OSHADHI MUNASINGHE
OKMUNASI@TEC.RJT.AC.LK

# SOFTWARE VERIFICATION AND VALIDATION

ICT 3312

# COURSE DETAILS

**Course Code** : ICT 3312

**Course Name** : Software Verification and Validation

**Number of credits** : 3 credits

**Assignment – 40%**

**Final Exam – 60%**

**Practical**

# LEARNING OUTCOMES

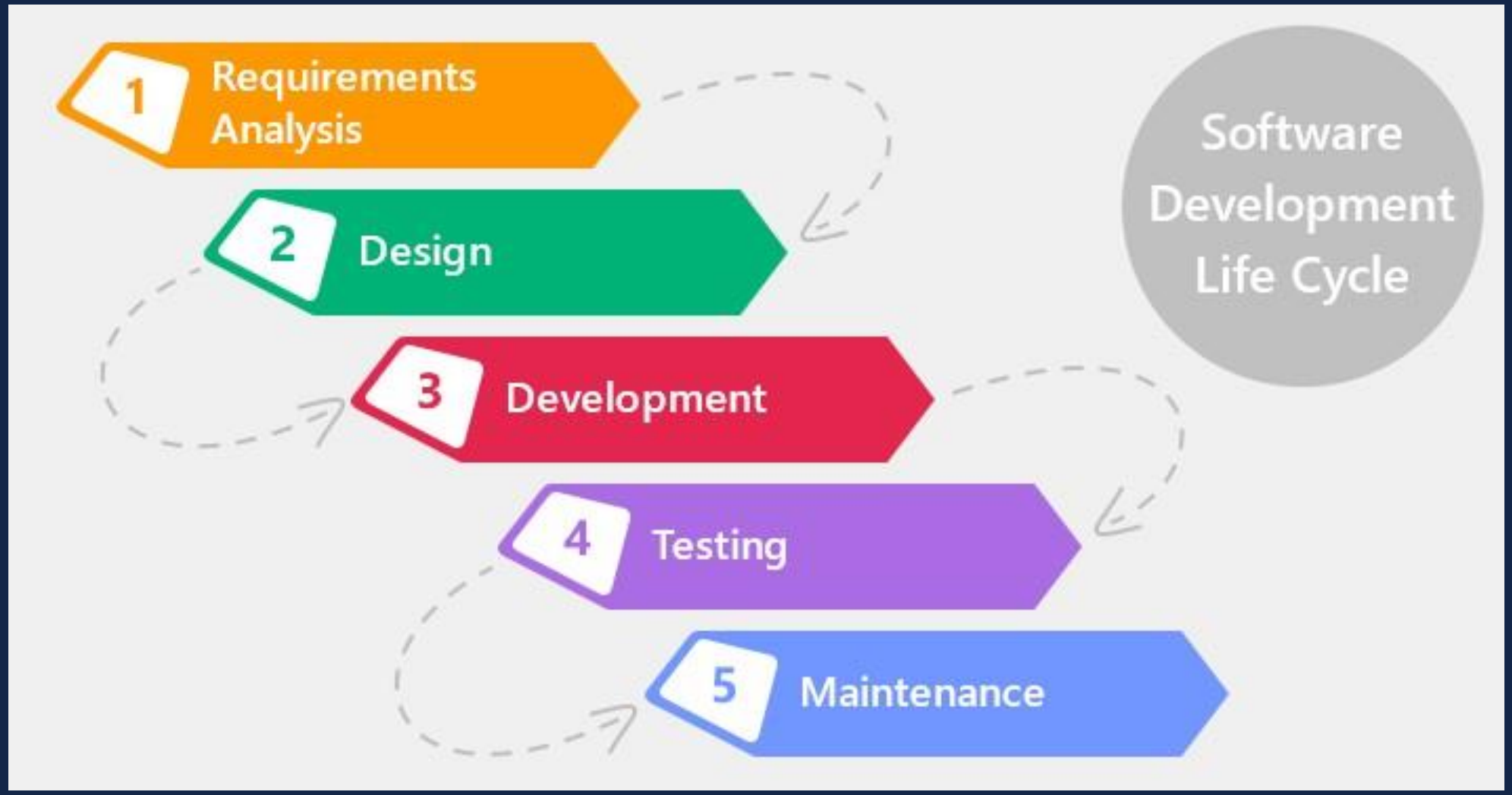
- Explain the meaning and importance of quality in relation to software systems.
- Identify the processes and techniques which make high-quality systems and achievable goal.
- Describe and classify quality assurance practices in various stages of software projects (namely in pre-project, design, development, deployment and maintenance stages)
- Apply basic software testing strategies to a given scenario
- Identify the Software quality infrastructure and managerial aspects
- Describe the standards applicable in software quality
- Explain current trends in SQA
- Perform automated testing Front Ends, Mobile Applications and Load testing

