Derivation: the creation of requirement based on an analysis of higher-order requirements (ex. availability requirements)

Allocation: the assignment of a requirement to one or more system elements within the architecture. (ex. thrust requirement > propulsion requirements)

Flow Down: assigning requirement to one or more lower-level elements. (ex. water resistant requirement > all elements exposed to outside environment)

Traceability: the creation of linkage from one system elements to others (ex. mtbf requirements > availability requirements)

System engineer must know how:

- test able

- Produced

- packaged

- transported

Pre-planned product improvement requirements (P3I)

Sys Engg mush work with reliability SMEs

RFA - request for action

RFI - request for information

RVTM - requirements verification and traceability matrix

How to write requirement statement:

* Specific intent
* Desired capability
* Desired level of performance
* Functions or behavioral characteristics
* Not specifying how functions/behaviors are performed (calling for specific components)
* requirement should not specify specific piece of hardware of software into the system
* Determining proper hardware or software is not job description of the systems engineer. It is job of SMEs.
* each requirement should be unambiguous (all parties have same understanding of the requirements)
* requirement should be complete (stand alone, independent of other requirements, avoid cross referencing other requirements, avoid pronoun)
* requirement must be feasible (should not request impossible or unrealistic behavior or feature, some time all req are individually feasible but collection of them are not feasible)
* requirements should be verifiable (analysis, demonstration, test or inspection)
* requirement can use illustration/graphics

Architecture Definition

* it is abstract representation of system elements
* architecture should be implementation free
* is being performed as enabled for other process
* It is primary artifact for the system requirements specifications etc.

Primary goal for architecture definition:

* produce alternate concepts
* compare concepts in trade studies
* select one or more concept alternative(s)

Inputs to architecture definitions:

* Life cycle concepts
* system function definition
* system requirements
* system functional interface identification
* System requirements traceability
* Updated RVTM
* Design traceability
* Interface definition update identification
* Life cycle constraints.

Goes into architecture definition

* Prepare for architecture definition
* Develop architecture viewpoints
* Develop models and views
* Relate the architecture to the design
* Assess architecture candidates
* Manage the selected architecture

Output of architecture definition

* Architecture definition strategy
* System architecture description
* System architecture rationale
* Documentation tree
* Preliminary interface definition
* Technical performance measure needs and data
* Architecture traceability
* Architecture definition record.

Architecture views

* operational (user's perspective)
* logical (acquirer/customer's perspective)
* Physical (designer's perspective)

Architecting is abstract/inductive. More art than science designing is deductive. More science than art

Architecture is not a modeling language.

Model based systems engineering (MBSE)

* systems engineering (requirement development, functional and mission development, physical definition)
* MBSE tool
* MBSE language (sysML)

A thing that performs a function is called physical including software

System model domains -

* Requirement domain: description of mandatory things your widget needs to do
* Behavioral domain: all the ways your widget should behave goes
* Structural domain: what your widget is made out of is in structural domain

Interfaces: (forces, energy, data, signals or materials)

SysML - Systems modeling language and UML - unified modeling language:

* Modeling language: artificial language of communicating concept in standardized way
* agreed upon grammar and vocabulary
* mostly graphical

Unified modeling language:

* general purpose modeling language for software engineer that is intended to provide a standard way to visualize the design of a software system

System modeling language:

* general purpose modeling language for systems engineer that is intended to provide a standard way to visualize the design of a system (hardware and software)
* it adds two new diagram (requirements and parametric)

Type of diagrams:

* Structure Diagram types
  + Bdd - block definition diagram
  + ibd - internal body diagram
  + par - parametric diagram
  + pkg - package diagram
* Behavior diagram types
  + act - activity diagram
  + seq - sequence diagram
  + stm - state machine diagram
  + ucd - use case diagram
* Requirement diagram type
  + req - requirement diagram

BDD (block definition diagram)

* it is a structure of system showing system blocks, their attributes, property and interactions
* analogue to class diagram in UML
* good for systems hierarchy

IBD (internal block diagram)

* internal structure of single structure of the block
* inputs on the perimeter of diagram are brought in

Activity diagram

* Depicts system behaviors, sequencing, decision and other logic between system behaviors
* functional interfaces (force, energy, data, signal, material)
* control flow is dotted line and object flow is solid line
* decision node needs logic

Sequence diagram (never saw using this diagram)

* blocks
* relation between blocks and actors
* synchronous messages or call messages

State machine

* behavioral diagram
* on state, off state, degraded state
* nested state (substate)
* start point and end point are important

Parametric diagram

* performance, reliability and SWaP analysis (size weight and power)

