BINUS UNIVERSITY

People
Innovation
Excellence

Course : COMP6100/Software Engineering

Effective Period : Desember 2017

Software Testing and Software Configuration Management

Session 23 (Tutorial)

Acknowledgement

These slides have been adapted from Pressman, R.S. (2015). *Software Engineering : A Practioner's Approach. 8th ed*. McGraw-Hill Companies.Inc, Americas, New York. ISBN : 978 1 259 253157. Chapter 23, 24, 25, 26 and 27

Learning Objectives

LO 4 : Analyze the software project management and the proposed potential business project

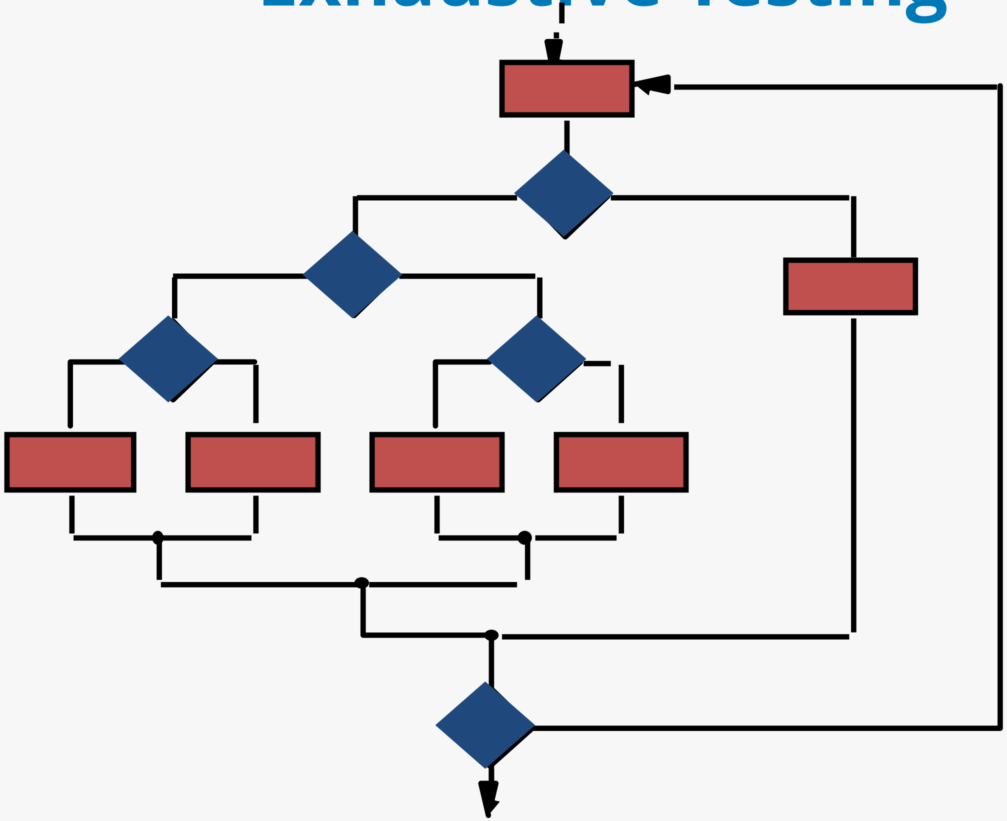
People
Innovation
Excellence

Contents

- Software Testing
- White-Box Testing
- Black-Box Testing
- Testing Concepts for Web and Mobile Apps
- The Software Configuration
- SCM Repository
- The SCM Process
- Configuration Management for Web and Mobile Apps

Software Testing

Exhaustive Testing

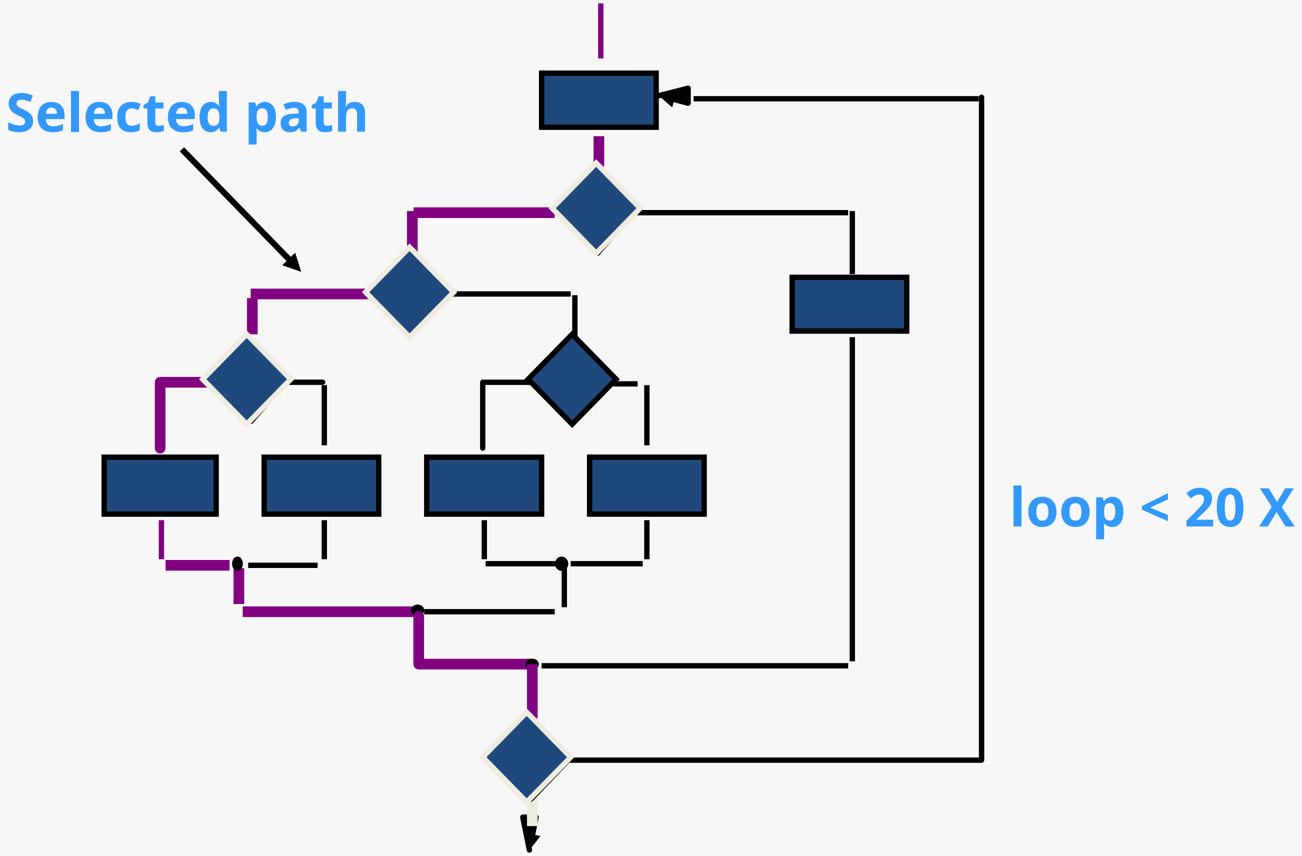


loop < 20 X

There are 10^{14} possible paths! If we execute one test per millisecond, it would take 3,170 years to test this program!!

Software Testing

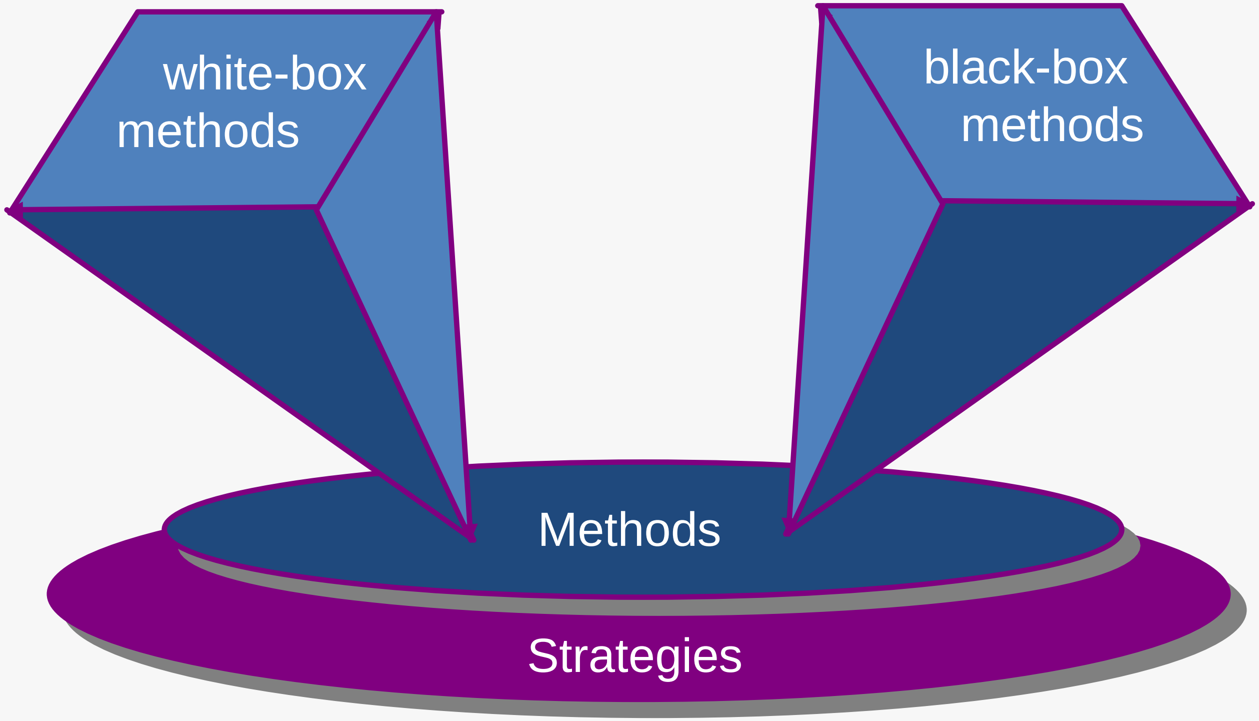
Selective Testing



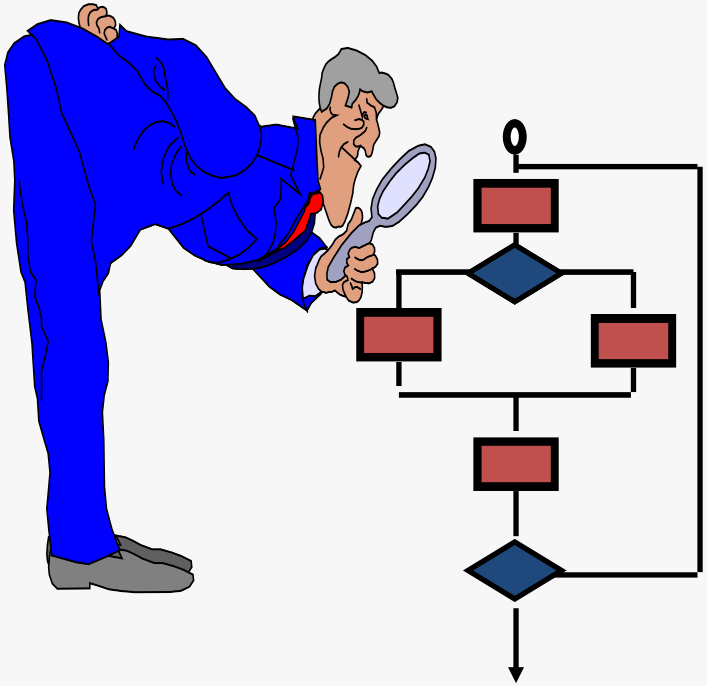
People
Innovation
Excellence

People
Innovation
Excellence

Software Testing



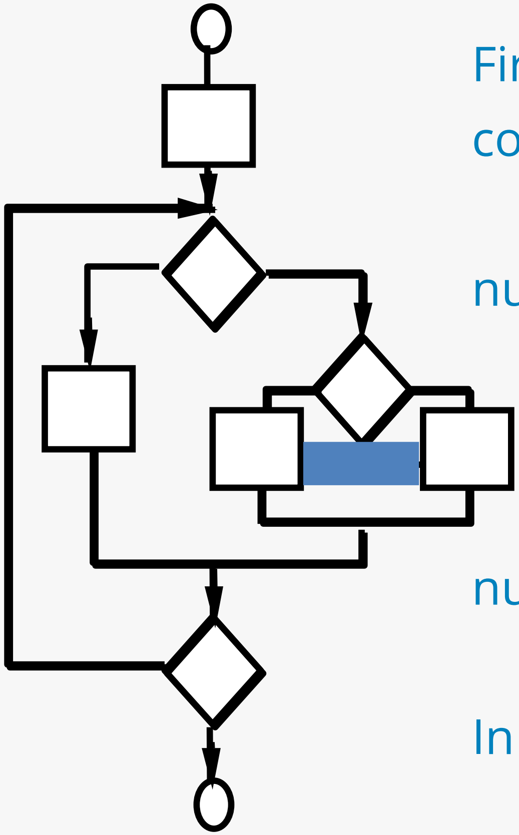
White-Box Testing



... our goal is to ensure that all statements and conditions have been executed at least once ...

