Software Development Models and Testing

Global CyberSoft

By: Phu Nguyen

Course duration: 3h

Last update: Oct. 2014

www.globalcybersoft.com

A World of Difference

CMMI Level 4

TÜV SÜD ISO 9001

Microsoft GOLD CERTIFIED Partner

Outline

- Purposes
- Software development models and testing
- Test levels
- Test types

Purposes

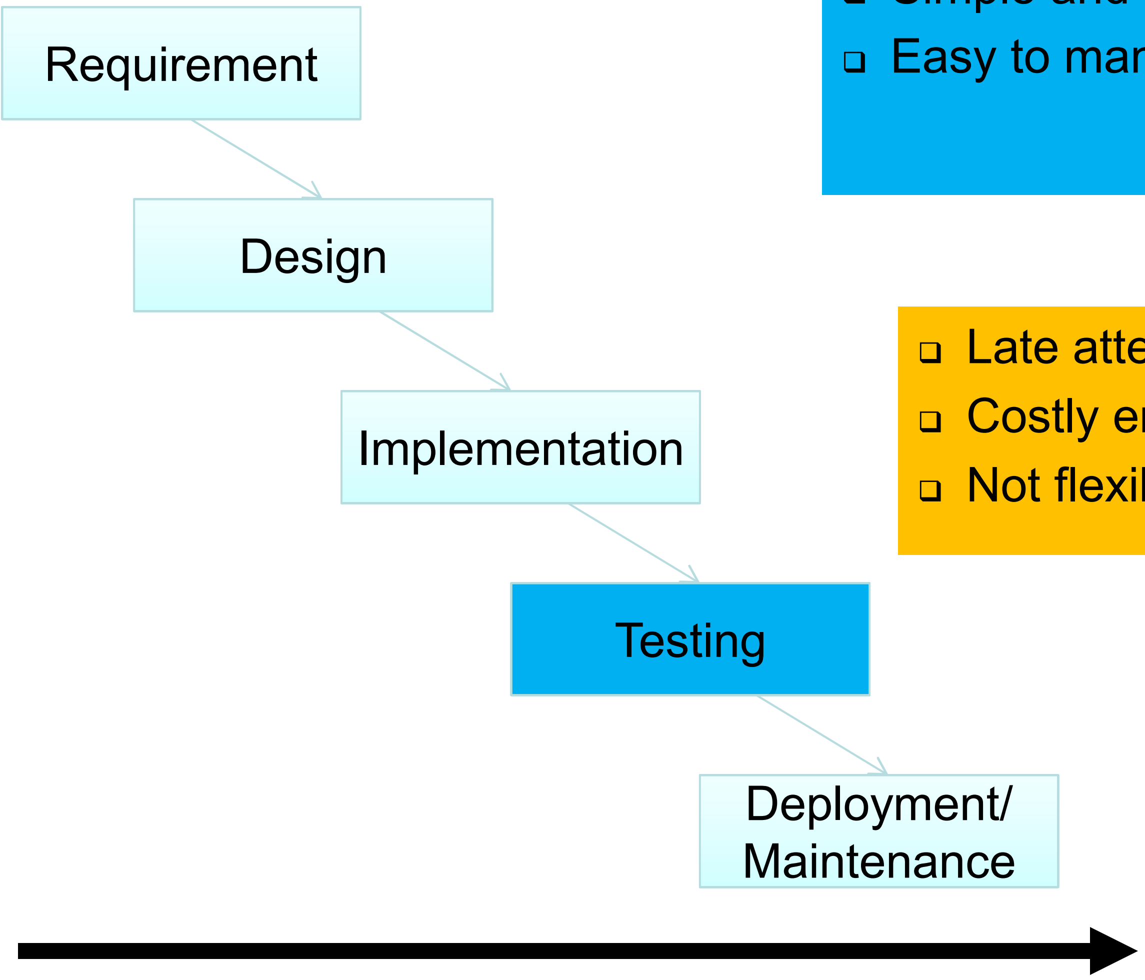
- The purpose of this presentation is to introduce the popular software development models as well as the testing methods and activities that are performed on each model.
- The presentation also introduces the testing levels and testing types that are needed to be performed on all models, helping the readers to have overall picture about software development models and their own testing approach.

SOFTWARE DEVELOPMENT MODELS & TESTING

Software Development Models

- ❑ Waterfall model
- ❑ V model
- ❑ Iterative model
- ❑ Spiral model
- ❑ Agile development methodologies (Scrum)