# OVERVIEW (LECTURES)

- What is software quality?
- Software quality factors
- The components of the software quality assurance system –an overview
- The SQA system /architecture
- Pre-project components
- Software project life cycle components
- Infrastructure components for error prevention and improvement
- Management SQA components
- SQA standards, system certification and assessment components
- Organizing for SQA –the human components

# INTRODUCTION

## CHAPTER 01


1
Requirements Analysis
2
Design
3
Development
4
Testing
5
Maintenance
Software Development Life Cycle

# WHAT IS ‘SOFTWARE QUALITY ASSURANCE’?

- **Software QA involves the entire software development PROCESS –**
  - **monitoring and improving the process,**
  - **making sure that any agreed-upon processes,**
  - **standards and procedures are followed,**
  - **and ensuring that problems are found and dealt with.**
- **It is oriented to 'prevention'. (*)**

**Quality assurance** is a system of activities designed to ensure production that meets pre-established requirements and standards.

**Software quality** is defined as the quality that ensures customer satisfaction by offering all the customer deliverables on performance, standards and ease of operations.

# WHAT IS 'SOFTWARE TESTING'?

- **Testing involves operation of a system or application under controlled conditions and evaluating the results.**

**(e.g., 'if the user is in interface A of the application while using hardware B, and does C, then D should happen')**

***Software testing** is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free.*

- It is oriented to **'detection'. (*)**
- "Test to break" attitude **(*)**

# WHY DOES SOFTWARE HAVE BUGS?

1. **Miscommunication or no communication**
   - as to specifics of what an application should or shouldn't do (the application's requirements)
2. **The exponential growth in software/system complexity**
   - the complexity of current software applications can be difficult to comprehend for anyone without experience in modern- day software development.
     - E.g.: Multi-tier distributed applications utilizing multiple local and remote web services, use of cloud infrastructure, data communications, enormous/distributed data stores, security complexities, etc.
3. **Programming errors**
   - programmers, like anyone else, can make mistakes.

# WHY DOES SOFTWARE HAVE BUGS? CONTINUED…

4. **Dependencies among code modules, services, systems, other projects, etc. may not be well understood, and may cause unexpected problems.**

5. **Changing requirements**

   - The end-user may not understand the effects of changes, or may understand and request them anyway – redesign, rescheduling of engineers, effects on other projects, work already completed that may have to be redone or thrown out, hardware requirements that may be affected.

6. **Time pressures:**

   - scheduling of software projects is difficult at best, often requiring a lot of guesswork. When deadlines approaches and the due to the pressure, mistakes can be made.

# WHY DOES SOFTWARE HAVE BUGS? CONTINUED…

7. **Poorly designed/documented code**

   - It's tough to maintain and modify code that is badly written or poorly documented; the result is bugs. In many organizations management provides no incentive for programmers to document their code or write clear, understandable, maintainable code. In fact, it's usually the opposite: they get points mostly for quickly turning out code, and there's job security if nobody else can understand it ('if it was hard to write, it should be hard to read').

