Configuration Management for Client/Server Applications

Session 540

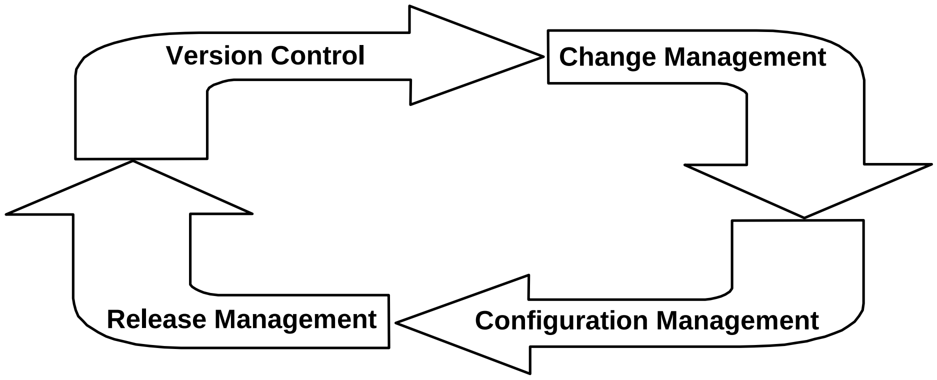
Rebecca Lawson and Adel Harris
Texas Instruments

What is Configuration Management?

ISO 9000-3 Definition

“Configuration Management provides a mechanism for identifying, controlling and tracking the versions of each software item.”

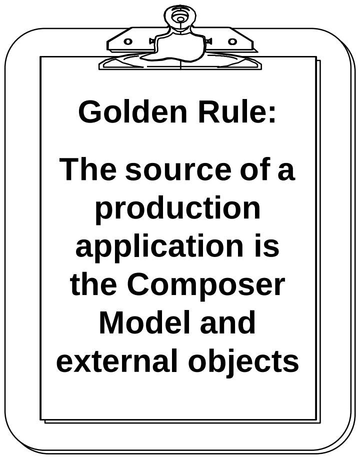
Configuration Management for Composer Development



Why Care About Configuration Management?

- Framework for planning, scheduling, and controlling application releases
- Supports delivery of integrated product
- Improves communication
- Prevents errors and manages cost
- Satisfies requirements and provides feedback loop

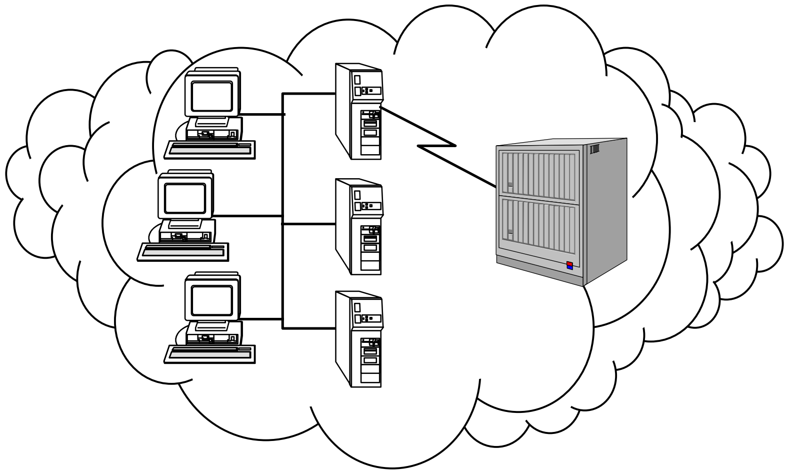
Composer Support for Configuration Management


Golden Rule:

The source of a production application is the Composer Model and external objects

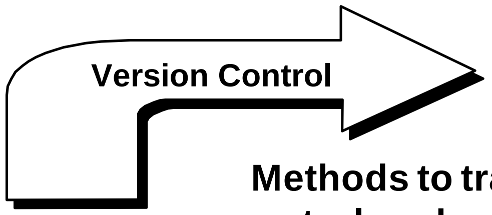
- For Composer objects:
  - – Stores multiple versions
  - – Tracks object versions
  - – Supports impact and difference analysis
  - – Propagates change
  - – Retains history of change

Client/Server Challenges



The application is everywhere...

What is Version Control?



Methods to track and control each version of applications

What to Control?