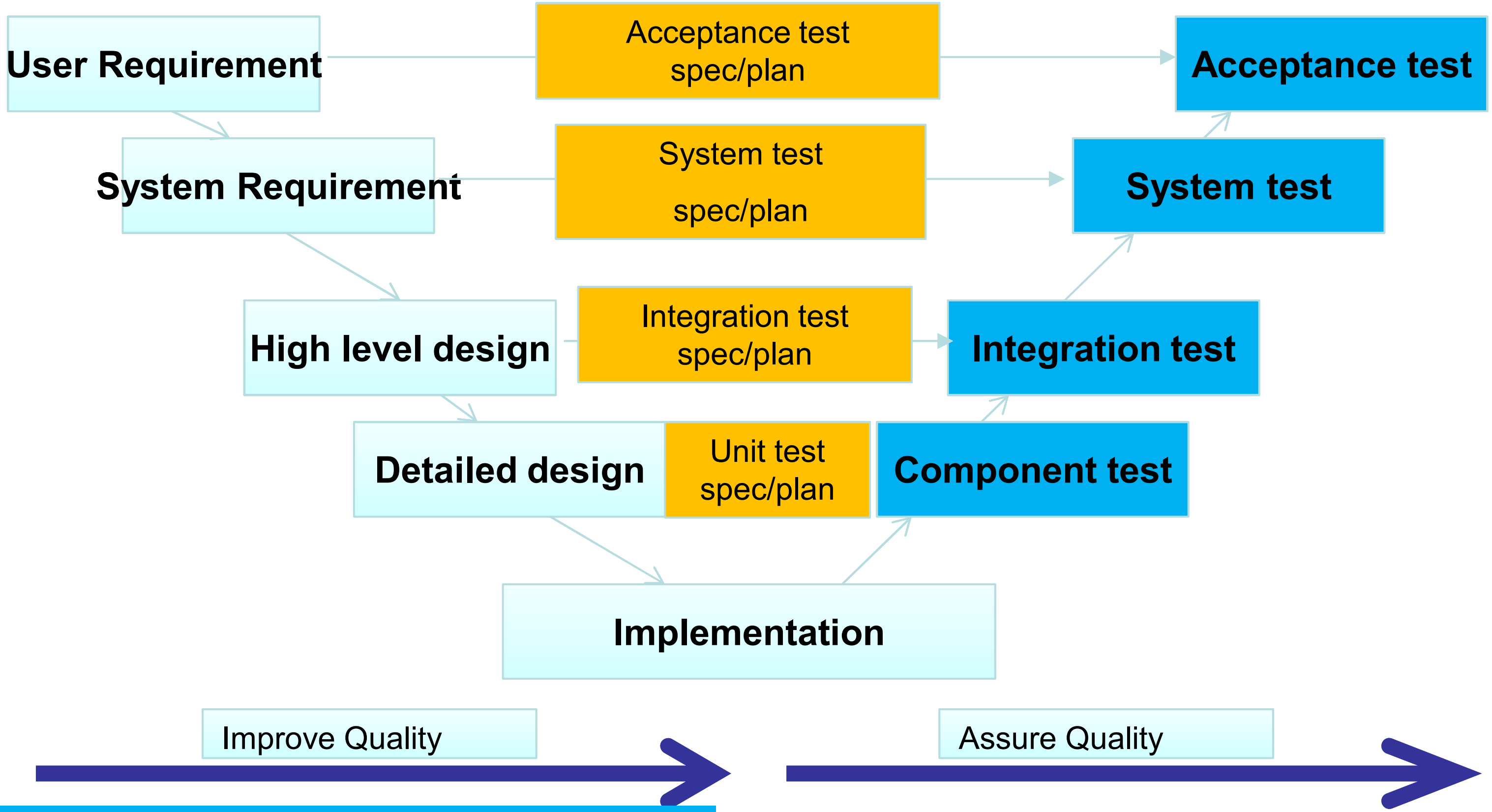
Waterfall Model



- Simple and easy
- Easy to manage

- Late attention to testing
- Costly errors
- Not flexible to change

V Model



User Requirement

Acceptance test spec/plan

Acceptance test

System Requirement

System test spec/plan

System test

High level design

Integration test spec/plan