Package diagram

* hierarchy diagram
* architecture diagram
* requirement domain, behavior domain and structural domain\

UML diagrams:

* Behavioral Diagram
* Activity diagram
* State machine diagram
* Interaction diagram
* Timing diagram
* Interaction overview diagram
* Communication diagram
* Sequence diagram
* Use case diagram

Structure diagram

* component diagram
* class
* composite
* deployment
* package
* object

Behavioral/functional domain:

* function/behavior and activity are synonymous in sysengg

Function is an action performed by system element that converts inputs to output. Function:

* uses input
* generates output
* based on requirements/context/scenario
* decomposes/decomposed by sub-function
* performs/performed by component

Activities vs action

* Activities are overarching system behavior, composed of actions. These are decomposed into lower-level actions.
* Actions are leaf level system behavior. Actions are not decomposed. Actions are basic units of functionality.

Activity must have at-least one input and one output.

Outputs are always different than inputs

Sequence diagram always draws relation between physical elements from functional perspective.

Functional block diagram:

- it depicts relation between internal elements to external entity

Use case:

it is a list of steps user will most likely deploy, operate, support or dispose of a system to achieve a desired goal.

* helps refine requirements.
* foster top-down design
* preempt errors and faults
* save time and money preventing faults/ errors from existing
* help decompose the problem into more manageable segment

Elements of an usecase:

* Actors: a stakeholder / one or more actors.
* Summary: description of an use case in paragraph from
* goal: primary outcome desired by an actor
* pre-condition: things that must be true prior to starting the use-case guarantees: the least the system can do for the actor, if the goal is not met.
* trigger: step 1 in course of an environment
* course of event: numbered list of actions the actor/s and system take to accomplish the goal.
* alternative path:

Primary source of function:

* Requirement specifications
* Context Diagram
* Life Cycle Concepts
* Legacy and predecessor system functions
* Parent functions

Top-down systems engineering:

* The behavioral domain is bases on
  + the requirements domain
  + the mission domain
  + itself
* the behavioral domain is not based on
  + the physical domain

Top-down sys engg

* Needs, technology driver, context basis of:
  + life cycle concepts and requirements basis of:
    - functions basis of:
      * hardware/ software

Hardware and software are based on functions. Functions are not based on hardware and software.

Stakeholder needs and requirements definition focus:

* high-risk systems ands subsystem entities
* defining systems and subsystem performance requirements/MOEs
* defining the lifecycle concepts

Architecture definition focus:

* Decompose system and subsystem functions until makes/buy decision can be made at component level
* Mitigate function-related risks into the green
* Complete the rest of the functional architecture for non-primary-mission scenarios, fail modes and recovery operations
* Heavy use of analysis, modeling and simulations

Interfaces:

* Force
* Energy
* Data
* Signal
* Material

Functional block diagram does not show logic or sequence, but it shows activities from one to other.

Primary source of function:

* Requirement specifications
* Context diagram
* Life cycle concepts
* Legacy and predecessor system functions
* Parent function

Requirement development -> functional and mission development -> Physical design

Validating system concept:

* Evaluate system performance using models, sims and prototypes
* Perform system level conceptual analysis and simulation “effectiveness” runs through missions, scenarios and use cases
* Primary focus is on mission critical performance attributes (KPP)
* Prototype may be used to demonstrate critical technologies
* Rely on independent evaluations and user feedback!

System/ product development process (INCOSE)

* Agreement Process
  + Acquisition process
  + Supply process
* Enterprise Process:
  + Enterprise environmental management process
  + Investment management process
  + System lifecycle management process
  + Resource management process
  + Quality management process
* Technical management process:
  + Project planning process
  + Project assessment and control process
  + Decision management process
  + Risk management process
  + Configuration management process
  + Information management process
  + Measurement process
  + Quality assurance process
* Technical process:
  + Business/ mission analysis process
  + Stakeholders needs and requirement definition process
  + System requirement definition process
  + Architecture definition process
  + System analysis process
  + Design definition process
  + Implementation process
  + Integration process
  + Verification process
  + Transition process
  + Validation process
  + Operation process
  + Maintenance process
  + Disposal process
* Organizational Project-enabling processes:
  + Life cycle model management process
  + Infrastructure management process
  + Portfolio management process
  + Human resource management process
  + Quality management process
  + Knowledge management process

# Design Definitions

Inputs:

* Lifecycle concepts
* System function definitions
* System requirements
* Functional interface identification
* System architecture description
* System architecture rationale
* Preliminary interface definition
* Preliminary TPM needs and data
* Architecture traceability
* Interface definition update identification
* Implementation traceability
* Life cycle constraints

Process:

* Prepare for design definition
* Establish design characteristics and design enablers related to each system elements
* Assess alternatives for obtaining system elements
* Manage the design

Output:

* Design definition strategy
* System design description
* System design rationale
* Interface definition
* TPM needs and data
* Design traceability
* System element description
* Design Definition record

# System Analysis Process

Concept

* Business/ mission analysis
* Stakeholder needs/ requirement definitions
* System requirement definition
  + These are reviewed and approved at system requirement gate review
* Architecture definition

Agreement process: In this process company hire potential external supplier to help with the system requirement generation and analysis may be.