White-Box Testing

Basis Path Testing



First, we compute the cyclomatic complexity:

number of simple decisions + 1

or

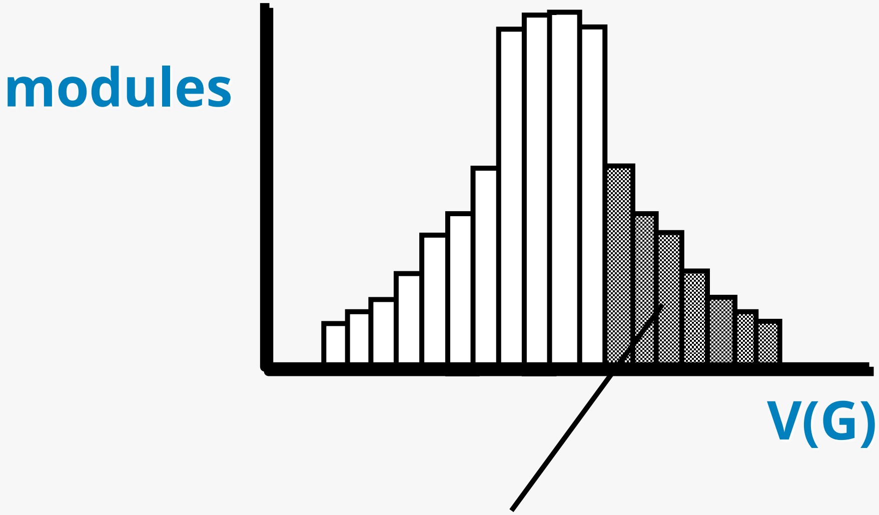
number of enclosed areas + 1

In this case, V(G) = 4

White-Box Testing

Cyclomatic Complexity

A number of industry studies have indicated that the higher V(G), the higher the probability or errors.

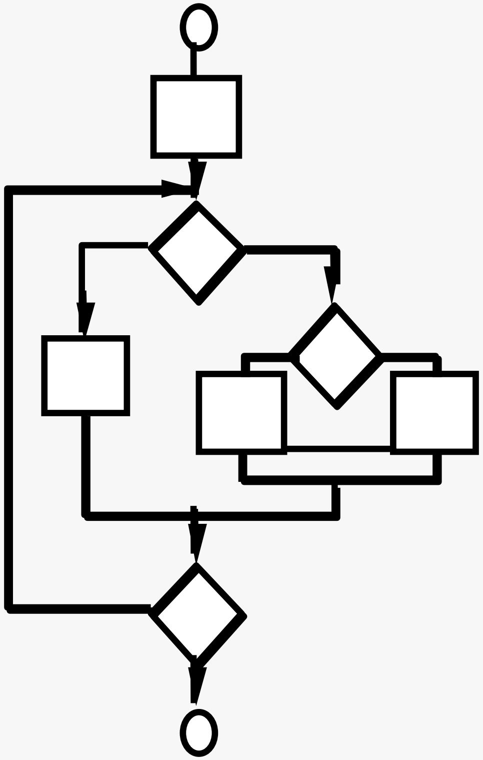


modules in this range are more error prone

People
Innovation
Excellence

White-Box Testing

Basis Path Testing Notes



- ❑ you don't need a flow chart, but the picture will help when you trace program paths
- ❑ count each simple logical test, compound tests count as 2 or more
- ❑ basis path testing should be applied to critical modules

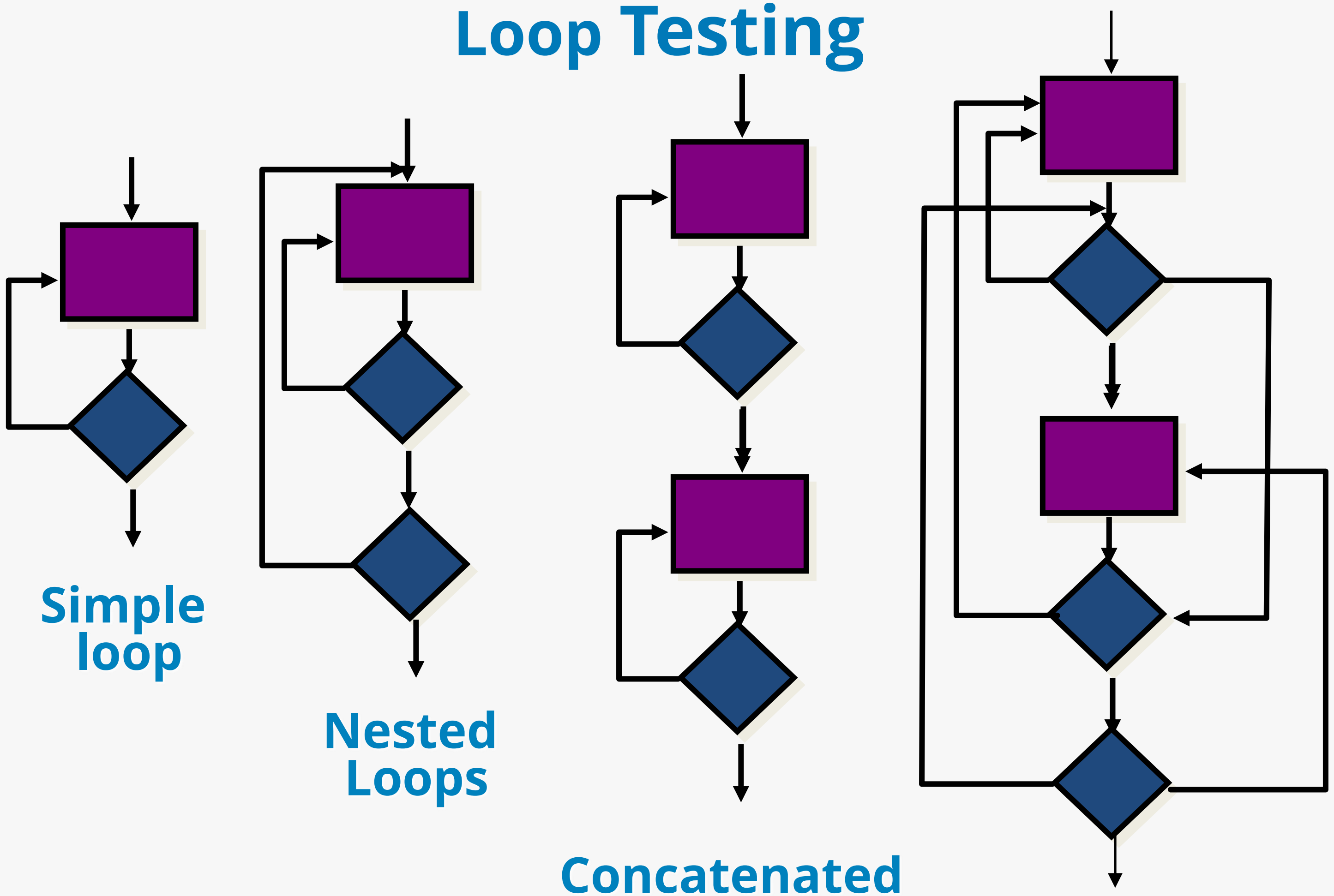
White-Box Testing

Graph Matrices

- A graph matrix is a square matrix whose size (i.e., number of rows and columns) is equal to the number of nodes on a flow graph
- Each row and column corresponds to an identified node, and matrix entries correspond to connections (an edge) between nodes.
- By adding a *link weight* to each matrix entry, the graph matrix can become a powerful tool for evaluating program control structure during testing