8. **Software development tools**

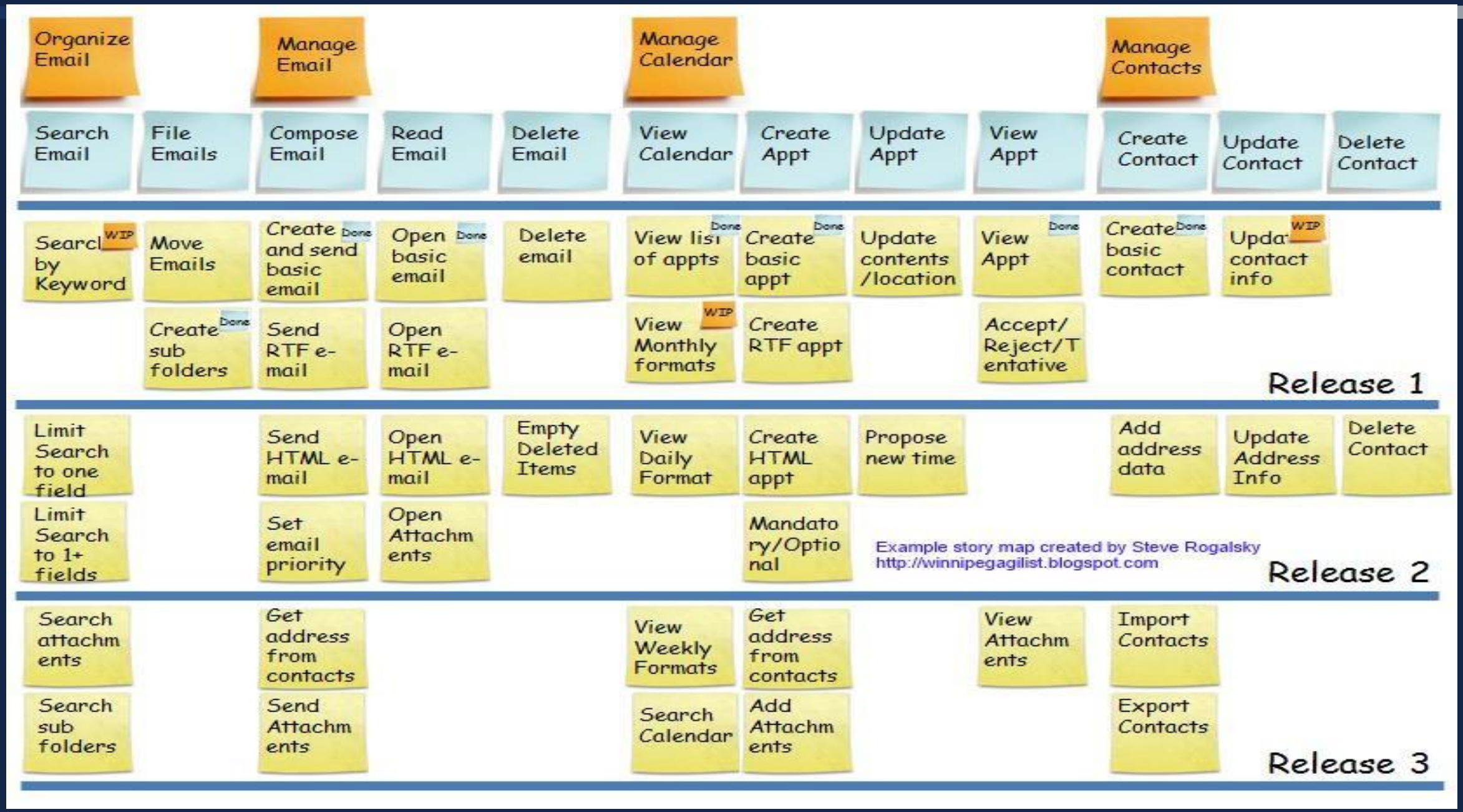
   - Visual tools, class libraries, compilers, scripting tools, etc. often introduce their own bugs or are poorly documented, resulting in added bugs.

# WHAT ARE THE COMMON PROBLEMS IN THE SOFTWARE DEVELOPMENT PROCESS? & SOLUTIONS ?

- **Q1. Poor Requirements**
  - if requirements are unclear, incomplete, too general, and not testable, there may be problems.