System Development:

* Design definition
* Implementation
* Integration
* Verification

System Production:

* Validation
* Transition

Utilization

* Support
* Operation

Maintenance:

Request for Proposal:

* Also called as solicitation
* Formal means of initiating a contract for products and services from experts
* Can include purchasing already-existing systems, as-is
* Can include only labor
* Can include developing and end items
* Can span relatively brief periods of item, or
* Can span quite long periods of time.

Statement of work: services and service-related product

System requirement specification: end item description

Technology readiness level: it is a measurement scale of how mature (ready) as system segment is

* Applicable to individual system elements
* Also applicable to the system as whole

TRL1 : is least mature system

TRL9 : is most mature system

TRL1:

* An idea, based on known scientific research
* Expressed in analytical terms
* Documented in scientific white papers

TRL2:

* Rudimentary analytical studies
* Technology unproven/ does not exist
* Relies on analysis of scientific principle
* Idea validated with stakeholders
  + May invest in proof-of-concept development

TRL3:

* Proof of concept development
* Use of analysis and lab experiments
* Software logic, pseudo-code (UML model)

TRL4:

* Lab testing
* Validated concept against original idea
  + Without rigorous testing
  + Just to see if it works/ is feasible
* May be integrated with other elements

TRL5:

* Mature the design
  + Design-in the “ilities”
* Software alpha release
  + Focus on functionality “does it work?”

TRL6:

* Mature the prototype
* Entity exhibits full functionality of primary characteristics
* Software beta release
  + Focus on usability/ use satisfaction
* Sunny day scenario tests

TRL7:

* Demonstrated in controlled operational scenarios (user group sessions)
* System possesses primary features, functions, characteristics
* More rigorous environmental/ rainy day testing

TRL8:

* Baseline the design
* Build test articles, conduct operational testing

TRL9:

* Finalized product
* Fielded, in use by actual users
* Entity performs mission in real environment with real user(s)

Systems Engineer needs to consider forward integration into the production line and also to retrofit

