by management
- Respect within the project team and among peers
- Adequate rewards for the work done (including salary, merit increases and bonuses)

De-motivators

- product ship before it should, due to external influences with poor quality
- No recognition they feel is due for a job well-done.

Three Communication levels:

- Documentation of test products: test strategy, test plan, test cases, test summary reports, defect reports, etc.
- Feedback on reviewed documents
- Information gathering and dissemination: interaction with developers, other test team members, management, etc.