Integration test

Detailed design

Unit test spec/plan

Component test

Implementation

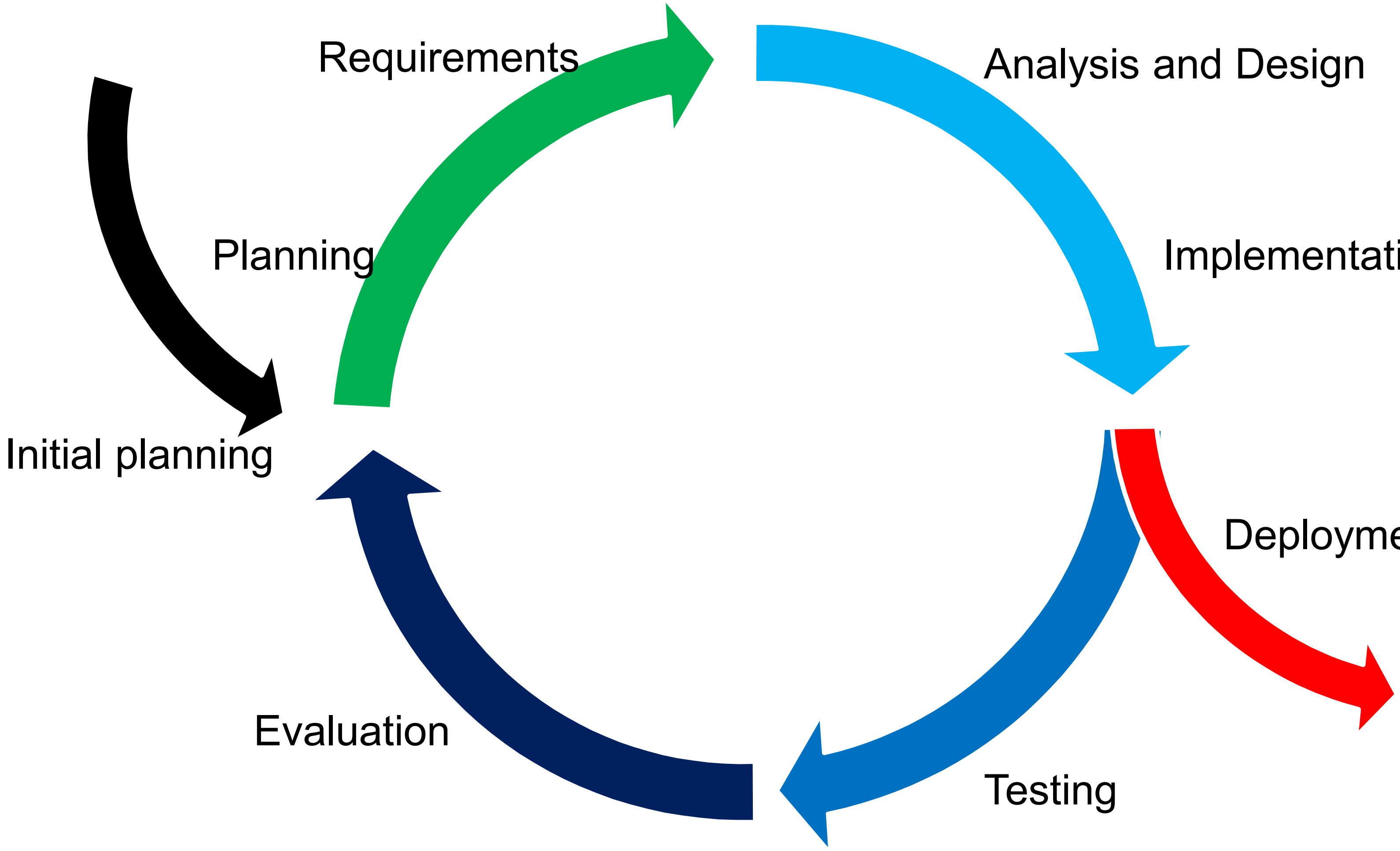
Improve Quality

Assure Quality

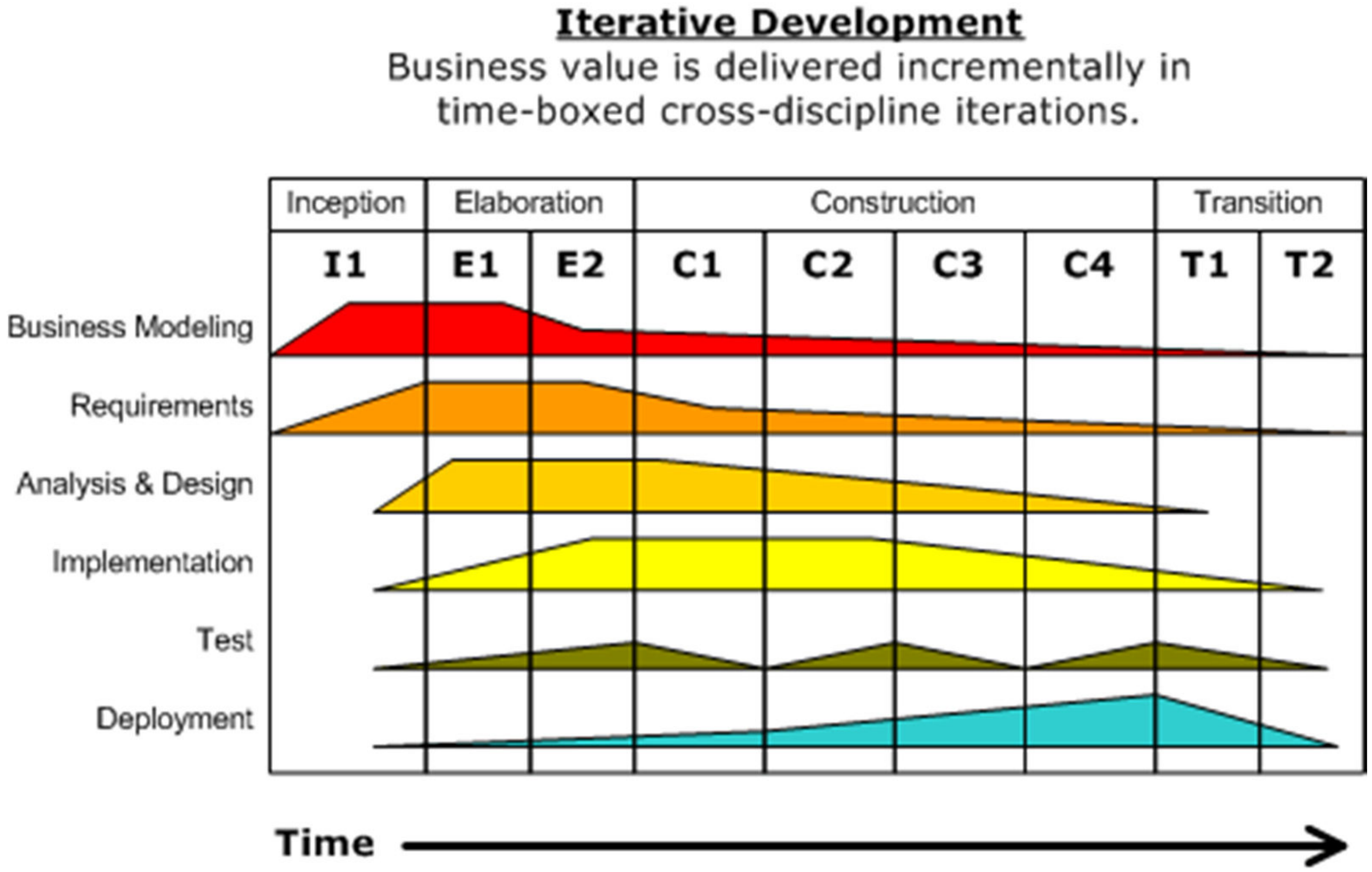
- Simple and easy
- Easy to manage
- Testing work done early
- Deliverables: many kinds of test spec
- Improve and assure quality through development