Lead systems engineer/ chief engineer/ technical director

SEMP: Systems Engineering Management Plan

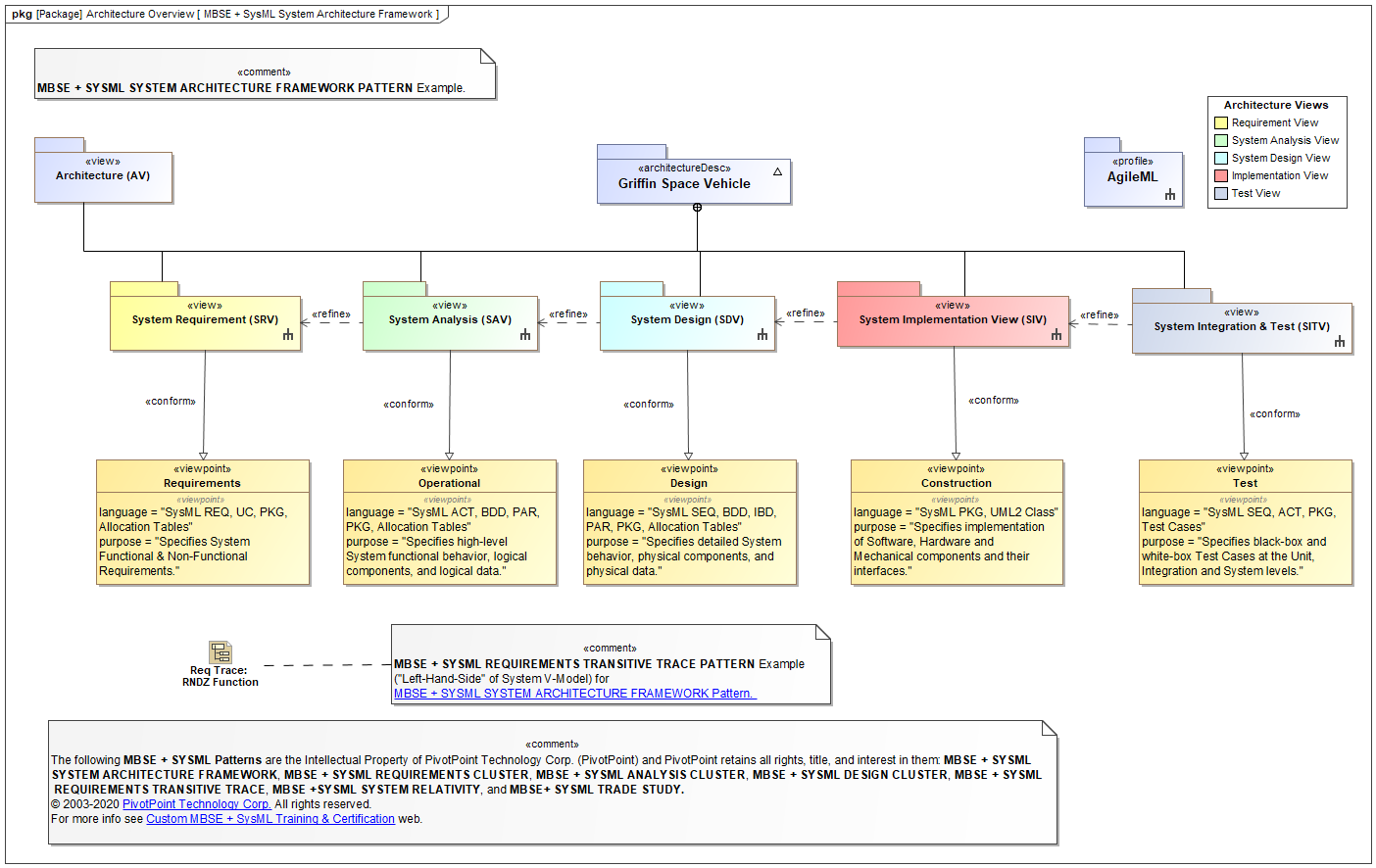
Interfaces:

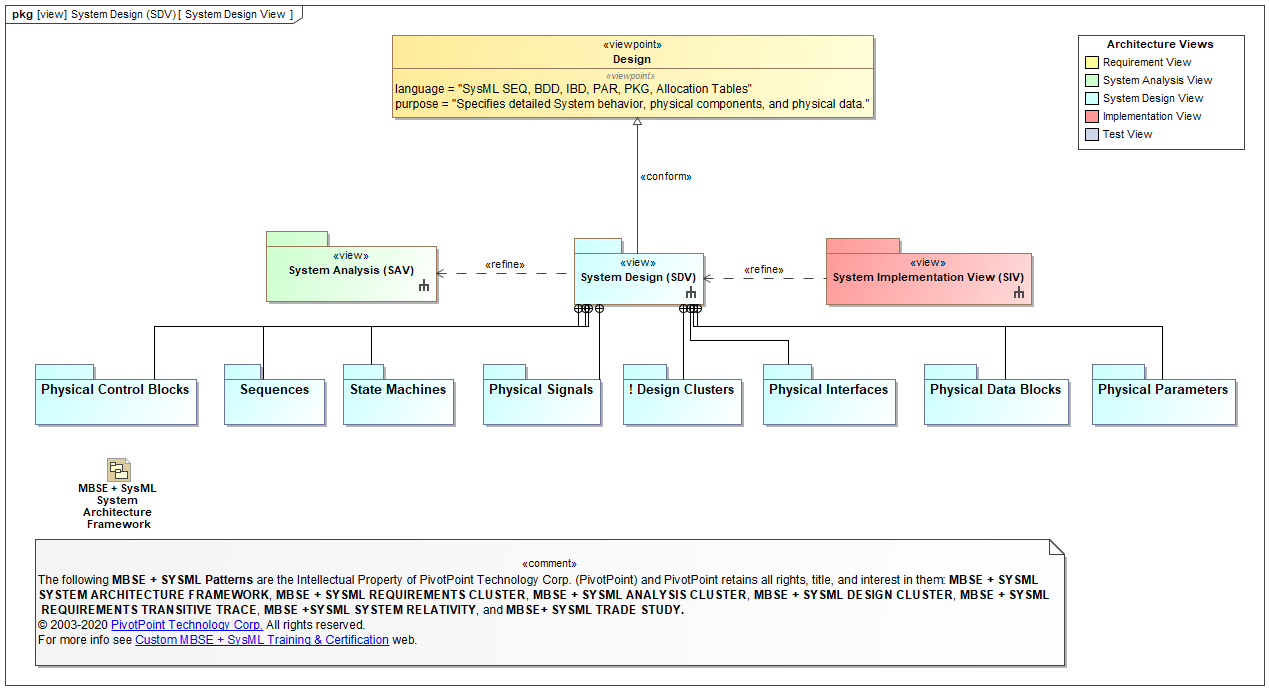
* Human-machine interfaces
* Environmental interfaces
* Segment interfaces
* Software module interfaces

Interfaces are showed in the segment level context diagram.