White-Box Testing

Loop Testing



Simple loop

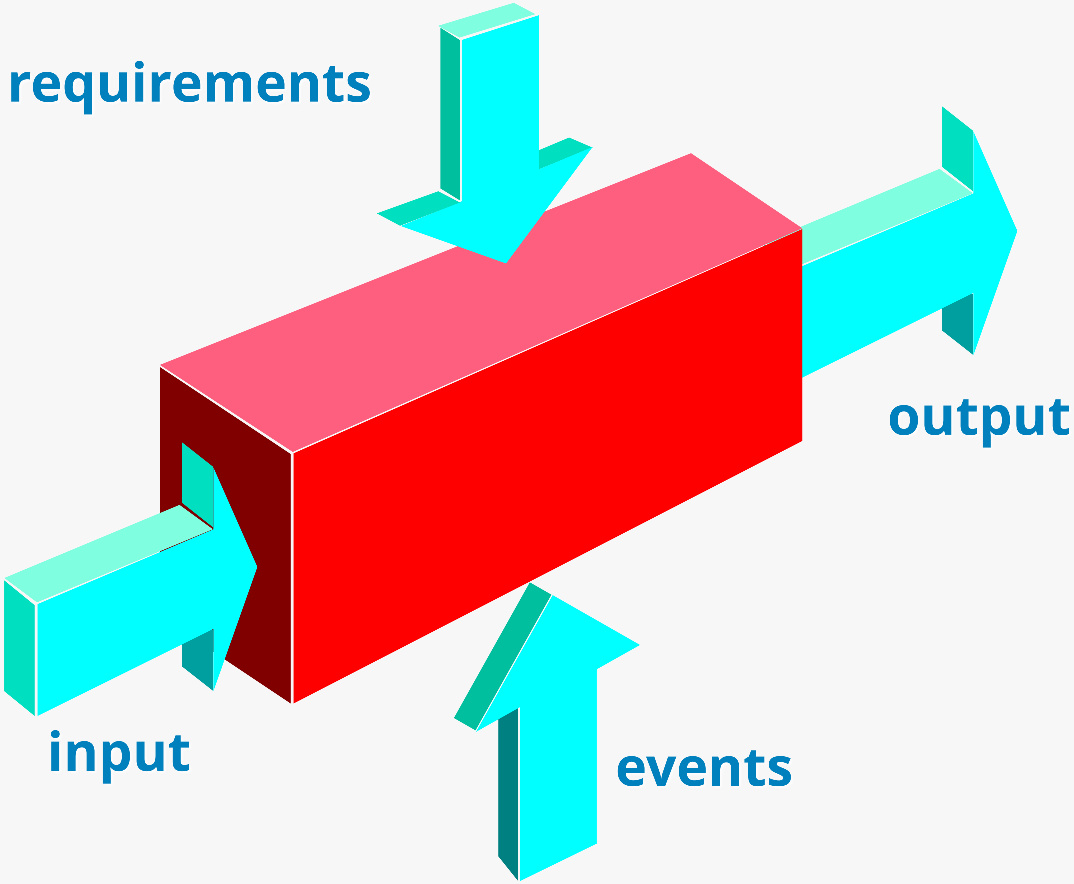
Nested Loops

Concatenated Loops

Unstructured Loops

People
Innovation
Excellence

Black-Box Testing



Testing Concepts for Web and MobileApps

Testing Quality Dimensions-I

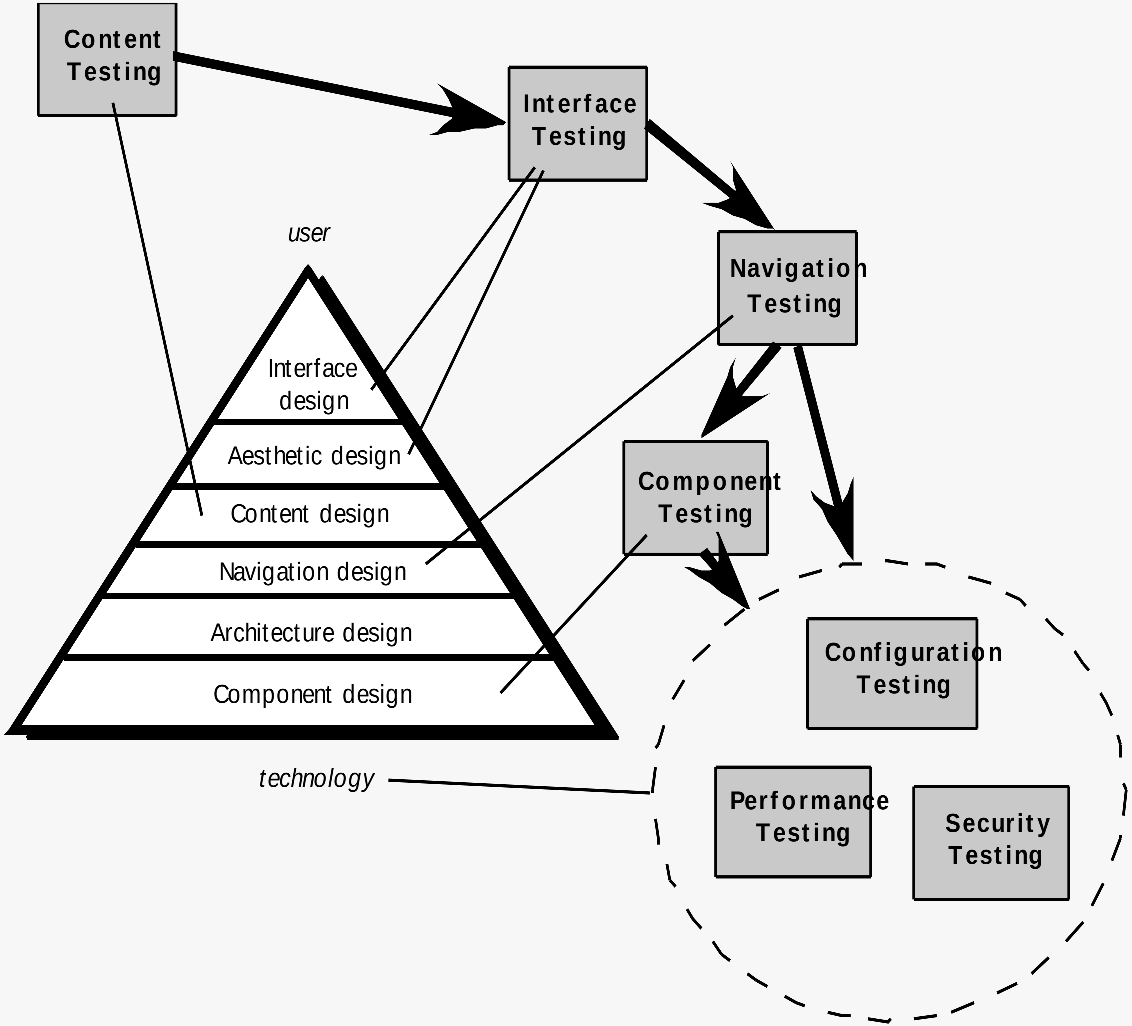
- ***Content*** is evaluated at both a syntactic and semantic level.
  - syntactic level
  - semantic level
- ***Function*** is tested for correctness, instability, and general conformance to appropriate implementation standards (e.g.,Java or XML language standards).
- ***Structure*** is assessed to ensure that it
  - properly delivers WebApp content and function
  - is extensible
  - can be supported as new content or functionality is added.
- ***Usability*** is tested to ensure that each category of user
  - is supported by the interface
  - can learn and apply all required navigation syntax and semantics
- ***Navigability*** is tested to ensure that
  - all navigation syntax and semantics are exercised to uncover any navigation errors (e.g., dead links, improper links, erroneous links).

Testing Quality Dimensions-II

- ***Performance*** is tested under a variety of operating conditions, configurations, and loading to ensure that
  - the system is responsive to user interaction
  - the system handles extreme loading without unacceptable operational degradation
- ***Compatibility*** is tested by executing the WebApp in a variety of different host configurations on both the client and server sides.
  - The intent is to find errors that are specific to a unique host configuration.
- ***Interoperability*** is tested to ensure that the WebApp properly interfaces with other applications and/or databases.
- ***Security*** is tested by assessing potential vulnerabilities and attempting to exploit each.
  - Any successful penetration attempt is deemed a security failure.

Testing Concepts for Web and MobileApps

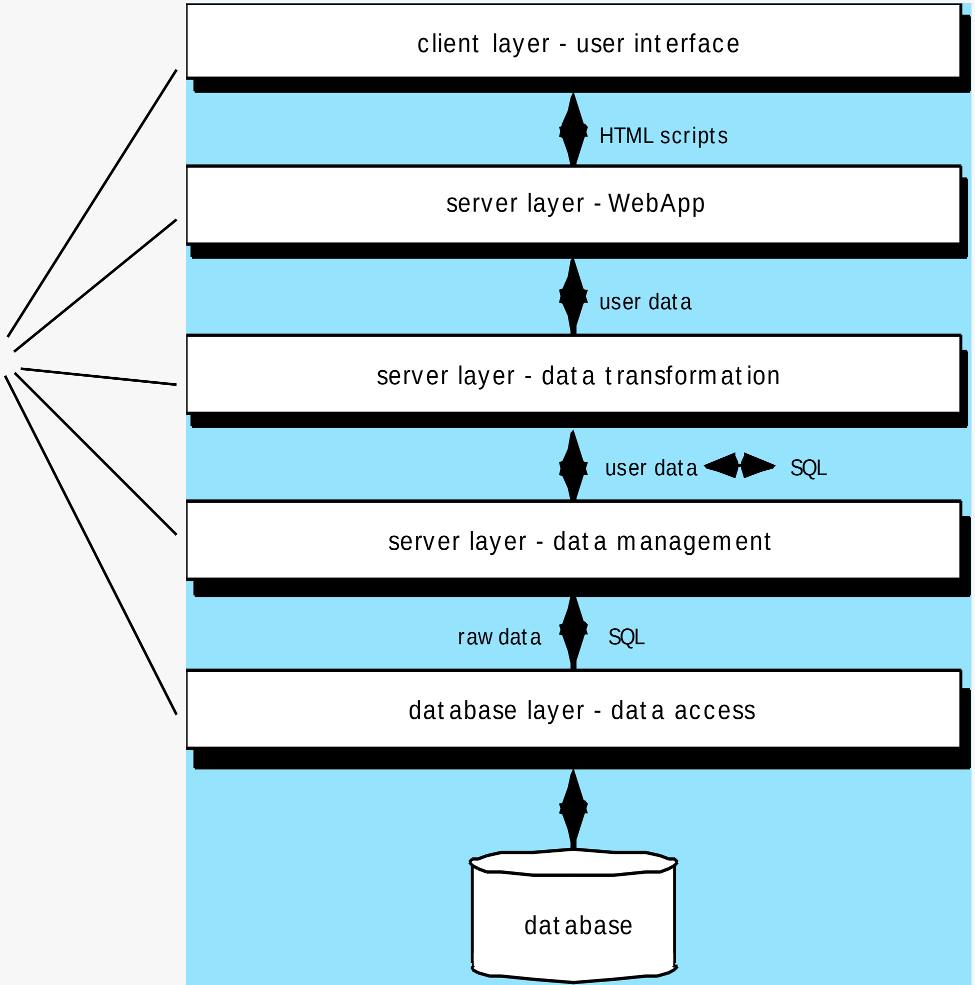
The Testing Process



Testing Concepts for Web and MobileApps

Database Testing

Tests are defined for each layer



client layer - user interface

HTML scripts

server layer - WebApp

user data

server layer - data transformation

user data

SQL

server layer - data management

raw data

SQL

database layer - data access

database

People
Innovation
Excellence

Testing Concepts for Web and MobileApps

Testing Interface Mechanisms-I

- ***Links***—navigation mechanisms that link the user to some other content object or function.
- ***Forms***—a structured document containing blank fields that are filled in by the user. The data contained in the fields are used as input to one or more WebApp functions.
- ***Client-side scripting***—a list of programmed commands in a scripting language (e.g., Javascript) that handle information input via forms or other user interactions
- ***Dynamic HTML***—leads to content objects that are manipulated on the client side using scripting or cascading style sheets (CSS).
- ***Client-side pop-up windows***—small windows that pop-up without user interaction. These windows can be content-oriented and may require some form of user interaction.

Testing Interface Mechanisms-II

- ***CGI scripts***—a common gateway interface (CGI) script implements a standard method that allows a Web server to interact dynamically with users (e.g., a WebApp that contains forms may use a CGI script to process the data contained in the form once it is submitted by the user).
- ***Streaming content***—rather than waiting for a request from the client-side, content objects are downloaded automatically from the server side. This approach is sometimes called "push" technology because the server pushes data to the client.
- ***Cookies***—a block of data sent by the server and stored by a browser as a consequence of a specific user interaction. The content of the data is WebApp-specific (e.g., user identification data or a list of items that have been selected for purchase by the user).
- ***Application specific interface mechanisms***—include one or more "macro" interface mechanisms such as a shopping cart, credit card processing, or a shipping cost calculator.

Navigation Testing

The following navigation mechanisms should be tested:

- ***Navigation links***—these mechanisms include internal links within the WebApp, external links to other WebApps, and anchors within a specific Web page.
- ***Redirects***—these links come into play when a user requests a non-existent URL or selects a link whose destination has been removed or whose name has changed.
- ***Bookmarks***—although bookmarks are a browser function, the WebApp should be tested to ensure that a meaningful page title can be extracted as the bookmark is created.
- ***Frames and framesets***—tested for correct content, proper layout and sizing, download performance, and browser compatibility
- ***Site maps***—Each site map entry should be tested to ensure that the link takes the user to the proper content or functionality.
- ***Internal search engines***—Search engine testing validates the accuracy and completeness of the search, the error-handling properties of the search engine, and advanced search features

Configuration Testing

- Server-side
  - Is the WebApp fully compatible with the server OS?
  - Are system files, directories, and related system data created correctly when the WebApp is operational?
  - Do system security measures (e.g., firewalls or encryption) allow the WebApp to execute and service users without interference or performance degradation?
  - Has the WebApp been tested with the distributed server configuration (if one exists) that has been chosen?
  - Is the WebApp properly integrated with database software? Is the WebApp sensitive to different versions of database software?
  - Do server-side WebApp scripts execute properly?
  - Have system administrator errors been examined for their affect on WebApp operations?
  - If proxy servers are used, have differences in their configuration been addressed with on-site testing?