- Costly errors
- A Little bit flexibility to change

Iterative Model



Iterative Phases



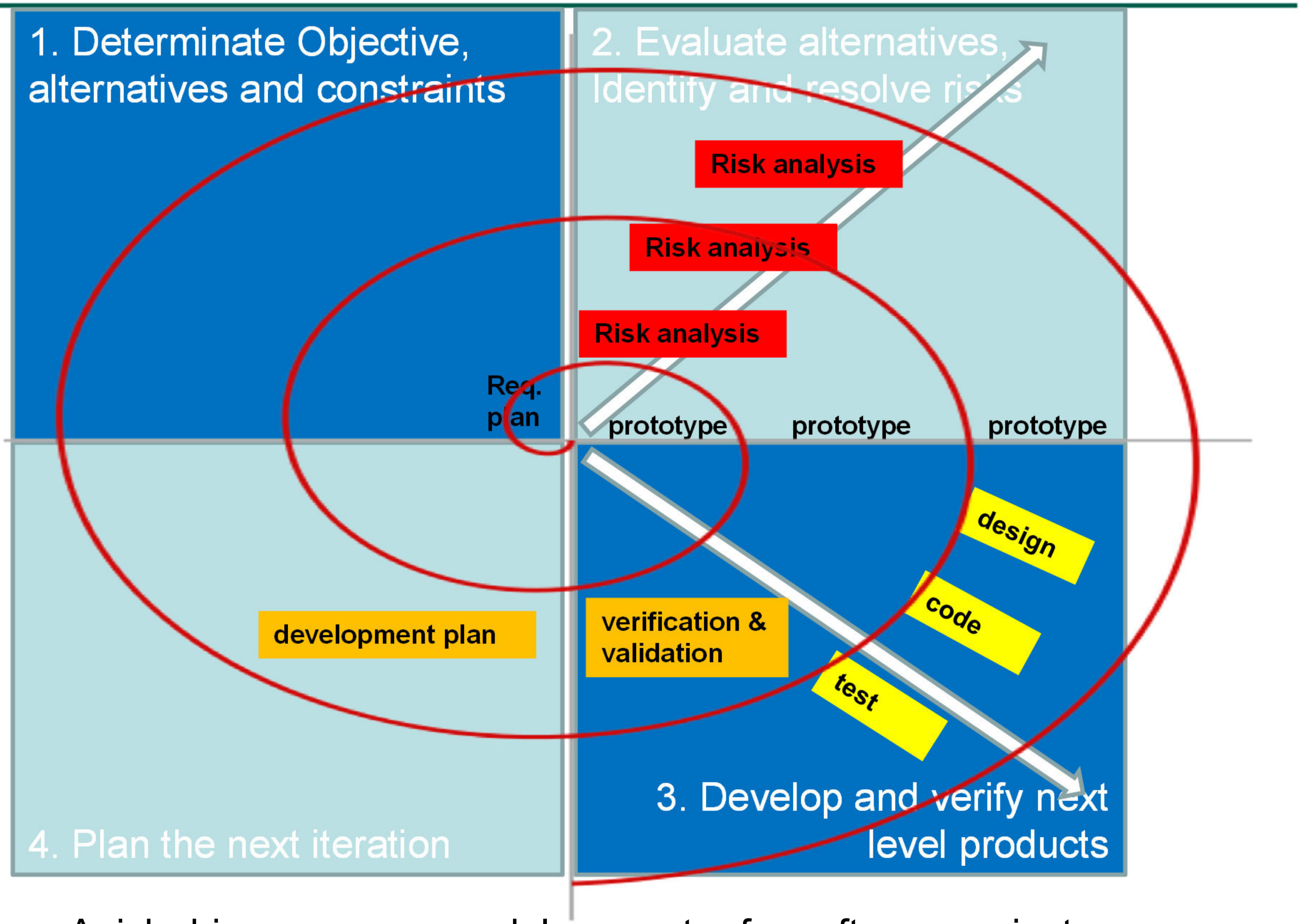
- Serial in the large
- Iterative in the small
- Delivering incremental releases over time

Iterative Model

- Generate software quickly and early during software life cycle
- More flexible
- Earlier and easier to test
- Easier to manage risks

- High cost for planning at the beginning
- Constant change may erode system architecture

Spiral Model


1. Determinate Objective, alternatives and constraints
2. Evaluate alternatives, Identify and resolve risks
Risk analysis
Risk analysis
Risk analysis
Req. plan
prototype
prototype
prototype
design
code
test
verification & validation
development plan
3. Develop and verify next level products
4. Plan the next iteration