Behavioral diagram:

* Activity diagram, sequence diagrams and state machine diagram are the three options that sysml offers to specify the system behavior.
* The behavior diagram is intended to provide clarity for example about the internal process, business process, or the interaction of different systems.





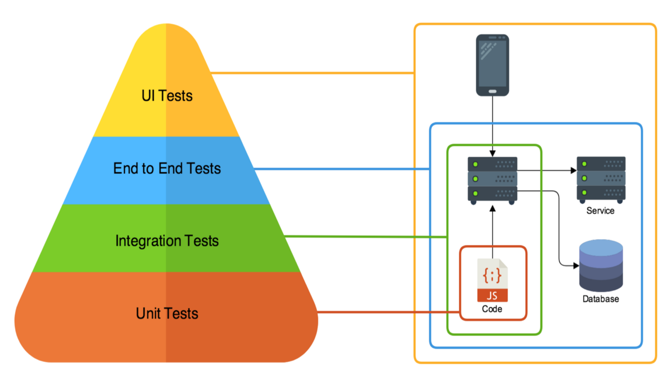
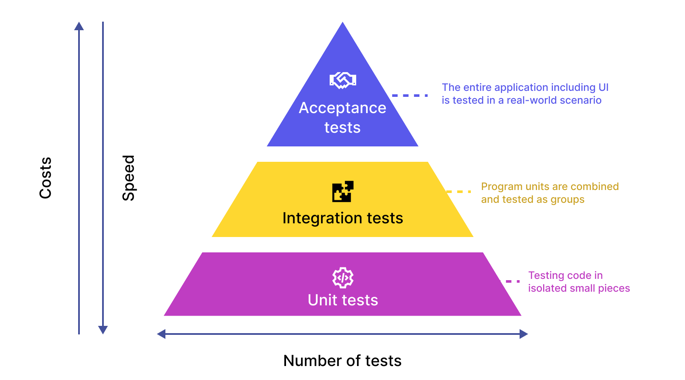
Loosely bound and tightly coupled functions: BAD example

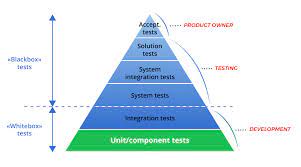
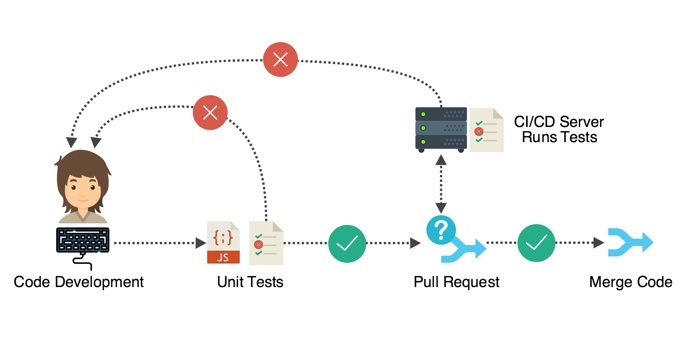
Tightly bound and loosely coupled functions: GOOD example

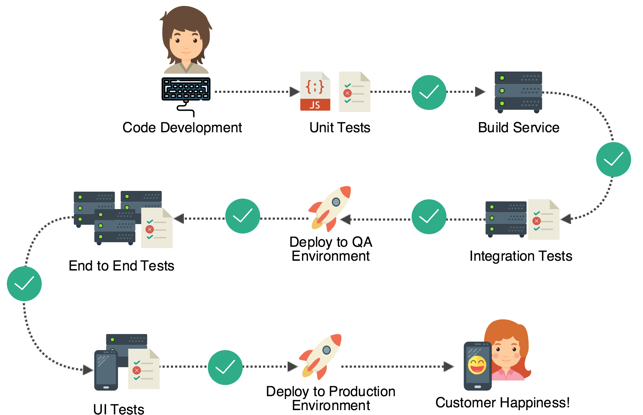
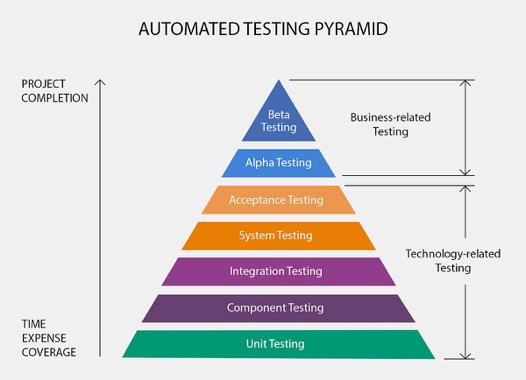
Test types in test plan:

* Segment level (unit)
* System integration
* Stress/ failure mode
* Validation (user) tests
* Regression (re-test)
* System-level (post-integration)

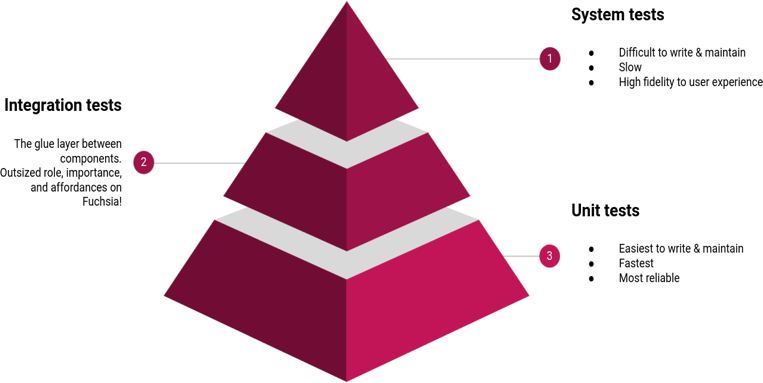
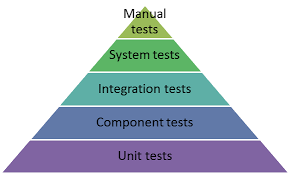
Unit test: it is a software development process to test smallest testable part of an application called units, are individually scrutinized for proper operation.

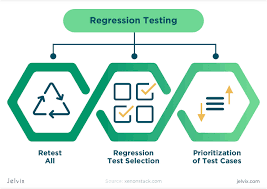
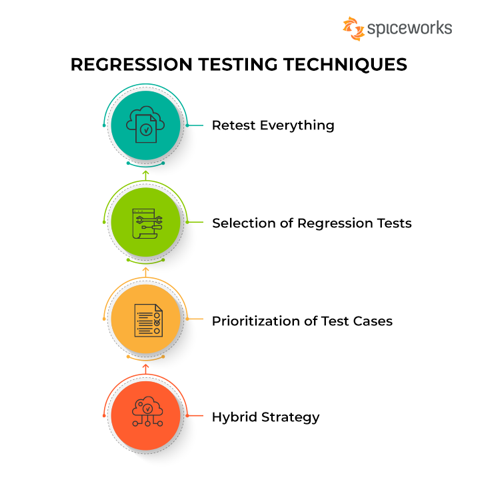
 

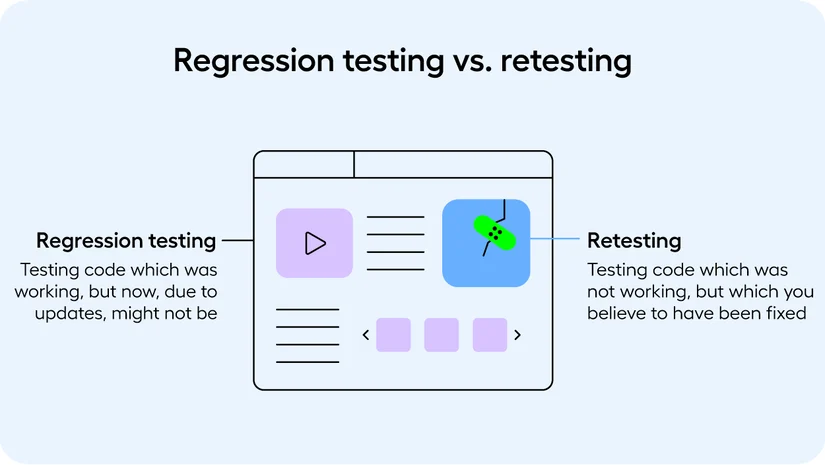
Integration tests: white-box testing and black-box testing