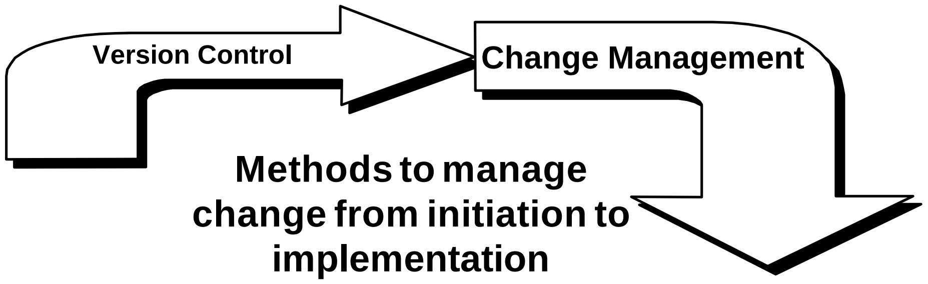
Composer objects

- Data Model
- Activity Model
- Action Blocks
- Business System Defaults
- Exit States
- Prototypes
- Procedures
- Data Structure

External objects

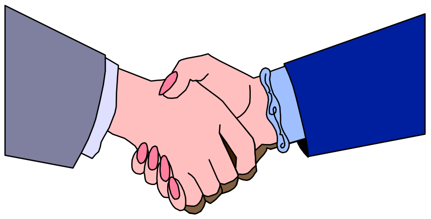
- Project and release Scopes
- Standards
- Test plans and scripts
- Bitmaps and graphics
- Batch processing scripts, JCL
- External action block source, make files
- Data Definition Language
- User Exits

What is Change Management?



Methods to manage change from initiation to implementation

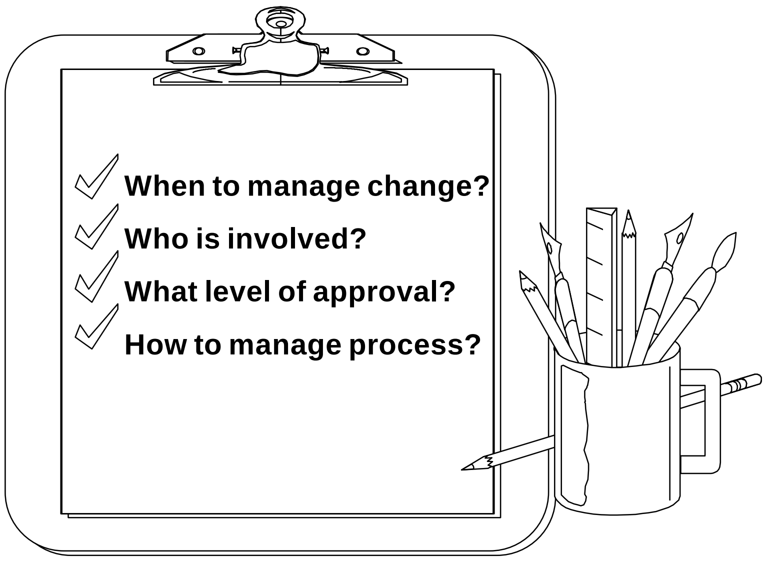
Why Care About Change Management?



Supports changing business requirements while managing technical impact

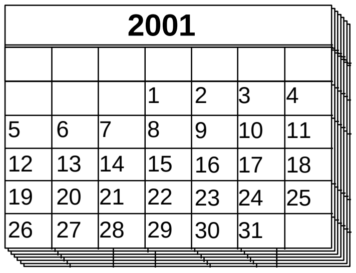
- Provides mechanism to:
  - Assess impact
  - Analyze costs and benefits
  - Prioritize
  - Approve
  - Communicate
  - Implement

Change Management Issues



- ✓ When to manage change?
- ✓ Who is involved?
- ✓ What level of approval?
- ✓ How to manage process?

When to Control?