A risk-driven process model generator for software projects

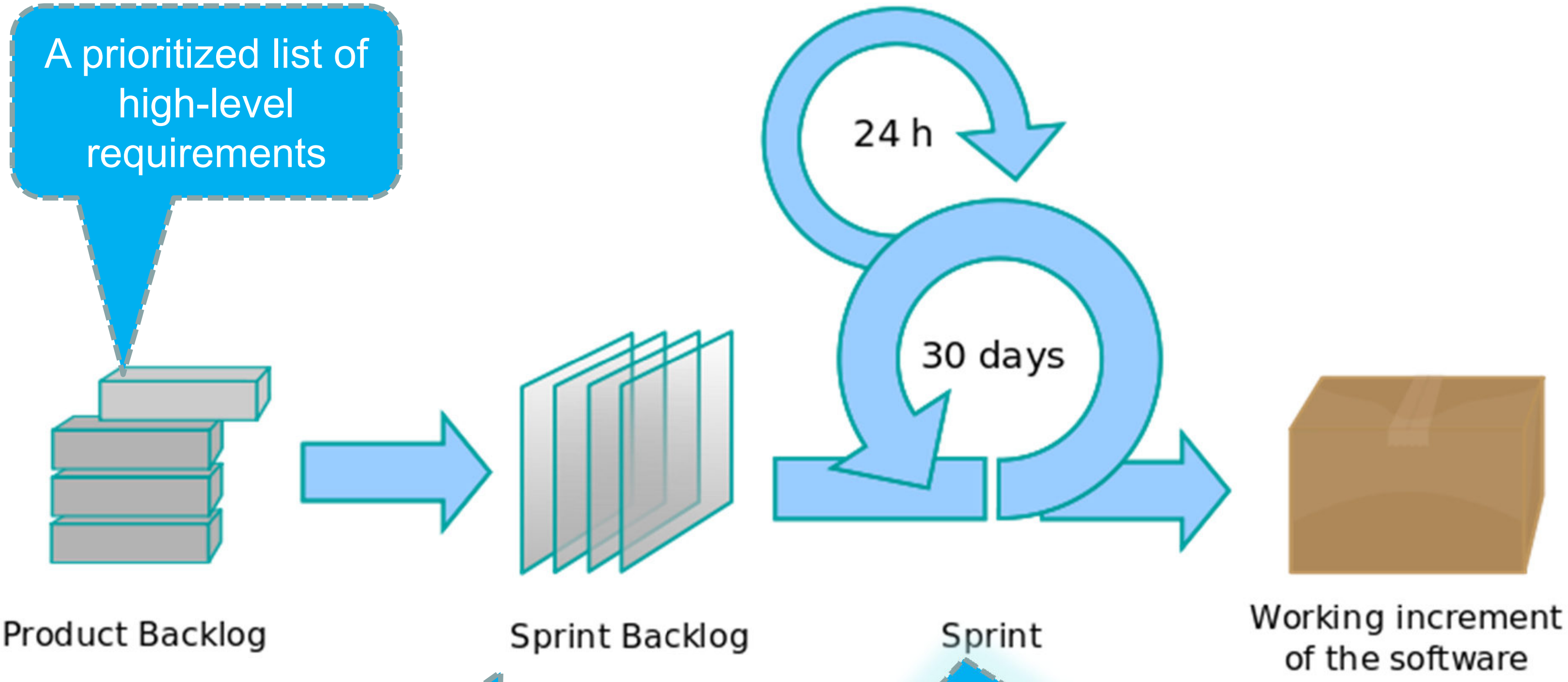
Spiral Model

- High amount of risk analysis
- Good for large and mission-critical projects
- Software is produce early
- Testing is performed early

- Could be costly model
- Risk analysis requires expertise
- Project's success is highly dependent on risk analysis phase

Scrum Model Overview (Agile Methodology)

An iterative and incremental agile software development framework for managing product development



A prioritized list of high-level requirements

A prioritized list of tasks to be completed during the sprint

A time period (typically 1–4 weeks) in which development occurs on a set of backlog items that the team has committed to. Also commonly referred to as a Time-box or iteration

Scrum Model Overview

- A Product Backlog of prioritized work to be done
- The entire product backlog items are split into a fixed set of items called Sprints
- A daily Scrum Meeting is organized where the team discusses three question.
  - What have you done since the last daily meeting?
  - What will you be doing until the next daily meeting?
  - What impediments, if any, are in your way?
- A Sprint Planning session where the Sprint Backlog items are split with the team members
- A Sprint Retrospective session where the team member would put forward the best practice they followed or introduce a process would need to implement for a successful Sprint.
- All the meetings will be facilitated by the Scrum Master.