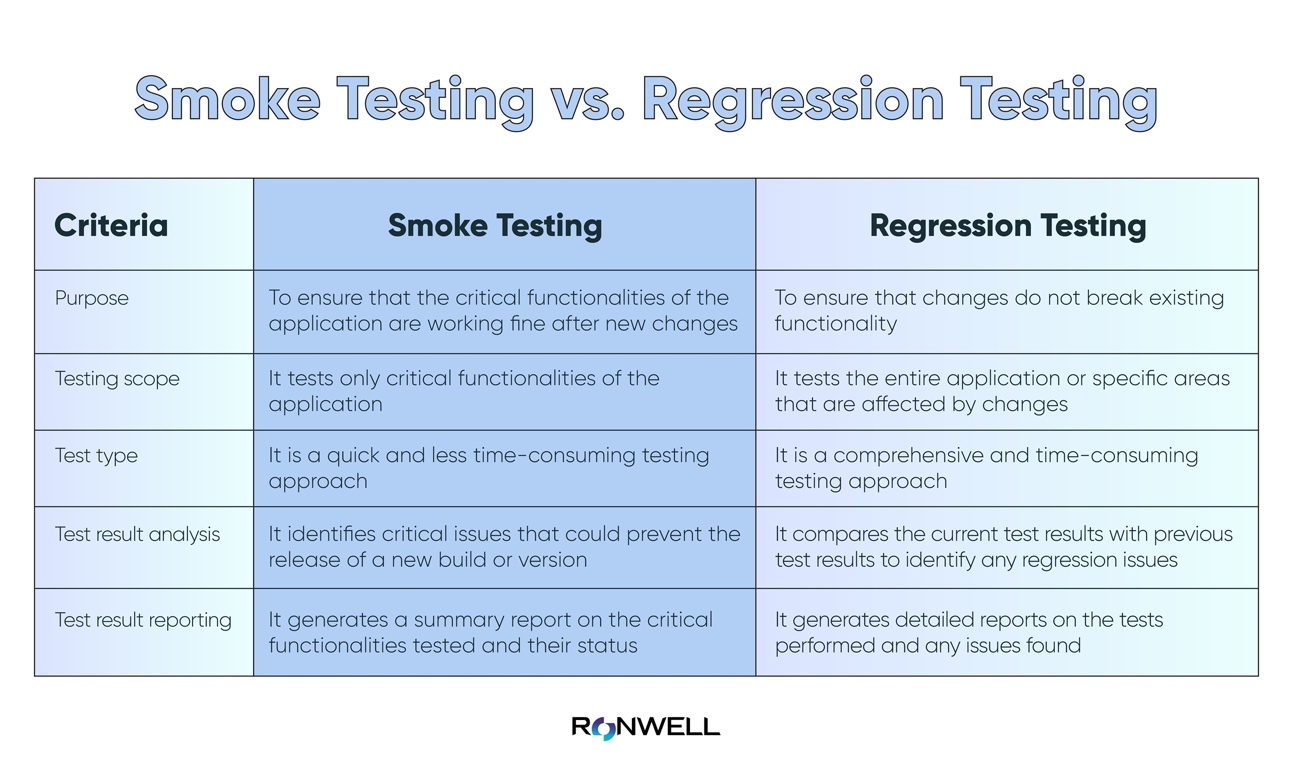
 

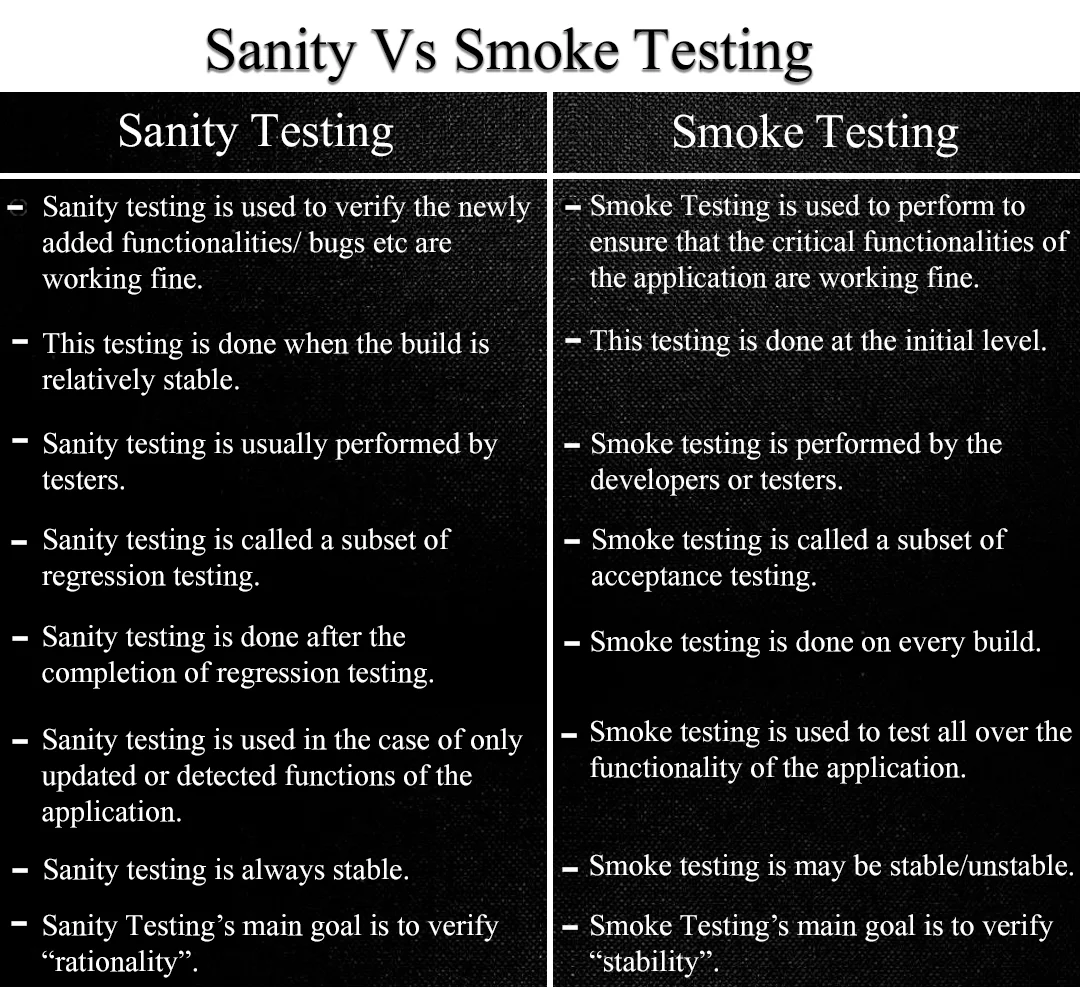
Regression test:

* It is a type of software testing conducted after a code update to ensure that the update introduced no new bugs.
* It is re-running functional and non-functional tests to ensure that previously developed and tested software still performs as expected after a change. If not, that would be called a regression.







System Analysis:

* All-encompassing (used throughout lifecycle)
* Perform modeling, simulation, math analysis, prototyping
* Reduce cost
* Reduce risk

Implementation (segment construction)

* Inputs
  + Life cycle concepts
  + System architecture description
  + System architecture rationale
  + System design description
  + System design rationale
  + Interface definition
  + Design traceability
  + System element description
* Plan for implementation
* Perform implementation
  + Convert segment/ design spec descriptions in HW/SW
  + Capture processes used in necessary
  + Refine designs down to part levels
  + Perform analysis for optimal configuration
  + Conduct segment level peer review and working groups
  + Begin integration and verification process (bottom-up)
  + Perform and support HW/SW configuration audits
  + Develop enabling / support elements for segments
  + Work with other team members to foster integration
* Manage result of implementation
* Output
  + Implementation strategy
  + Implementation enabling system requirements
  + Implementation constraints
  + System elements
  + System elements documents
  + Training material
  + Implementation traceability
  + Implementation reports
  + Implementation record

Integration:

* Inputs:
  + Life cycle concepts
