Study Objectives

• Systems Implementation
• Data Migration
• Change Over
Implementation

• Which ones of these activities are part of software implementation?

A. Data Conversion
B. Code Migration to production
C. UAT
D. Prototyping
E. Training plans
F. Production SLA migration
G. Change Management
H. Monitoring
I. Incident Management
J. Hardware Deployment
K. Roll-back
L. Performance Testing
M. Regression Testing
Which ones of these activities are part of software implementation?

A. Data Conversion
B. Code Migration to production
C. UAT
D. Prototyping
E. Training plans
F. Production SLA migration
G. Change Management
H. Monitoring
I. Incident Management
J. Hardware Deployment
K. Roll-back
L. Performance Testing
M. Regression Testing
Data Conversion and Migration

• Objective is to convert existing-system data into a new-system data to meet the needs of new system, while preserving the meaning and integrity of data

• Conversion of Data from old system to new system
  – One database to a different database
  – Files to a database and vice versa

• Large Data Conversion is a project in itself which follows the SDLC process
Steps for Data Conversion

- What data to be converted
- Identify “from” to “to” format, including need for any changes in between
- Determine criteria and parameters of successful conversion
- Come up with the sequence of the data conversion tasks
- Come up with the fall-out reports for any manual clean up. Also come up with the people responsible for manual clean up (business, IT, or both)
- Maintain audit trail
- Developing and testing conversion programs and/or scripts
- Come up with a few iterations to identify “fall out” and “gaps” in the conversion process, to improve conversion success rate
- Making any hardware necessary in production to host the new data
- Take back-up of the data before actual conversion
- Running actual conversion
- Deploying converted data in production (sometimes it may require to deploy the code and configuration along with data)
Data Migration

• Multiple ways possible:
  1. Convert the data in a “staging” environment. Validate the data and then migrate the validated data to production
  2. Replicate existing data from current production environment to new production environment, while keeping the existing data and system unchanged. Apply the conversion logic to the new production environment thereafter
Data Migration Rollback

• Similarly, multiple-ways possible depending on how conversion was performed

  1. Restore the data from back-up
     • Restore the data during the Change Management, if the system is down, OR
     • If the new system is already cut-over (real-time transaction is applied to the new systems), then need to be retroactively apply real-time transactions that was already processed, to the old system’s back-up in case of roll back

  2. Undo the data changes (like DDL changes)
     • Undo the DDL change, during the Change Management, if the system is down
     • If the new system is already cut-over (real-time transaction is applied to the new systems), then also Undo the DDL change
Data Conversion Success Factors

• Completeness
  – Number of records in the new system vs. old system should be identical

• Integrity
  – Data relationship across different data elements should be maintained

• Storage and Security
  – Back-up (last copy before conversion and first copy after the conversion) to be kept for future issues in case they arise
  – Unauthorized access of data to be prevented

• Consistency
  – Is the new data format consistent with original system?
  – Better if that can be, sometimes it may not be possible

• Continuity
  – Business continuity w.r.t. system’s use to be maintained
Implementation Planning

• Support Team
  – New systems requires IT support team to operate it
  – At times there could be a need to plan and add incremental resources. A gap analysis helps determine the incremental needs
  – Generally tiered support team structure is created to address production issue and outage scenarios
    • Level 1 - service desk
    • Level 2 – next level
    • Level 3 – application SMEs (subject matter experts)
Implementation Planning

• Support Functions
  – Application operational manual often need to be developed by the Support Staff. The goal is to operate the systems without active help from the Development teams
  – Generally tiered support team structure is created to address production issue and outage scenarios
    • Level 1- service desk
    • Level 2 – next level
    • Level 3 – application SMEs (subject matter experts)
  – Support process needs to be developed
  – Communication plan in case of outages needs to be defined

• SLAs (Service Level Agreement) need to be developed such as
  – Availability, for example 99.99%
  – Mean time to repair (MTTR), for example 30 minutes
  – Mean time between failures, for example 200 hours
  – Response time on the outage telephone bridge, for example 15 minutes
End User Training

- Users need to be trained before the system is implemented in production
- Training could be for
  - All the users
  - Train the trainers
  - Only users who would start using the new system right away for phased deployment of the new systems
    - Phased deployment would typically mean the new system is deployed; however, only limited number of users have access to the system based on user’s “authentication and authorization”
- Training needs are typically discussed well in advance during systems development
- Training covers how to use the system, and training modules need to be developed
- Training could be provided in multiple ways
  - Classroom training
  - CBT (computer based training)
  - Online training
Change Over (Go Live)

• Implementation of Data and Systems changes in production
• Need to be well-planned
• Requires proper Change Management (to be covered later) to ensure Business and IT leadership agree on the new systems implementation based on the test results including UAT
• Requires proper Support Function (monitoring, incident management, outage management, etc.) to be in place
• Generally requires a Go/No-Go call with the stakeholder before (for example day before) go-live
• Generally ensures User training of the new system is completed
• Multiple techniques of go-live possible...
Parallel Changeover (for Data and System)

- Running old and new system in parallel for some times
- Pros
  - Overlap reduces any Business or User risk as the old system is available in parallel
  - Systems maturity time by design
- Cons
  - Generally require separate hardware platform for the new system
  - IT and support staff must maintain both systems for some time

Source: Adapted from CISA Review Manual, ISACA
Phased Changeover (for Data and System)

- Running old system’s module in parallel to the corresponding new system module for some times. The system is broken into modules and changeover can be one of more modules at a time, but not all the modules at the same time.