People
Innovation
Excellence

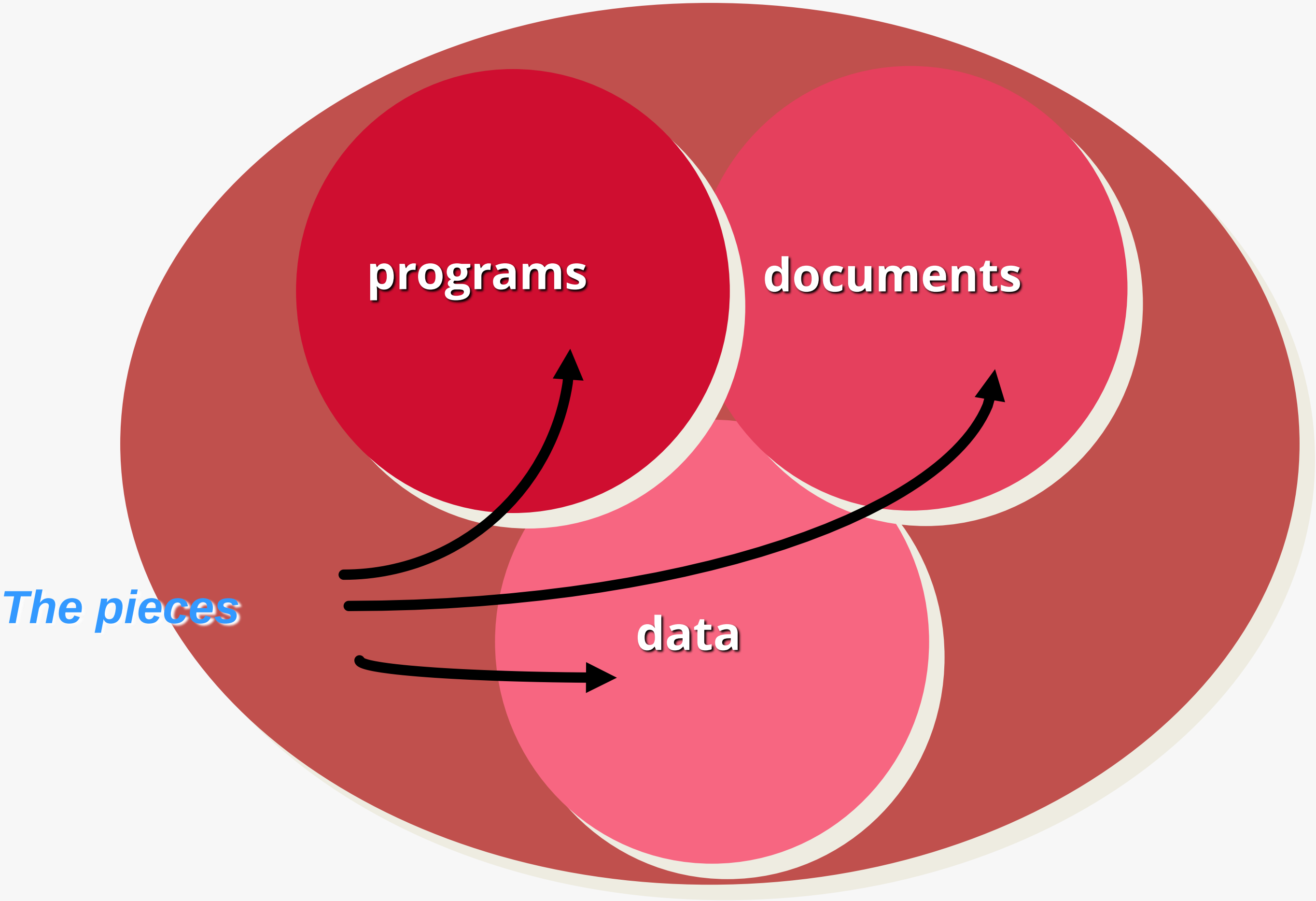
Configuration Testing

- Client-side
  - *Hardware*—CPU, memory, storage and printing devices
  - *Operating systems*—Linux, Macintosh OS, Microsoft Windows, a mobile-based OS
  - *Browser software*—Internet Explorer, Mozilla/Netscape, Opera, Safari, and others
  - *User interface components*—Active X, Java applets and others
  - *Plug-ins*—QuickTime, RealPlayer, and many others
  - *Connectivity*—cable, DSL, regular modem, T1
- The number of configuration variables must be reduced to a manageable number

Load Testing

- The intent is to determine how the WebApp and its server-side environment will respond to various loading conditions
  - *N,* the number of concurrent users
  - *T,* the number of on-line transactions per unit of time
  - *D,* the data load processed by the server per transaction
- Overall throughput, *P,* is computed in the following manner:
  - $P = N \times T \times D$

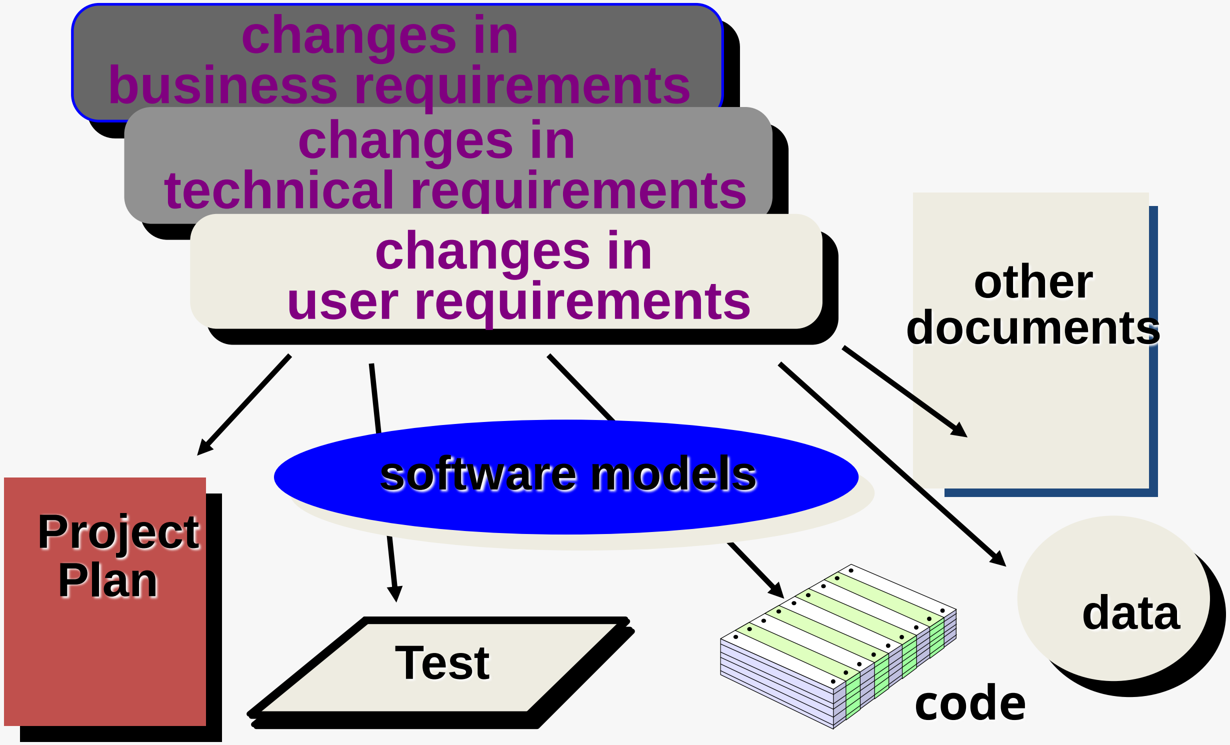
The Software Configuration



People
Innovation
Excellence

The Software Configuration

What Are These Changes?


changes in business requirements
changes in technical requirements
changes in user requirements
other documents
software models
Project Plan
Test
code
data

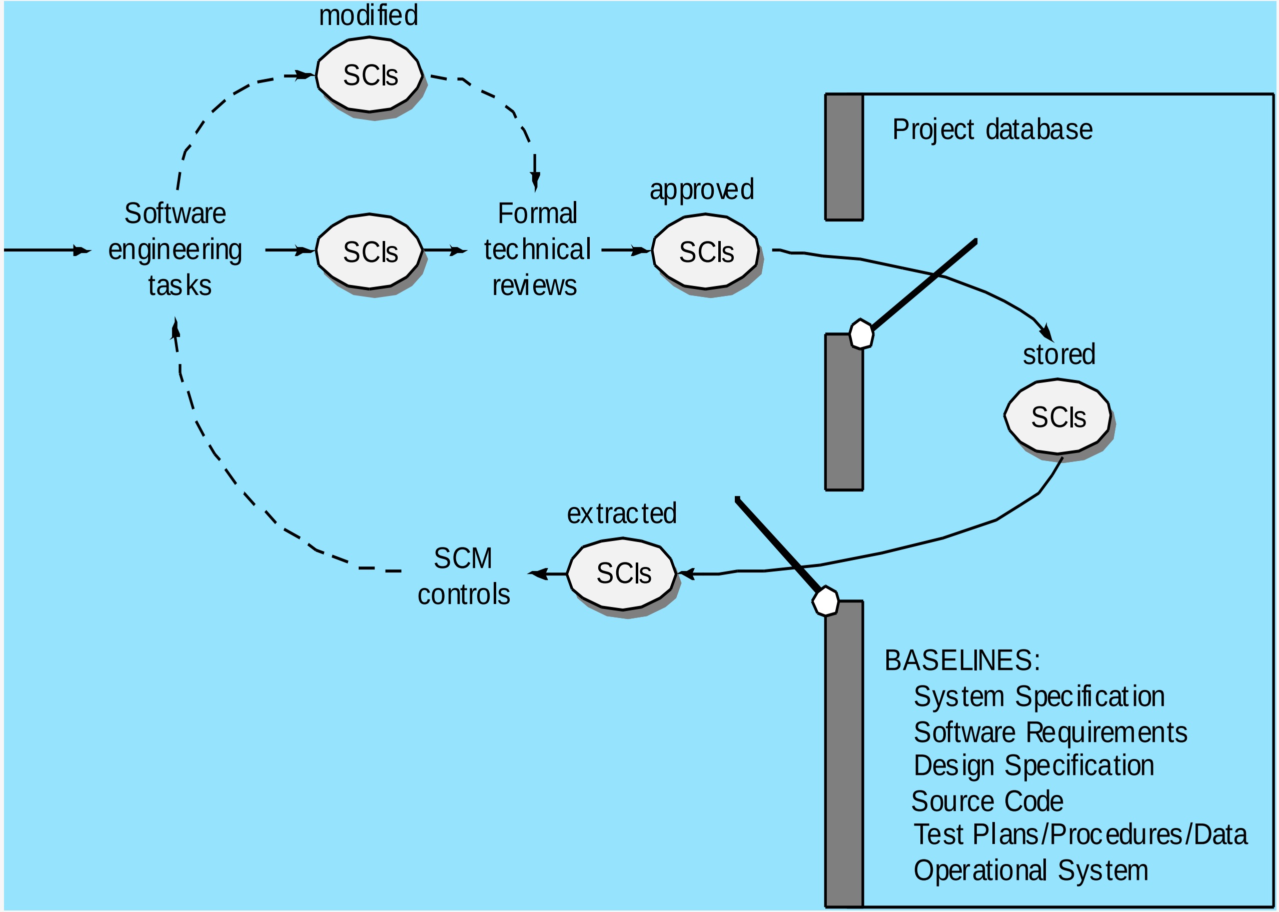
The Software Configuration

Baselines

- The IEEE (IEEE Std. No. 610.12-1990) defines a baseline as:
  - A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures.
- a baseline is a milestone in the development of software that is marked by the delivery of one or more software configuration items and the approval of these SCIs that is obtained through a formal technical review