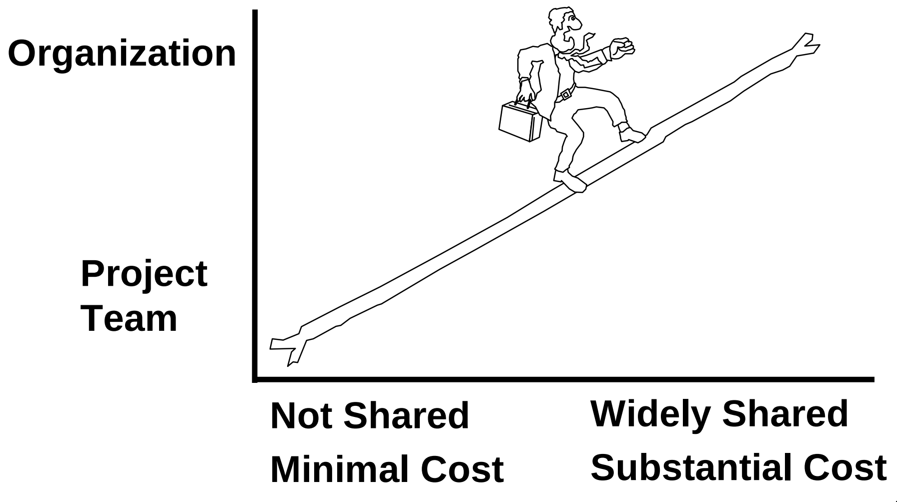


2001						
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

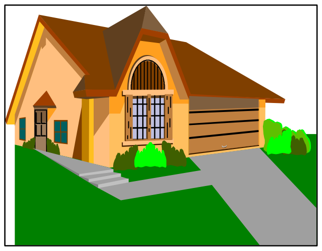
- During development and project test
  - – Reused and common objects
  - – Approved or “frozen” objects
  - – Objects with variants in test and production
- *All* objects in test and production

Who is Involved?

What Level of Approval?

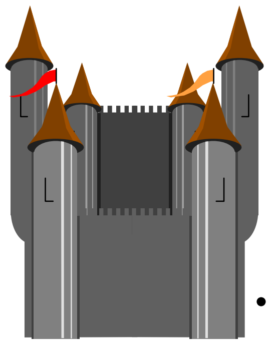


How to Manage at Project Level?



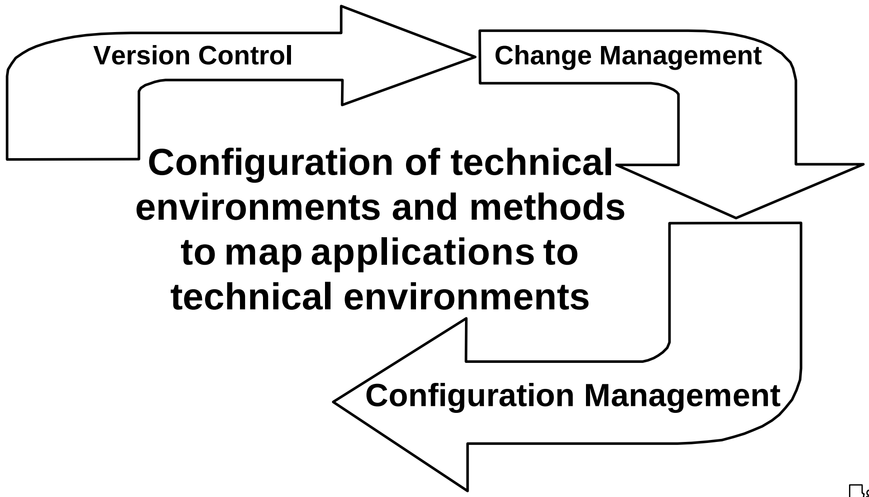
- Process
  - Requirements and impact are documented by team member
  - Project Manager approves
  - Communicate at team meeting
  - Implement change
- Technique
  - Prevent inadvertent and unapproved changes with “protective” subsetting

How to Manage at Organizational Level?

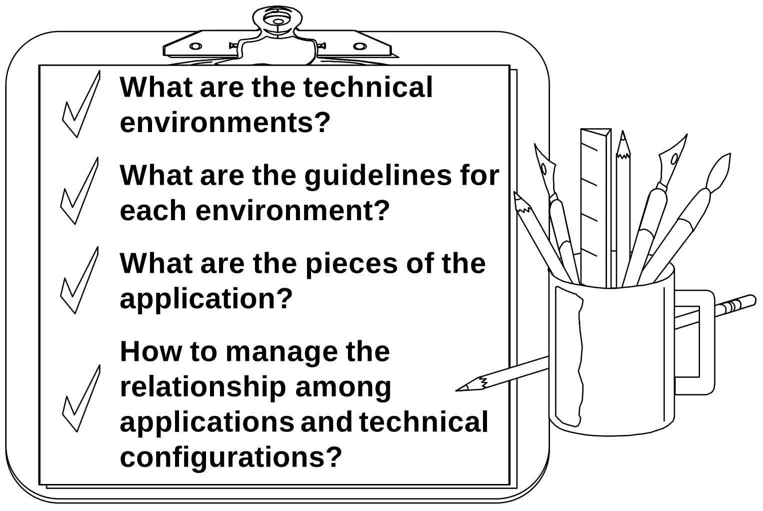


- Process
  - Change initiated by any interested party
  - Assess impact, cost, and benefits
  - Approve by consensus or authority
  - Complete by owner, test, and approve
  - Associate non-priority change with non-priority release; schedule new release for priority changes
  - Propagate change to impacted parties
- Technique
  - Prevent unapproved change to strategic, reused, and common objects by establishing centralized models