**Pros**
- Overlap of module reduces Business or User risk as the old system’s module is available in parallel to the corresponding new system’s module.
- Migration could be at the component level (small and manageable chunks).

**Cons**
- Generally require separate hardware platform for the new system.
- IT and support staff must maintain both systems for some time.
- More complex to maintain modules and their relationships. A business process, like ordering, may not be easier to break into multiple modules which can work independently.
- Extension of project timeline.

Source: Adapted from CISA Review Manual, ISACA
Abrupt Changeover (for Data and System)

- Changing old system to new at a particular point of time
- Pros
  - System migration complexity is not huge
  - Only one system is to be maintained at a time by IT and support staff
- Cons
  - The new system may not function well initially, as the data conversion may not be highly successful (but we won’t know that until we convert!)
  - This could adversely impact Business or Users
  - Not a good idea for a mission critical/complex application

Source: Adapted from CISA Review Manual, ISACA
Change Management

- What is applicable to Change Management?

A. Change Made to Production Environment
B. Change must be signed off by Business, Development, and Operation Teams
C. Operations is responsible for making change to the Production environment
D. Change Management requires artifacts to indicate the functionality to be implemented is tested in lower environment and UAT is completed
Change Management

- What is applicable to Change Management?

  A. Change Made to Production Environment
  B. Change must be signed off by Business, Development, and Operation Teams
  C. Operations is responsible for making change to the Production environment
  D. Change Management requires artifacts to indicate the functionality to be implemented is tested in lower environment and UAT is completed
Change Management Document

• Often a formal document
• Generally created by Development and Operation Teams
• Generally approved by Stakeholders
  – Business Partners
  – Operations
  – Development
• Scope of Change must be clearly defined
  – Modules changes and to be deployed. All deployable must be outlined including executable files, dlls, JAR files
  – Configuration changes to be made, should be outlined
  – Hardware and Network changes, if any, should be defined
  – Checklist of the changes is completed
Change Management Document Contd.

• Expected Benefit of the Change is outlined
• Any changes to the Business and Systems SLA changes should be included
  – Include any performance degradation
  – Service disruption
  – Downtime
  – Work-around and fall back during Change implementation
Change Management Document Contd.

- **Impact of Change**
  - **Criticality—Penetration grid**
  - High Critical and High Penetration changes are high severity changes and should be carefully planned

<table>
<thead>
<tr>
<th>Criticality</th>
<th>Penetration</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Critical, Low Penetration (Mid Severity Change)</td>
<td>Low Critical, High Penetration (Low Severity Change)</td>
</tr>
<tr>
<td>High Critical, High Penetration (High Severity Change)</td>
<td>Low Critical, Low Penetration (Mid Severity Change)</td>
</tr>
</tbody>
</table>
Change Management Document Contd.

• Implementation Document
  – Implementation Document, which typically has detailed (minute-by-minute) instruction of the various steps of the changes, along with the person responsible for making the change should be complete before the change
  – Implementation should be done by Operations team

• Roll-back plan
  – should be tested in the lower environment prior to change to care for any unexpected urgency
Change Management Document Contd.

• Certification of the Change
  – Change to the system should be tested. This testing is sometimes called “smoke” testing
  – Change certification testing is typically less elaborate than progression or regression testing; however, the testing should ensure
    • The existing key functionality continue to work
    • Major progression cases work in production environment
    • Performance of the system is not adversely impacted after the change

• Communication of the change
  – to the stakeholders during the change
  – to the stakeholders after the change
Change Management Document Contd.

• Review of Sample Change Request Exhibit 3.25 textbook (CISA Review Manual, ISACA)

• Change, Configuration, and Release Management Example for SQL Server
  
Change Management Controls

Controls should be in place to:

- Documenting change request form
- Authorizing change
- Assessing impact
- Ensuring access to the program in production is limited to authorized team (Operation Team) only
Release Management

• Which of the following are applicable to Release Management?
  • Process of releasing software to the users
  • Process of Change Management and Deployment of the change
  • Back-out plan
  • Stakeholders buy-ins
Types of Software Release

• Major Release
  – Significant changes to the functionality
  – Number of Releases depends on the SDLC methodologies used by the organization, but typically, could be smaller number of major releases (2-3 per year for example)

• Minor Release
  – Smaller changes to the functionality
  – There could be couple of minor releases per major releases

• Emergency, Patch, or Unplanned Release
  – To address a software defect found in production
  – On as needed basis, and the goal is to minimize emergency releases
Release Planning

- Release scope determination
- Come up with the timeline of the release
- Roll-out strategy
  - For example, alpha, beta, and phased deployment to minimize risk
- Resources (User, Operations, Business, and IT) needs identification
- Roles and Responsibility Definitions
- Back-out plan
- Release performance metrics and KPIs
Upcoming Assignments/Tests

1. Group Case Study -3 (Testing): Thu 4/14 before the class

2. Final (multiple choice questions 40-50 modeled after CISA exam. Covers entire course.): Thu 4/28
   Note: We will use TU Classroom Exam Answer Sheet (bubble sheet).
   Bring your pencils in the class

Questions?
Summary of Today’s Class

• Data Conversion
• Implementation
• Focus of the Next Class and Reading
• Questions