The Software Configuration

Baselines

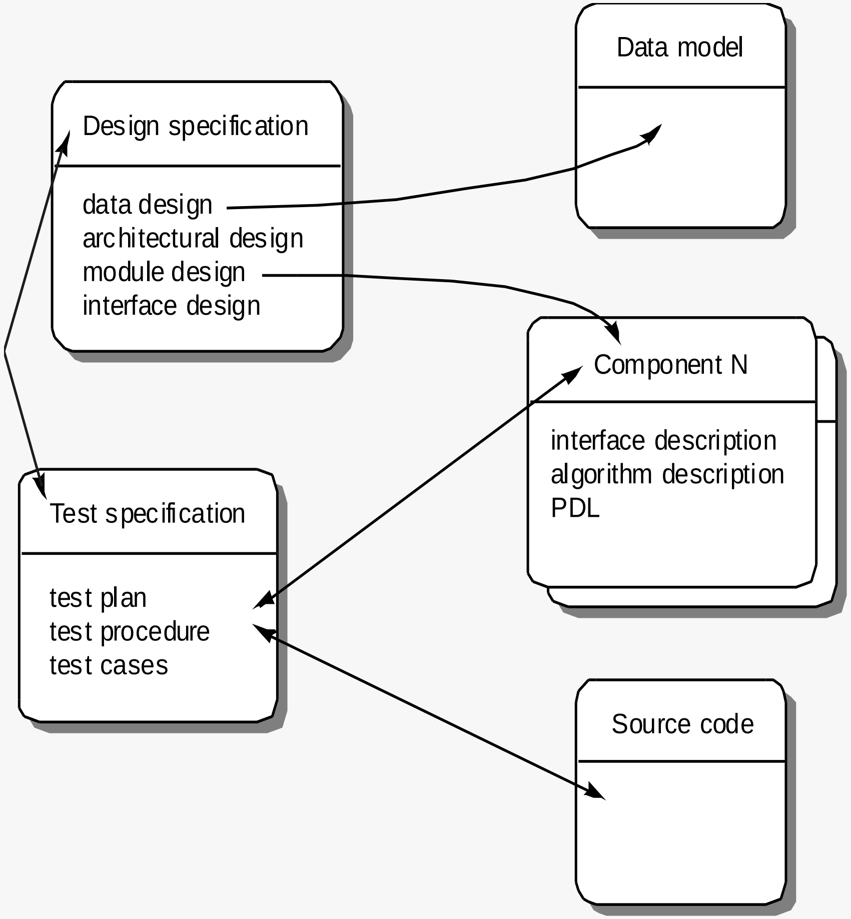


BASELINES:
- System Specification
- Software Requirements
- Design Specification
- Source Code
- Test Plans/Procedures/Data
- Operational System

People
Innovation
Excellence

The Software Configuration

Software Configuration Objects



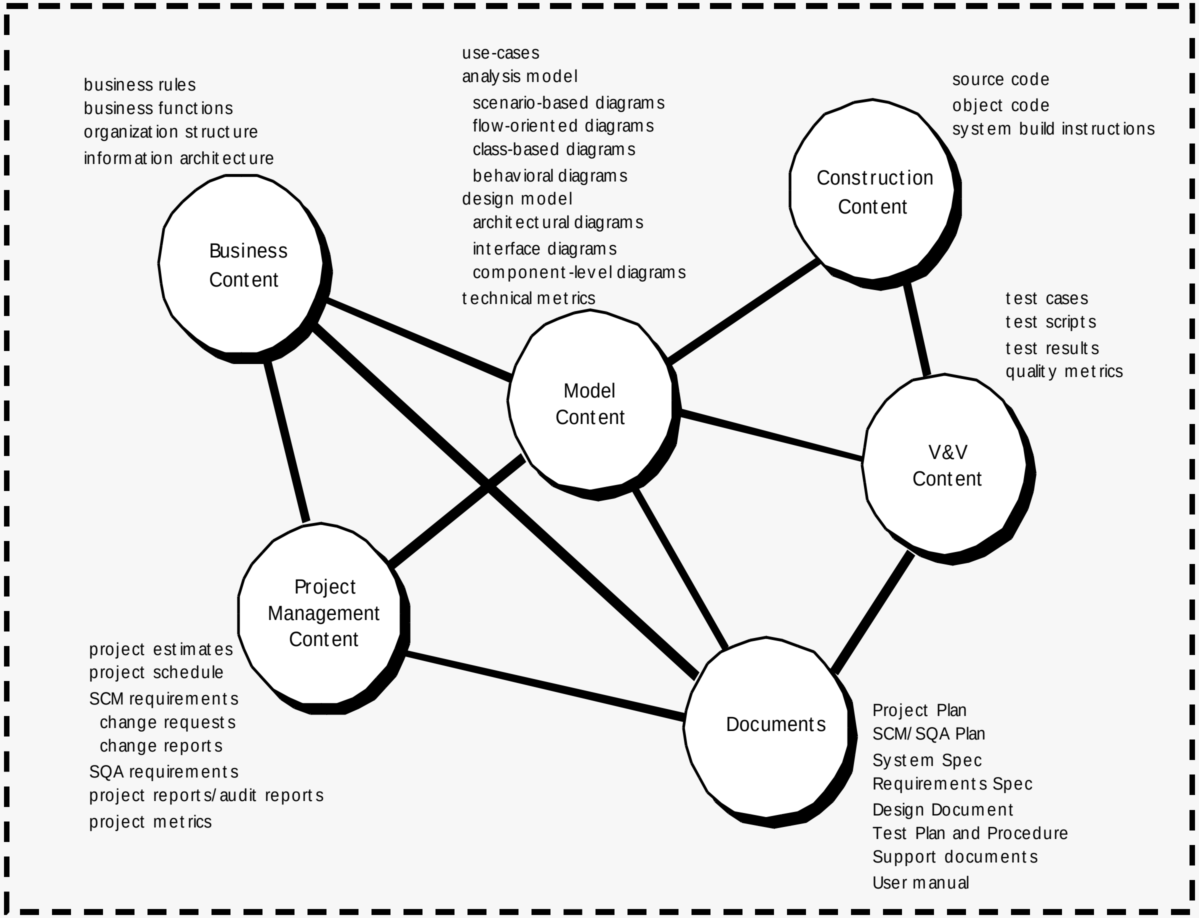
People
Innovation
Excellence

SCM Repository

- The SCM repository is the set of mechanisms and data structures that allow a software team to manage change in an effective manner
- The repository performs or precipitates the following functions [For89]:
  - Data integrity
  - Information sharing
  - Tool integration
  - Data integration
  - Methodology enforcement
  - Document standardization

The Software Configuration

Repository Content



People
Innovation
Excellence

The Software Configuration

Repository Features

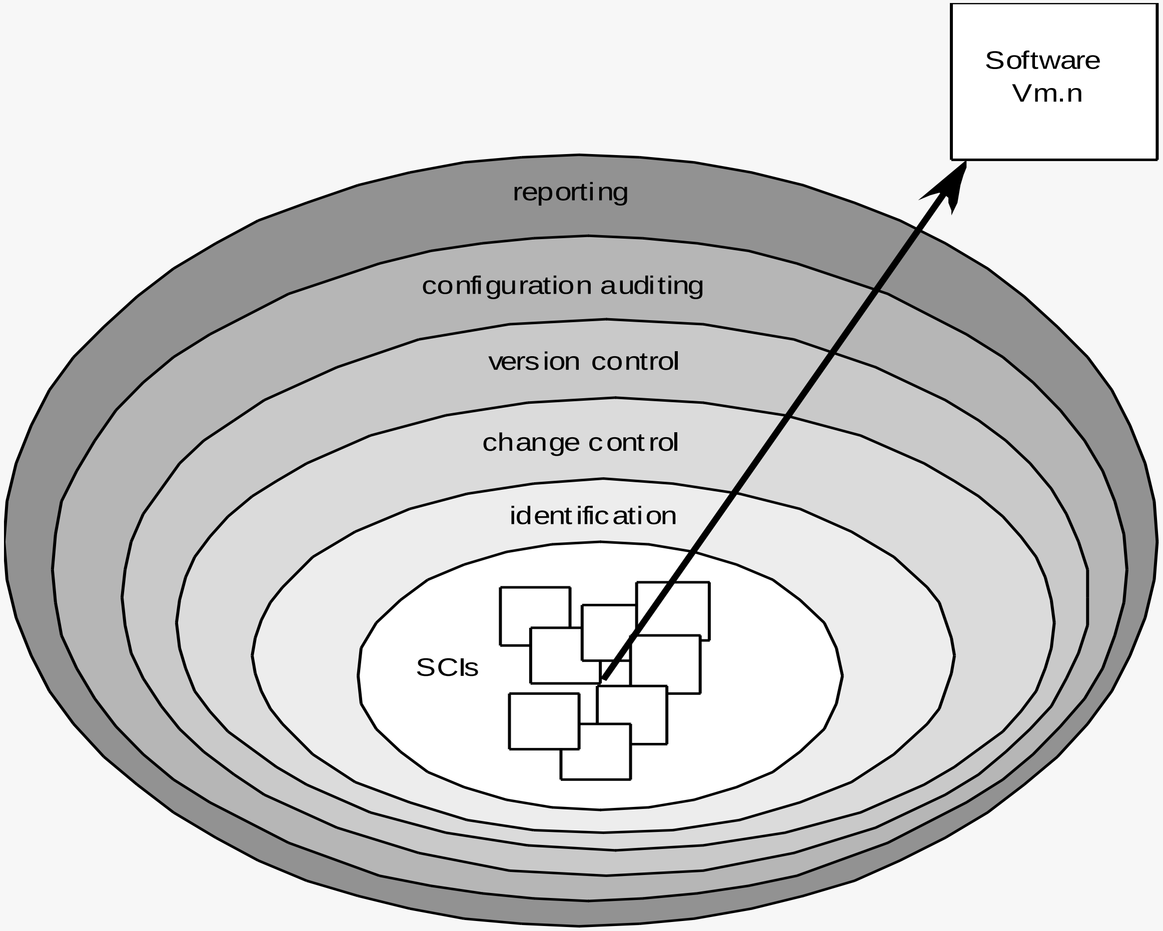
- **Versioning.**
  - saves all of these versions to enable effective management of product releases and to permit developers to go back to previous versions
- **Dependency tracking and change management.**
  - The repository manages a wide variety of relationships among the data elements stored in it.
- **Requirements tracing.**
  - Provides the ability to track all the design and construction components and deliverables that result from a specific requirement specification
- **Configuration management.**
  - Keeps track of a series of configurations representing specific project milestones or production releases. Version management provides the needed versions, and link management keeps track of interdependencies.
- **Audit trails.**
  - establishes additional information about when, why, and by whom