  + Interface definition
  + System element description
  + System element
  + System element documentation
  + Implementation traceability
  + Accepted system or system element
* Prepare for integration
* Perform integration
* Manage results of integration
* Output:
  + Integration strategy
  + Integration enabling system requirement
  + Integration constraints
  + Integration procedures
  + Integrated system or system elements
  + Interface definition update identification
  + Integration report
  + Integration record

Project planning:

* It is perpetual/ on-going
* Helps identify risks
* Helps identify resources
* Co-ordination is necessary
* Establish the foundation of the project.
* Can be used as enabler
* IMP: integrated master plan

Project Management Plan:

* Risk management
* Configuration management
* Schedule management
* Cost management
* Human resource management
* Quality management
* Communication management
* Procurement management
* Stakeholder management
* Integration management
* Technical management

Project Planning Process:

Input:

* + Organization strategy plan
  + Source document
  + Supply response
  + Project portfolio
  + Life cycle models
  + Project direction
  + Strategy documents
  + Project tailoring strategy
  + Project lessons learned
  + Documentation tree
  + Quality management corrective actions
  + Qualified personnel
  + Quality assurance plan

Process:

* + Define the project
  + Plan the project and technical management
  + Activate the project

Output:

* + SEMP
  + Project constraints
  + Project infrastructure needs
  + Project human resources
  + Work breakdown structure (wbs)
  + Project schedule
  + Project budget
  + Acquisition need
  + Project planning record