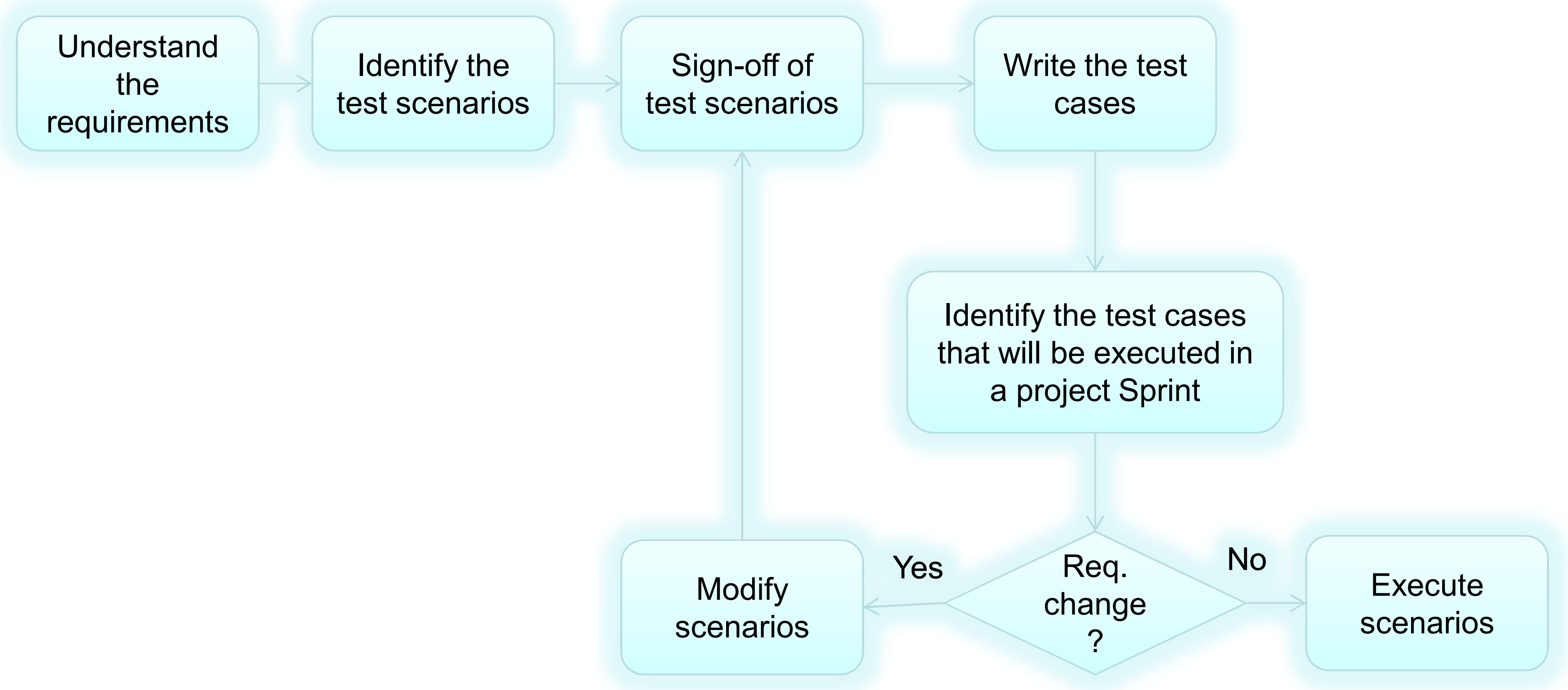
Scrum Roles and Responsibilities

- Scrum team
  - Product Owner, Scrum Master and Development Team
- Product owner
  - The person responsible for maintaining the Product Backlog by representing the interests of the stakeholders, and ensuring the value of the work the Development Team does
- Scrum master
  - The person responsible for the Scrum process, making sure it is used correctly and maximizing its benefits