**A1. Solid requirements/user stories**

- clear, complete, appropriately detailed, cohesive, attainable, testable specifications that are agreed to by all players.


Organize Email
Manage Email
Manage Calendar
Manage Contacts
Search Email
File Emails
Compose Email
Read Email
Delete Email
View Calendar
Create Appt
Update Appt
View Appt
Create Contact
Update Contact
Delete Contact
Search by Keyword
WIP
Move Emails
Create and send basic email
Done
Open basic email
Done
Delete email
View list of appts
Done
Create basic appt
Done
Update contents /location
View Appt
Done
Create basic contact
Done
Update contact info
WIP
Create sub folders
Done
Send RTF e-mail
Open RTF e-mail
View Monthly formats
WIP
Create RTF appt
Accept/ Reject/T entative
Release 1
Limit Search to one field
Send HTML e-mail
Open HTML e-mail
Empty Deleted Items
View Daily Format
Create HTML appt
Propose new time
Add address data
Update Address Info
Delete Contact
Limit Search to 1+ fields
Set email priority
Open Attachments
Mandatory/Optional
Example story map created by Steve Rogalsky
http://winnipegagilist.blogspot.com
Release 2
Search attachments
Get address from contacts
View Weekly Formats
Get address from contacts
View Attachments
Import Contacts
Search sub folders
Send Attachments
Search Calendar
Add Attachments
Export Contacts
Release 3

# WHAT ARE THE COMMON PROBLEMS IN THE SOFTWARE DEVELOPMENT PROCESS? & SOLUTIONS ?

**Continued…**

- **Q2.Unrealistic Schedule**
  - if too much work is crammed in too little time problems are inevitable.

  **A2. Realistic schedules** (*)
    - allow adequate time for planning, design, testing, bug fixing, re- testing, changes, and documentation
    - personnel should be able to complete the project without burning out,
    - and be able to work at a sustainable pace.

# WHAT ARE THE COMMON PROBLEMS IN THE SOFTWARE DEVELOPMENT PROCESS? & SOLUTIONS ?

- **Q3. Inadequate Testing** **Continued…**
  - No one will know whether or not the software is any good until customers complain or systems crash.

**A3. Adequate testing**

- start testing early on,
- re-test after fixes or changes,
- plan for adequate time for testing and bug-fixing.
- ‘Early’ testing could include Static code analysis, test-first
- development, unit testing by developers, built-in testing and diagnostic capabilities, etc.
- Automated testing can contribute significantly if effectively designed and implemented as part of an overall testing strategy.

# WHAT ARE THE COMMON PROBLEMS IN THE SOFTWARE DEVELOPMENT PROCESS? & SOLUTIONS ?

- **Q4. Misunderstandings about dependencies**

**Continued…**

**A4. stick to initial requirements where feasible**

- be prepared to defend against excessive changes and additions once development has begun,
- and be prepared to explain consequences.
- If changes are necessary, they should be adequately reflected in related schedule changes.
- If possible, work closely with customers/end-users to manage expectations.
- In agile environments, requirements may change often, requiring that true agile processes be in place and followed.

# WHAT ARE THE COMMON PROBLEMS IN THE SOFTWARE DEVELOPMENT PROCESS? & SOLUTIONS ?

**Continued…**

- **Q5. Miscommunication**
  - if developers don't know what's needed or customer's have erroneous expectations, problems can be expected.

**A5. communication**

- Require walkthroughs /inspections/reviews when appropriate;
- Make extensive use of group communication tools - groupware, wiki's,
- bug-tracking tools, change management tools, audio/video conferencing, etc.;

# WHAT ARE THE COMMON PROBLEMS IN THE SOFTWARE DEVELOPMENT PROCESS? & SOLUTIONS ?

**Continued…**

- Ensure that information/documentation/user stories are available, up-to- date, and appropriately detailed;
- promote teamwork and cooperation;
- use prototypes, frequent deliveries, and/or continuous communication
- with end-users if possible to clarify expectations.
- In effective agile environments most of these should be taking place

# The SQA system