What is Configuration Management?


Version Control
Change Management
Configuration of technical environments and methods to map applications to technical environments
Configuration Management

Configuration Management Issues



- What are the technical environments?
- What are the guidelines for each environment?
- What are the pieces of the application?
- How to manage the relationship among applications and technical configurations?

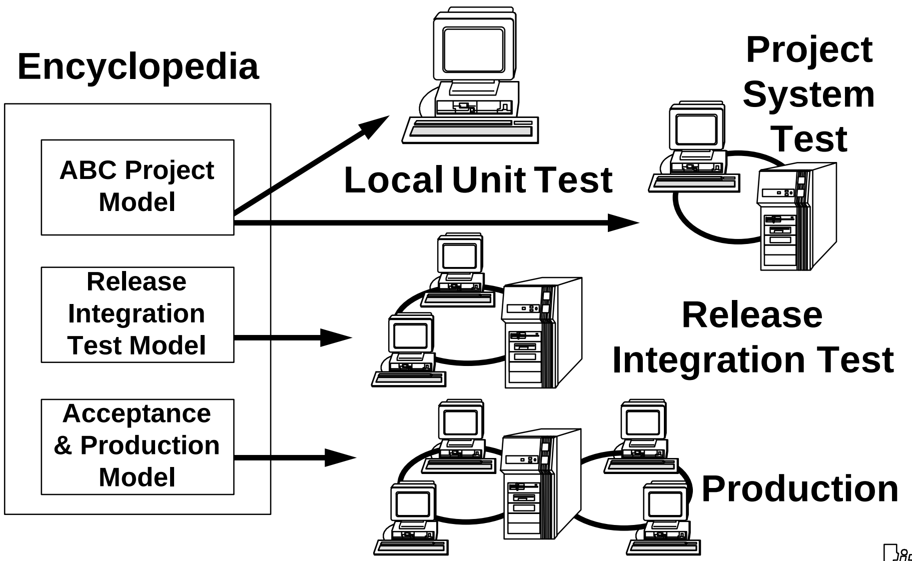
What are the Technical Environments?

- Factors
  - Organizational requirements for Configuration Management
  - Production release integration and scheduling
  - Availability of resources (technical and human)
- Examples
  - Development and Unit Test
  - Project System Test
  - Release Integration Test
  - Production
  - Production Support

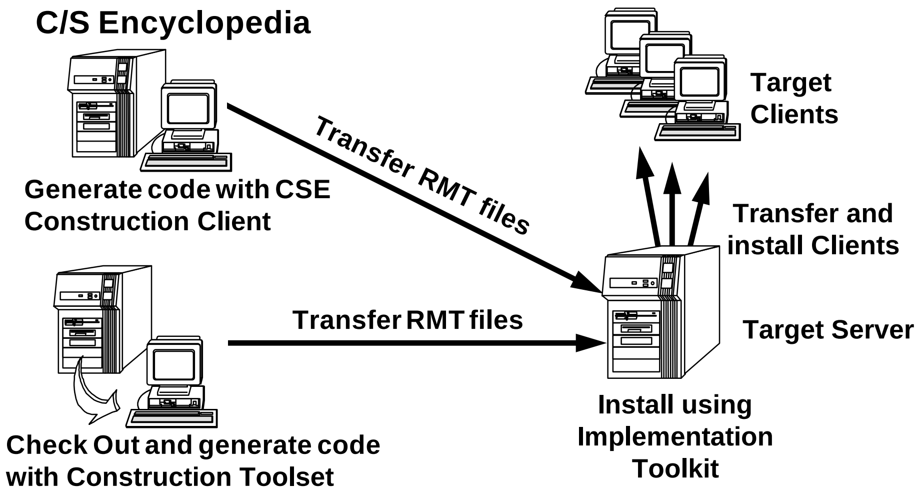
What are the Guidelines for Environments?