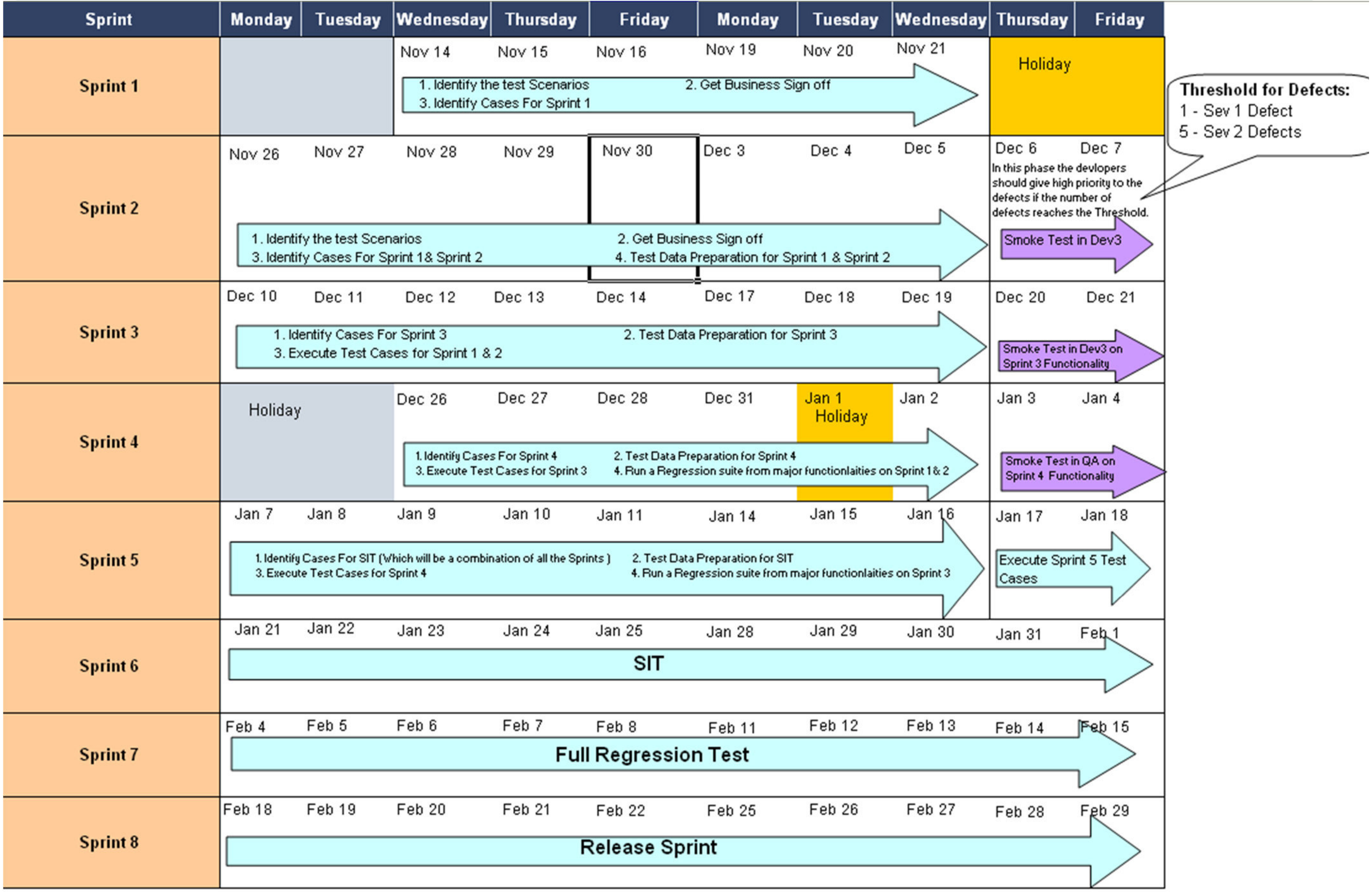
Scrum Testing Process

- Identify the Test Scenarios based on the Business Requirement document
- Obtain sign-off of the Test Scenario document from the respective Business Owners
- Test Cases will be written for Sprint by Sprint basics
- Execution will be done by delaying a Sprint (When Sprint 2 items are developed we will be executing Sprint 1 functionality)
- A change in a requirement or a defect will be added to the product backlog
- A constant regression bed is identified for all the completed Sprints
- Since delaying the testing by a Sprint may provide a gap between the development team and the testing team, we will have a pre deployment in each Sprint which helps us to address the major issues pro actively.

Scrum Testing Process Workflow


Understand the requirements
Identify the test scenarios
Sign-off of test scenarios
Write the test cases
Identify the test cases that will be executed in a project Sprint
Req. change ?
Yes
No
Modify scenarios
Execute scenarios

Scrum Testing Schedule (sample)



Sprint	Monday	Tuesday	Wednesday	Thursday	Friday	Monday	Tuesday	Wednesday	Thursday	Friday
Sprint 1			Nov 14	Nov 15	Nov 16	Nov 19	Nov 20	Nov 21	Holiday	
			1. Identify the test Scenarios 2. Get Business Sign off 3. Identify Cases For Sprint 1							
Sprint 2	Nov 26	Nov 27	Nov 28	Nov 29	Nov 30	Dec 3	Dec 4	Dec 5	Dec 6	Dec 7
									In this phase the devlopers should give high priority to the defects if the number of defects reaches the Threshold.	
	1. Identify the test Scenarios 2. Get Business Sign off 3. Identify Cases For Sprint 1& Sprint 2 4. Test Data Preparation for Sprint 1 & Sprint 2								Smoke Test in Dev3	
Sprint 3	Dec 10	Dec 11	Dec 12	Dec 13	Dec 14	Dec 17	Dec 18	Dec 19	Dec 20	Dec 21
	1. Identify Cases For Sprint 3 2. Test Data Preparation for Sprint 3 3. Execute Test Cases for Sprint 1 & 2								Smoke Test in Dev3 on Sprint 3 Functionality	
Sprint 4	Holiday		Dec 26	Dec 27	Dec 28	Dec 31	Jan 1 Holiday	Jan 2	Jan 3	Jan 4
			1. Identify Cases For Sprint 4 2. Test Data Preparation for Sprint 4 3. Execute Test Cases for Sprint 3 4. Run a Regression suite from major functionlaities on Sprint 1& 2						Smoke Test in QA on Sprint 4 Functionality	
Sprint 5	Jan 7	Jan 8	Jan 9	Jan 10	Jan 11	Jan 14	Jan 15	Jan 16	Jan 17	Jan 18
	1. Identify Cases For SIT (Which will be a combination of all the Sprints) 2. Test Data Preparation for SIT 3. Execute Test Cases for Sprint 4 4. Run a Regression suite from major functionlaities on Sprint 3								Execute Sprint 5 Test Cases	
Sprint 6	Jan 21	Jan 22	Jan 23	Jan 24	Jan 25	Jan 28	Jan 29	Jan 30	Jan 31	Feb 1
	SIT									
Sprint 7	Feb 4	Feb 5	Feb 6	Feb 7	Feb 8	Feb 11	Feb 12	Feb 13	Feb 14	Feb 15
	Full Regression Test									
Sprint 8	Feb 18	Feb 19	Feb 20	Feb 21	Feb 22	Feb 25	Feb 26	Feb 27	Feb 28	Feb 29
	Release Sprint									

Threshold for Defects:
1 - Sev 1 Defect
5 - Sev 2 Defects

Iterative testing activities

Phase	Activities
Inception	Initial test planning Review initial project management work products Review initial models
Elaboration	Validate the architecture Evolve your test model.
Construction	Test the software Evolve your test model
Transition	Validate the system Validate the documentation Finalize your test model.

Full Life Cycle Object-Oriented Testing Method

Requirement Testing	Analysis Testing	Architecture/ Design Testing	System Testing	User Testing
• Model review • Prototype walkthroughs • Prove it with code • Usage scenario testing	• Model review • Prototype walkthroughs • Prove it with code • Usage scenario testing	• Model review • Model walkthroughs • Prototype walkthroughs • Prove it with code • Code Testing	• Black-box testing • Boundary value testing • Class integration testing • Class testing • Code review • Coverage testing • Regression testing • Method testing • Path testing • White-box testing	• Alpha testing • Beta testing • Pilot testing • UAT