The Software Configuration

SCM Elements

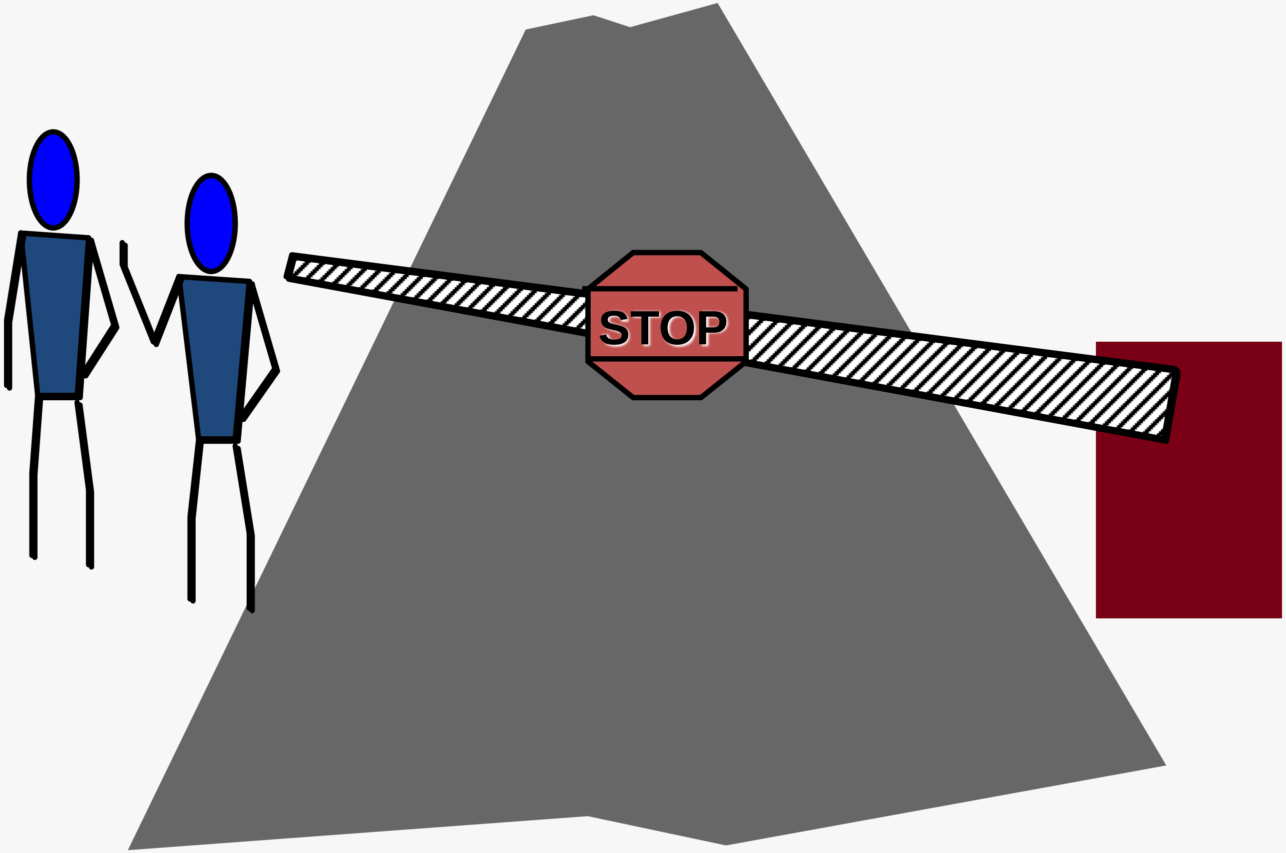
- *Component elements*—a set of tools coupled within a file management system (e.g., a database) that enables access to and management of each software configuration item.
- *Process elements*—a collection of procedures and tasks that define an effective approach to change management (and related activities) for all constituencies involved in the management, engineering and use of computer software.
- *Construction elements*—a set of tools that automate the construction of software by ensuring that the proper set of validated components (i.e., the correct version) have been assembled.
- *Human elements*—to implement effective SCM, the software team uses a set of tools and process features (encompassing other CM elements)