- Purpose of the environment
- Security requirements and procedures to obtain access
- Names of target environment objects (e.g., databases, servers, libraries)
- Name(s) of related Composer Model(s)
- Naming conventions
- Requirements to move into environment
- Procedures to move into environment
- Roles and responsibilities

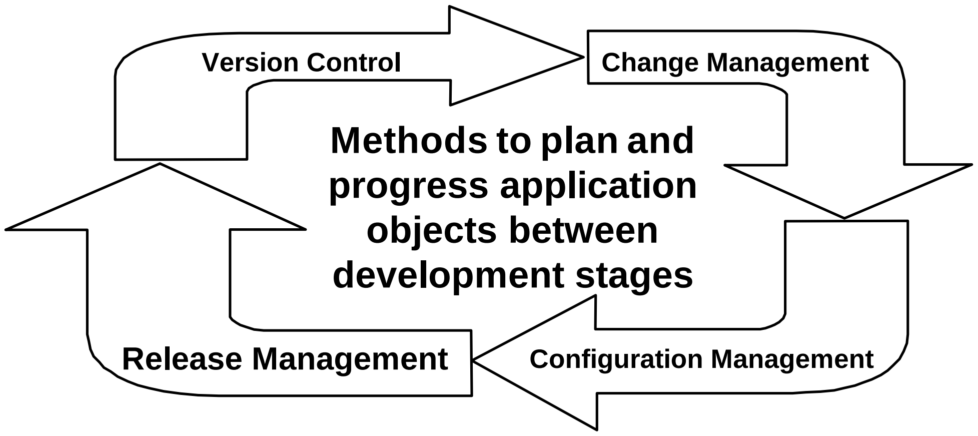
How to Manage the Relationship?



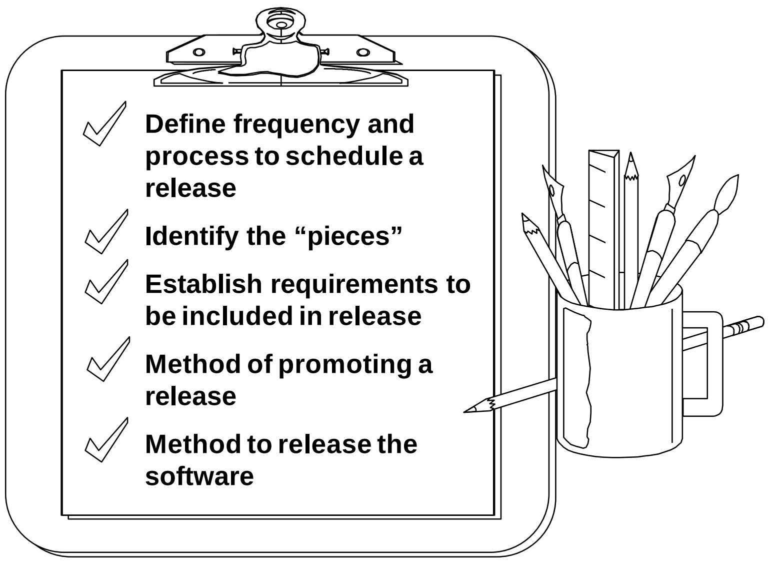
How to Generate and Install Runtime Executables?


C/S Encyclopedia
Generate code with CSE Construction Client
Transfer RMT files
Check Out and generate code with Construction Toolset
Transfer RMT files
Target Clients
Transfer and install Clients
Target Server
Install using Implementation Toolkit

What is Release Management?

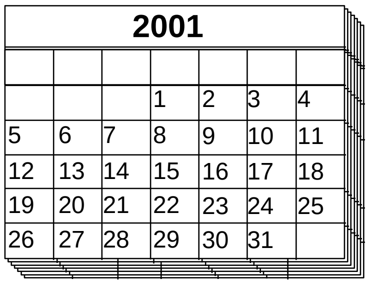

Version Control
Change Management
Configuration Management
Release Management
Methods to plan and progress application objects between development stages

Release Management



- Define frequency and process to schedule a release
- Identify the "pieces"
- Establish requirements to be included in release
- Method of promoting a release
- Method to release the software