Regression Testing, Quality Assurance

TEST LEVELS

Test levels

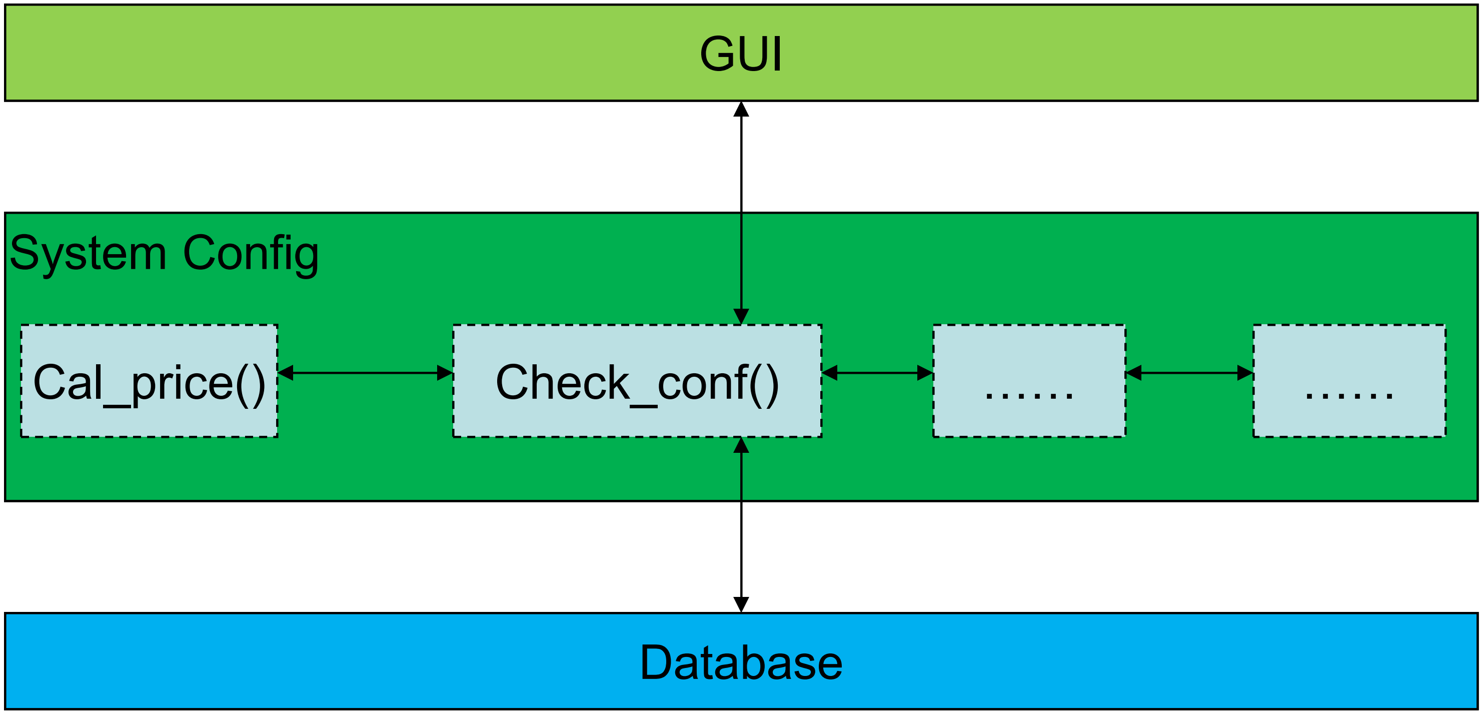
- Component testing
- Integration testing
- System testing
- Acceptance testing

Test level – Component testing

- The testing of individual software component
- Known as unit, module and program testing
- Component testing may including testing of functional and non-functional (ex: memory leaks)

Test level – Integration testing

- There are more than one level of integration testing
  - Component integration testing: test interactions between software component (done after component testing)
  - System integration testing: test the interactions between different system (may be done after system testing)



Test level – System testing

- The process of testing to verify that it meets specified requirements
- Most often the final test on behalf of development to verify system to be delivered meets the specification requirements
- Purpose to find as many defect as possible
- Investigate both functional and non-functional requirements of system
- Requires control test environment, version of system under test, test-ware and test data

Test level – Acceptance test

- Testing with respect to:
  - User needs
  - Requirement and business process conducted to determine whether or not a system satisfies the acceptance criteria
- The goal of acceptance testing is establish confidence in the system, part of system or specific non-functional characteristics.

TEST TYPES

Test types

- A group of test activities aimed at testing a component or system focused on specific test objective
- The following types of testing can be:
  - Functional testing
  - Non-functional testing

Test type – Functional testing

- Include all kind of tests which verify a system's input – output behavior
- Consider the specified behavior (Functional requirement)
- Often referred to as black-box testing
- Technical used for functional testing are often specification-based

Test type – Non-functional testing

- Non-functional requirements do not describe the function, but the attributes of functional behavior
- Non-functional testing (similar functional testing) is performed at all test levels
- Include kind of tests
  - Load test, performance test, volume test, stress test
  - Usability testing
  - Security testing
  - Maintainability testing

Test type – Non-functional testing (Cont.)

- **Performance test**: measuring of the processing speed and response time for particular user cases
- **Load test**: measuring of the system behavior for increasing system load (ex: No. of users that work simultaneously)
- **Volume test**: Observation of system behavior dependent on the amount of data
- **Stress test**: Observation of system behavior when it is overloaded

Test type – Non-functional testing (Cont.)

- **Reliability test**: measuring the system's respond to operating error, hardware failure, exception handling and recovery
- **Usability test**: Examination of the ease of learning, ease and efficiency of the operating system, understandability,..
- **Maintainability test**:
  - Check the understanding of the system documentation (Whether it is up to date)
  - Checking if the system has a modular structure,..

Maintenance testing

- ❑ Modifications for planned Enhancement Changes
- ❑ Maintenance testing for Migration
- ❑ Maintenance testing for the retirement of a System

References

- *Chapter 2: Testing throughout Lifecycle* by Louise Hasson
- *A Manager's Introduction to The Rational Unified Process (RUP) by Scott W. Ambler*
- *ISTQB Foundation Level Syllabus (version 2007)*
- *http://www.ambysoft.com/unifiedprocess/agileUP.html*
- *http://www.ambysoft.com/essays/floot.html*

Thank you

Thank you very much!