The SCM Process



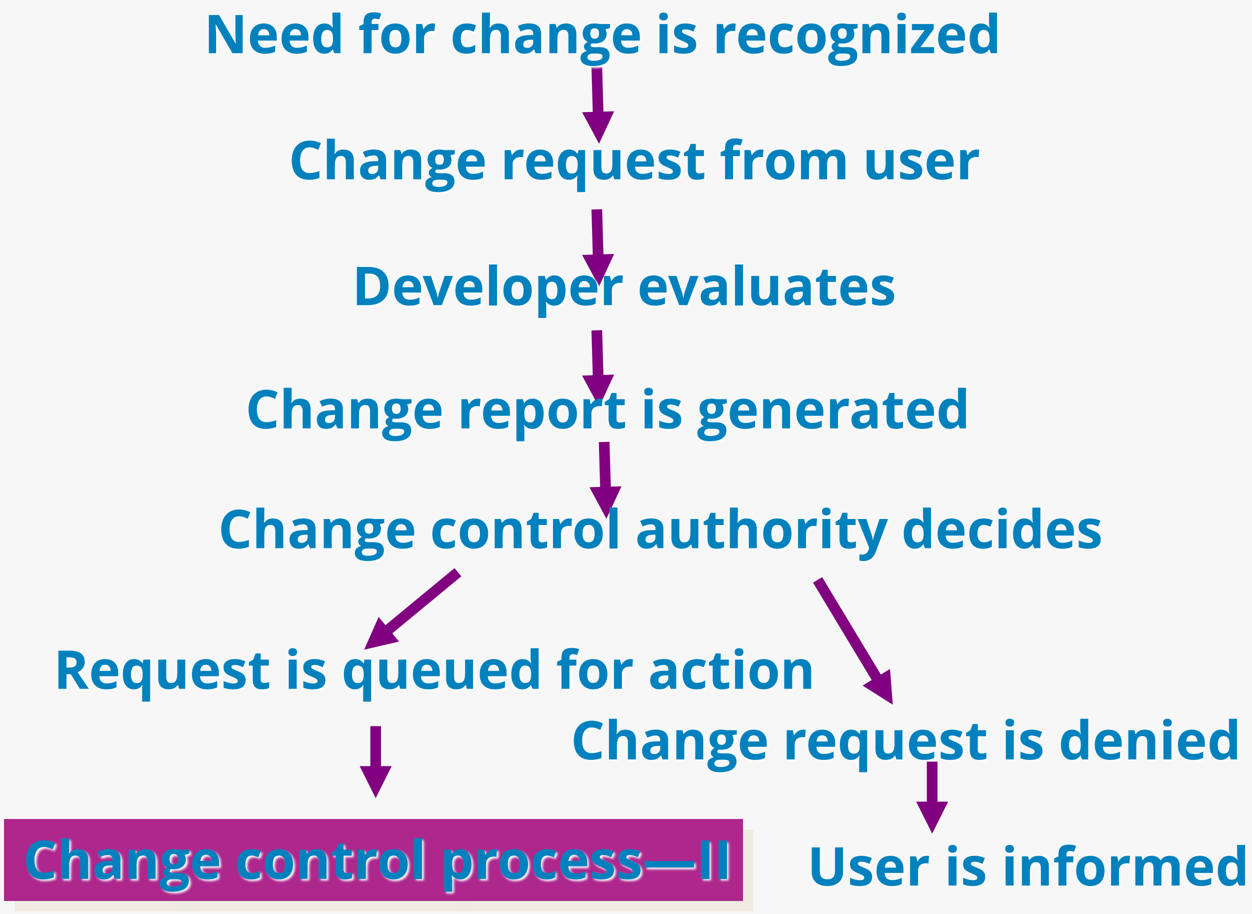
The SCM Process

Change Control



The SCM Process

Change Control Process—I



Need for change is recognized

Change request from user

Developer evaluates

Change report is generated

Change control authority decides

Request is queued for action

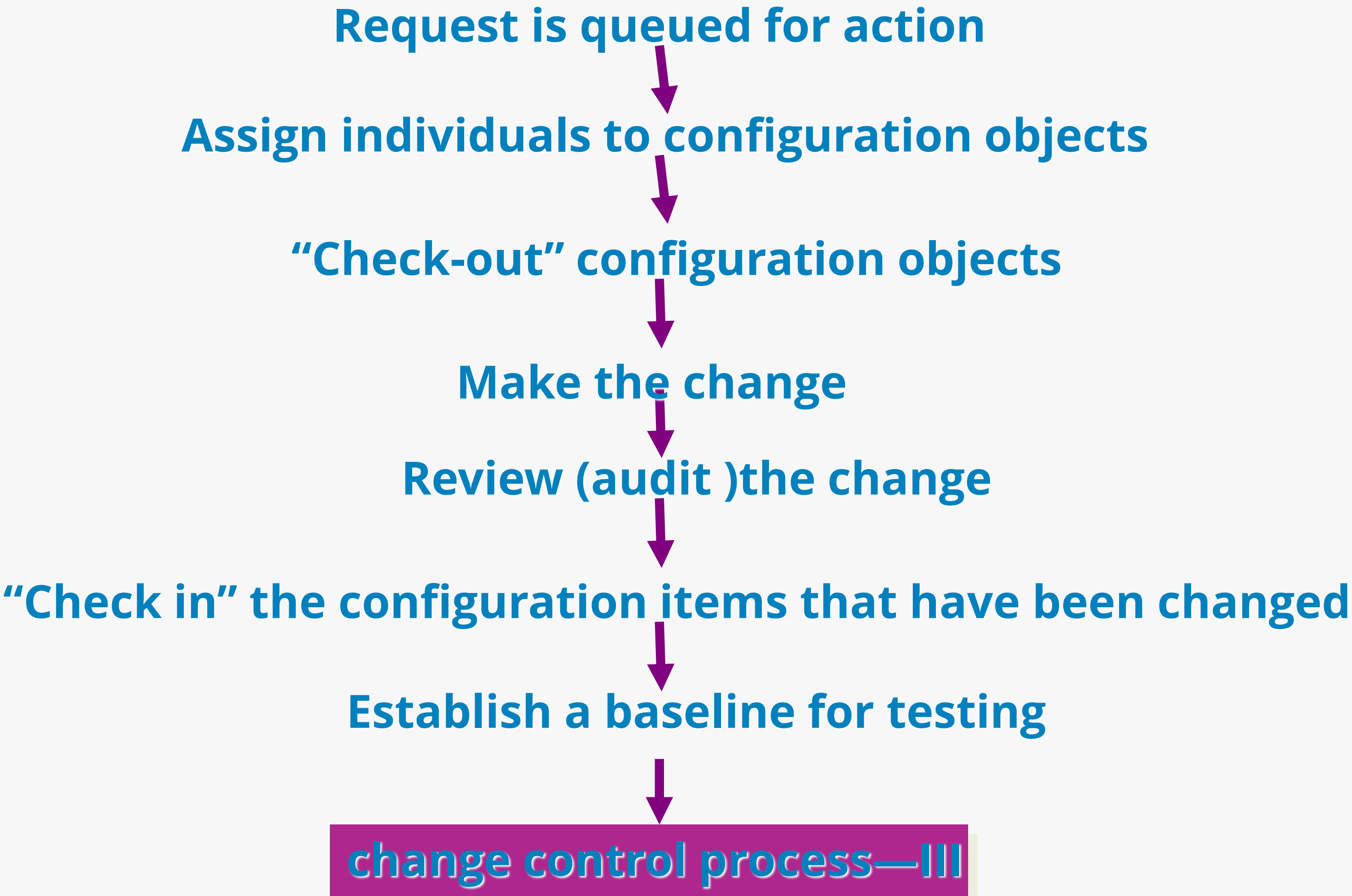
Change request is denied

Change control process—II

User is informed

The SCM Process

Change Control Process-II



Request is queued for action

Assign individuals to configuration objects

"Check-out" configuration objects

Make the change

Review (audit)the change

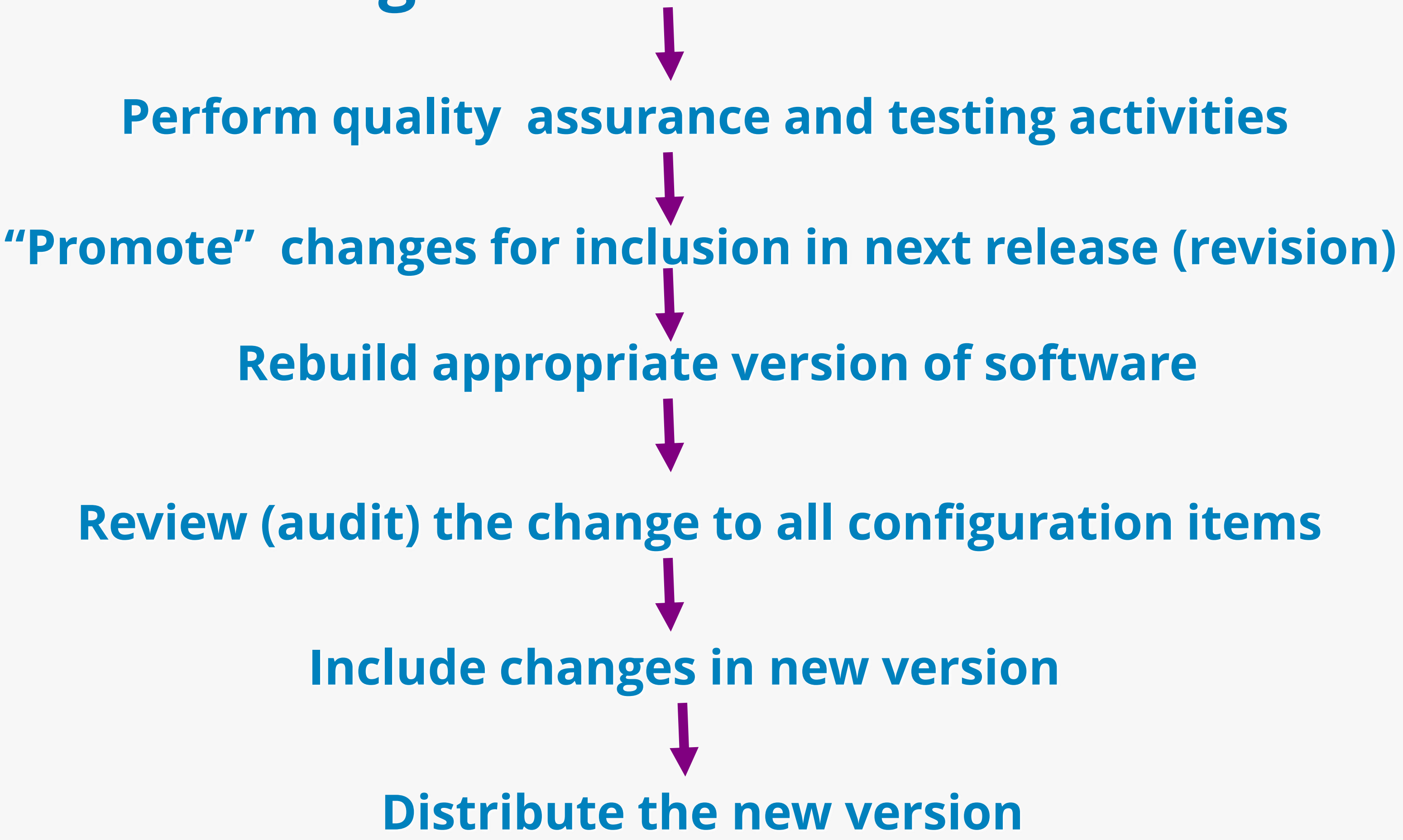
"Check in" the configuration items that have been changed

Establish a baseline for testing

change control process—III

The SCM Process

Change Control Process-III



Perform quality assurance and testing activities

“Promote” changes for inclusion in next release (revision)

Rebuild appropriate version of software

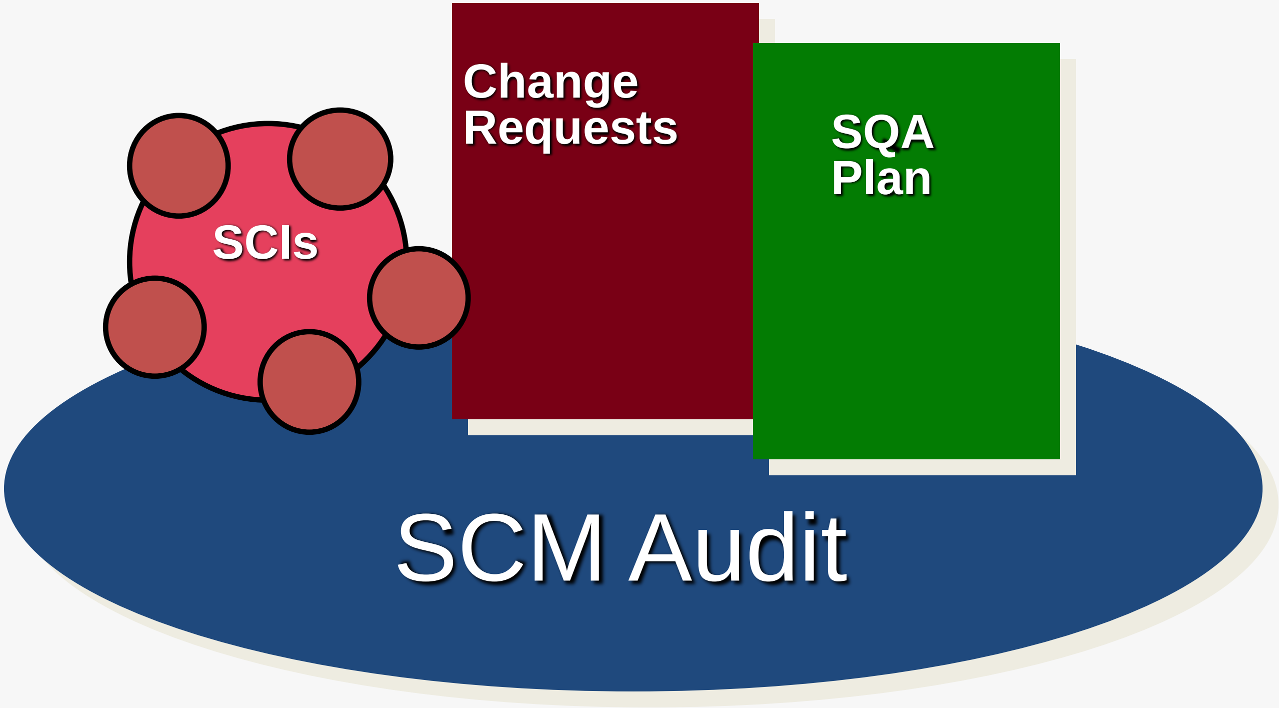
Review (audit) the change to all configuration items

Include changes in new version

Distribute the new version

The SCM Process

Auditing


SCIs

Change Requests

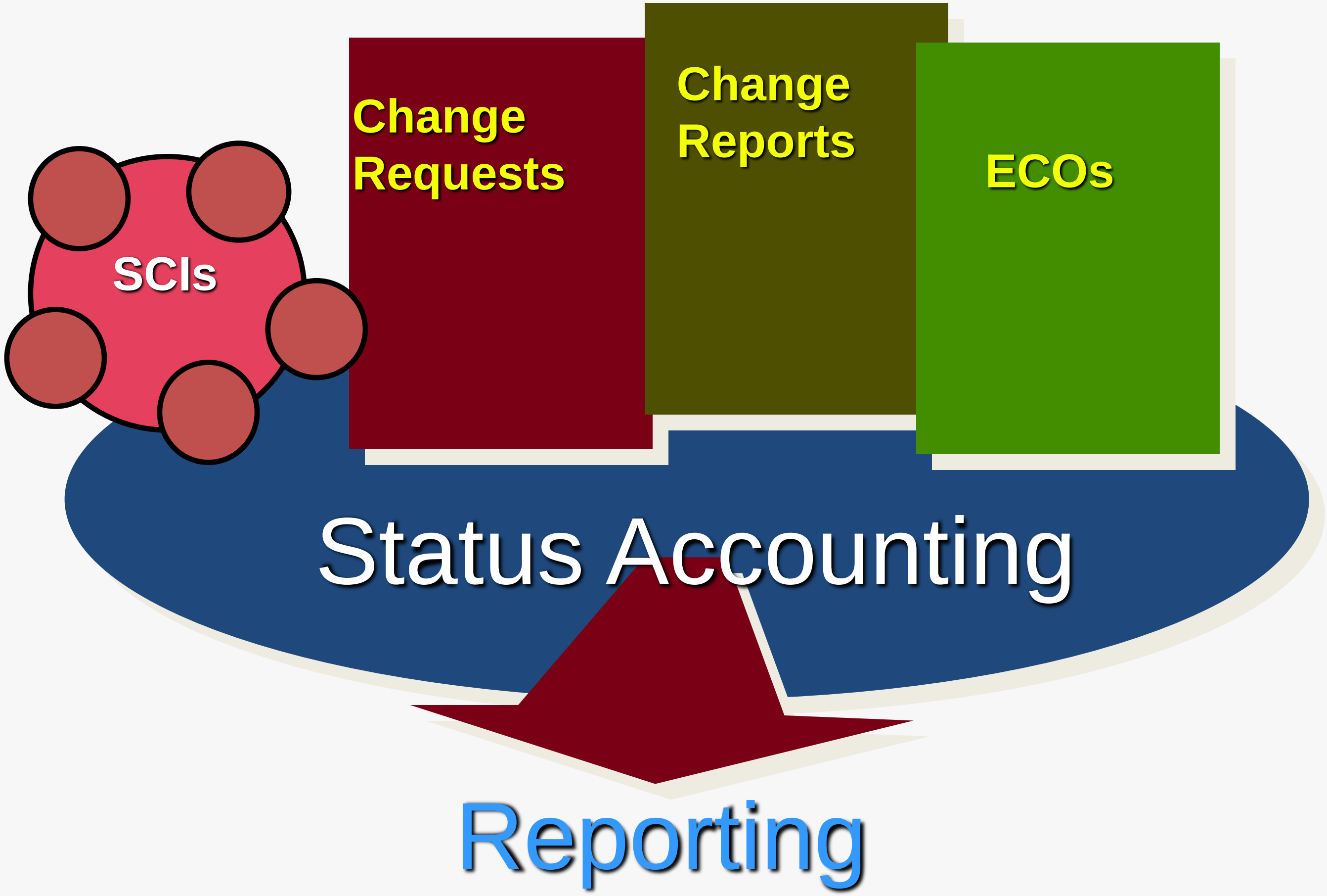
SQA Plan

SCM Audit

People
Innovation
Excellence

The SCM Process

Status Accounting



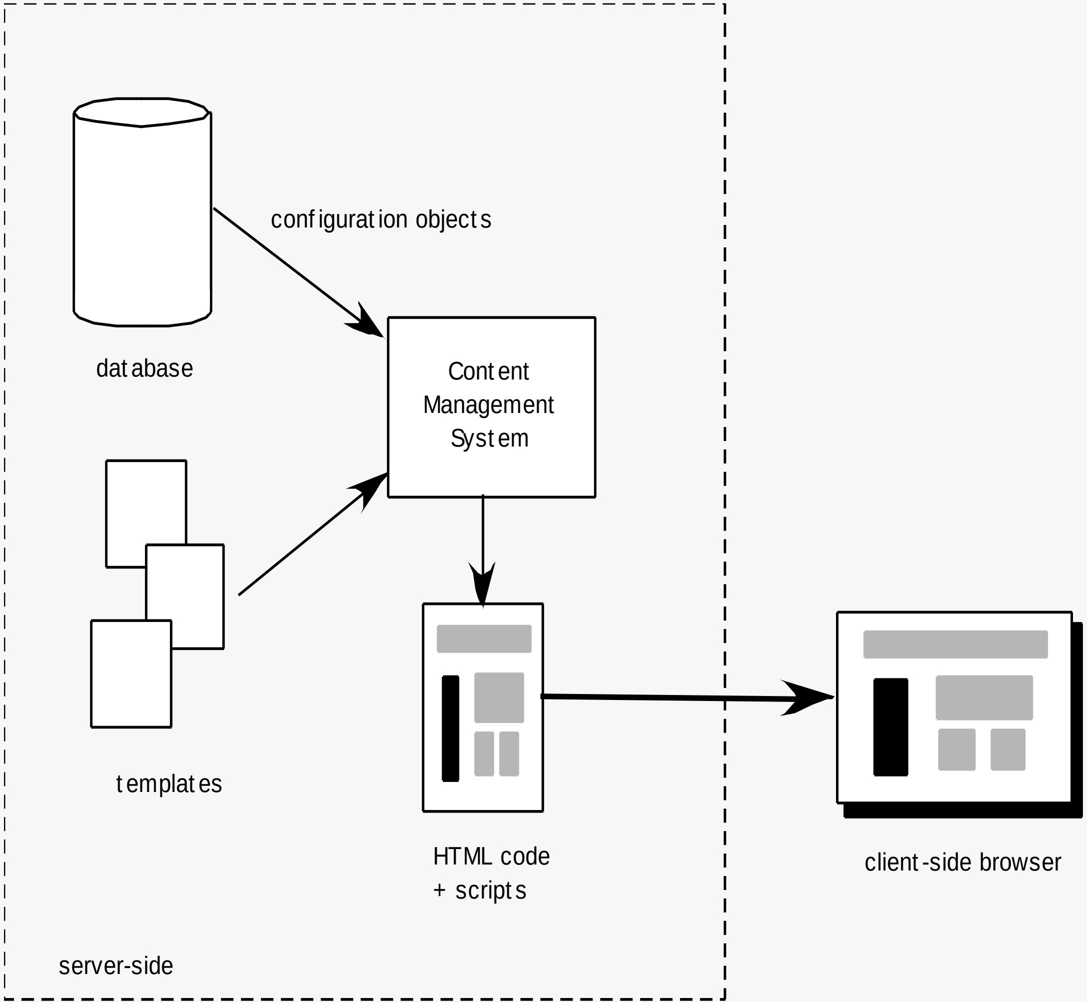
Configuration Management for Web and MobileApps

- **Content.**
  - A typical WebApp contains a vast array of content—text, graphics, applets, scripts, audio/video files, forms, active page elements, tables, streaming data, and many others.
  - The challenge is to organize this sea of content into a rational set of configuration objects and then establish appropriate configuration control mechanisms for these objects.
- **People.**
  - Because a significant percentage of WebApp development continues to be conducted in an ad hoc manner, any person involved in the WebApp can (and often does) create content.

Configuration Management for Web and MobileApps

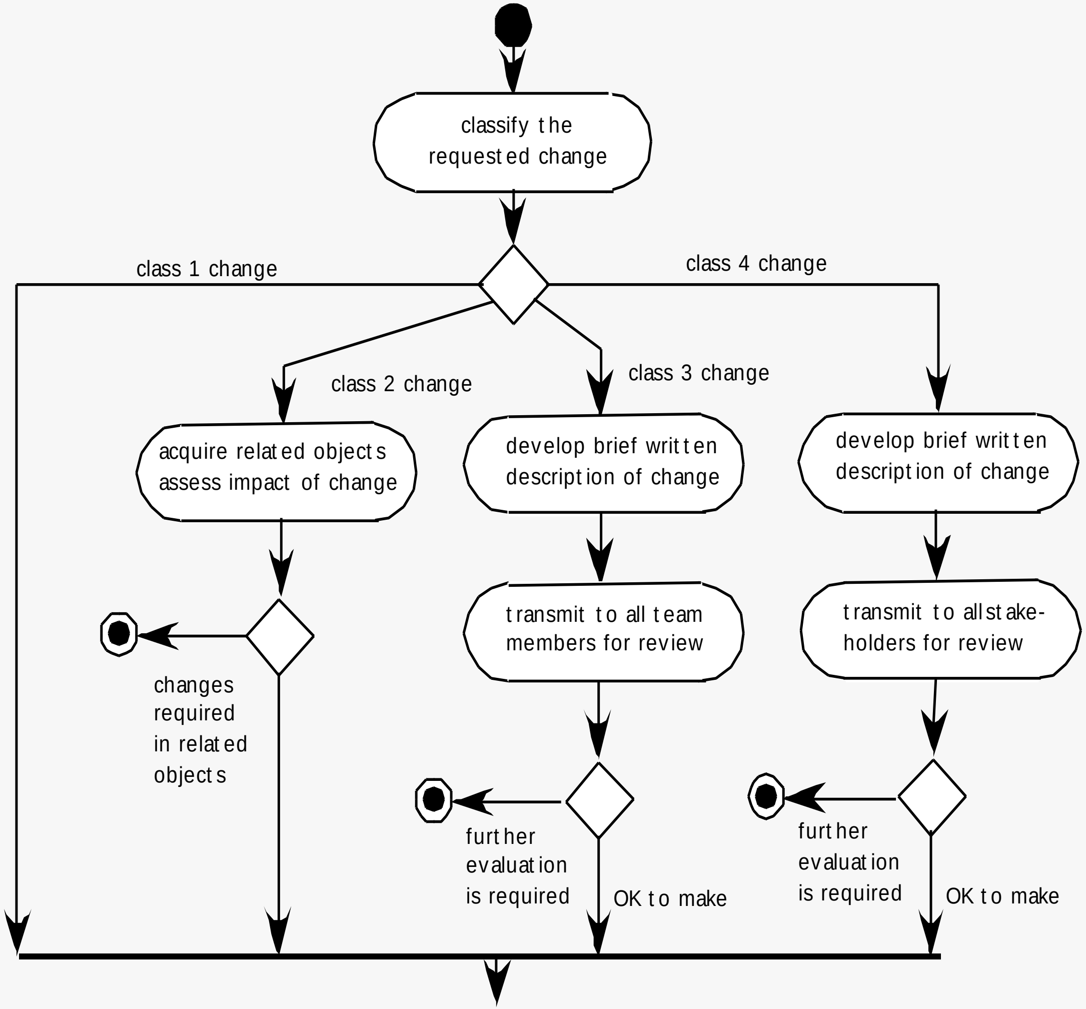
- **Scalability.**
  - As size and complexity grow, small changes can have far-reaching and unintended affects that can be problematic. Therefore, the rigor of configuration control mechanisms should be directly proportional to application scale.
- **Politics.**
  - Who 'owns' a WebApp?
  - Who assumes responsibility for the accuracy of the information on the Web site?
  - Who assures that quality control processes have been followed before information is published to the site?
  - Who is responsible for making changes?
  - Who assumes the cost of change?

Content Management



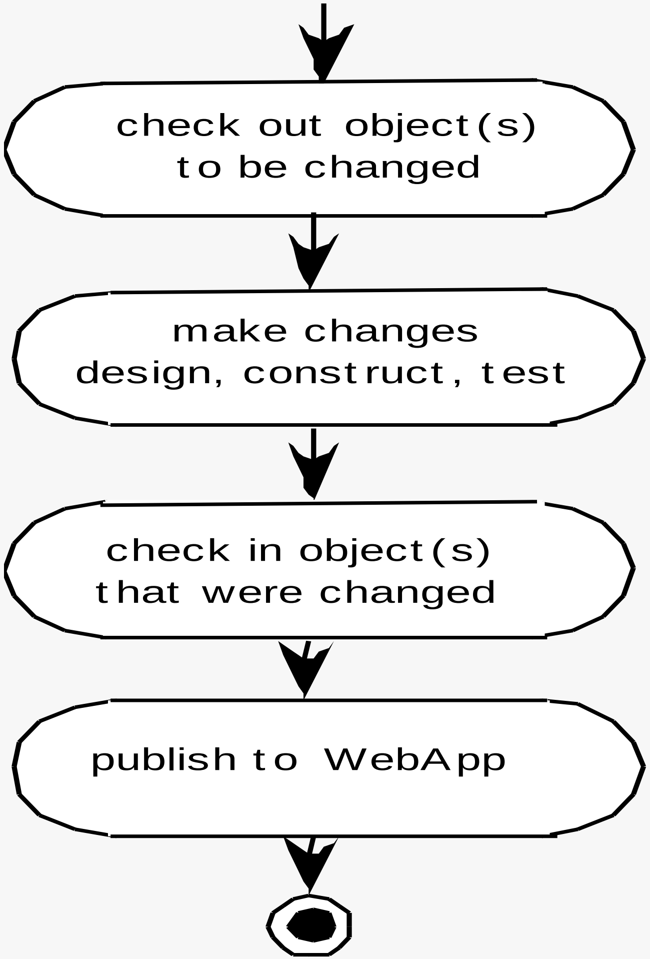
Configuration Management for Web and MobileApps

Change Management for WebApps-I


classify the requested change
class 1 change
class 4 change
class 2 change
class 3 change
acquire related objects assess impact of change
develop brief written description of change
develop brief written description of change
transmit to all team members for review
transmit to all stake-holders for review
changes required in related objects
further evaluation is required
OK to make
further evaluation is required
OK to make

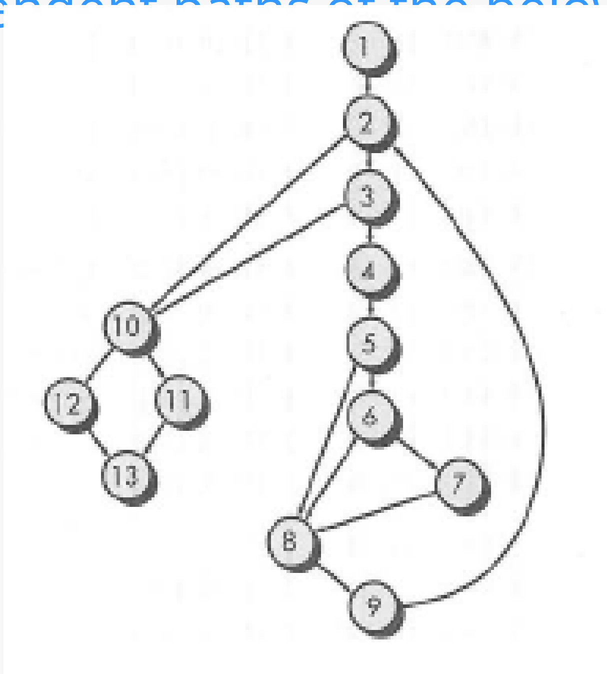
Configuration Management for Web and MobileApps

Change Management for WebApps-II


check out object(s) to be changed
make changes design, construct, test
check in object(s) that were changed
publish to WebApp

Exercises

1. What is the difference between black-box and white-box testing ?
2. Determine the cyclomatic complexity and define the independent paths of the below flow graph :



People
Innovation
Excellence

Exercises

3. In your own words, describe why the class is the smallest reasonable unit for testing wlthin an OO system
4. What is the difference between thread-based and use-based strategies for integration testing? How does cluster testing fit in?
5. Compatibility is an important quality dimension. What must be tested to ensure that compatibility exisis for a WebApp ?
6. What is the difference between testing for navigation syntax and navigation semantics?
7. Assume that you're the manager of a small project. What baselines would you define for the project and how would you control them?
8. Briefly describe the differences between SCM for conventional software and SCM for WebApps.

References

- Pressman, R.S. (2015). ***Software Engineering : A Practioner's Approach. 8th ed***. McGraw-Hill Companies.Inc, Americas, New York. ISBN : 978 1 259 253157.
- Software Testing

  http://www.io.com/~wazmo/qa/
- SW Testing Storm

  http://www.mtsu.edu/~storm
- Conventional sw testing

  http://www.testingexcellence.com/conventional-software-testing-on-an-extreme-programming-team/
- Testing OO SW

  http://diwww.epfl.ch/researchlgl/research/ongoing/testing.html
- Web testing

People
Innovation
Excellence

Q & A