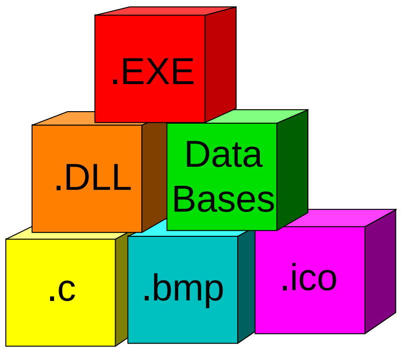
Schedule the Release



2001						
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

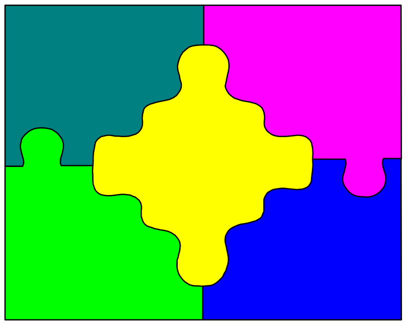
- Optimal frequency of annual releases balances resource costs while supporting business requirements
- Schedule environment usage and support staff involvement for each target environment
- Prevent multiple releases testing concurrently in same environment

Identify the Pieces



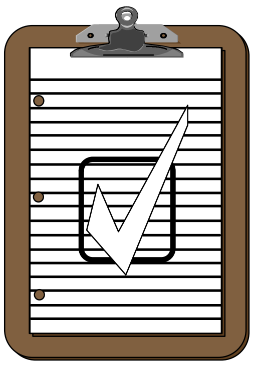
- Review the application and identify all objects (Composer and external) of the production release
- Include the hardware and software configurations

Requirements for Unit Test



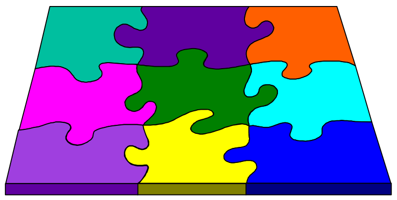
- Stable data model
- DDL generated
- Tables
- Test data
- User access defined
- Composer RI triggers

Procedures for Unit Test



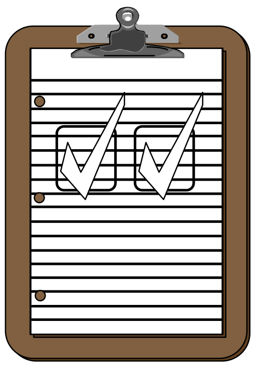
- Move external objects to local test environment
- Create external action block library
- Package for client server
- Generate and install applications
- Composer Test Facility

Requirements for Integration Test



- Stable Data Model
- DDL generated
- Tables
  - Validate storage, tablespace mapping
- Integration test data
- Limit user success
- Composer RI triggers for complete data model
- External objects baselined from unit test

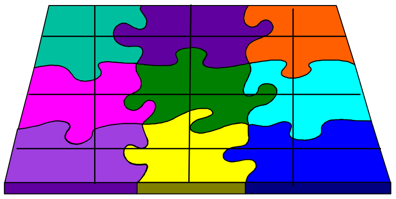
Integration Test Procedures



- Move external objects to integration test area
- Create external action block library in target environment
- Verify packaging
- Generate and install application
- Test applications in target environment

Requirements for Production

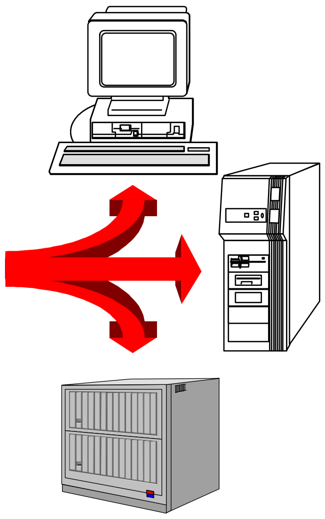
- Complete, consistent Data Model
- DDL generated
- Tables configured for production
- Convert/load production data
- Establish production user access
- Composer RI triggers
- External objects baselined from integration test



Production Release Procedures

- Define controlled area on application server
- Move external objects to production environment
- Complete external action block library
- Generate and install applications
- Deploy applications to client workstations
- Validate end user access

Release the Software



- For file-server-based software, “push” or “pull” the software to the remote locations through the LAN
- For workstation-based software, stage the software to a file server and download to the workstations
- Update the server environment by deploying applications to the production area

Composer Support for Configuration Management

- Difference Analysis – Determine the level of change between releases
  - Compare reports
  - Object cross-reference reports
  - Public interface queries
  - Migration and adoption functionality
  - Manual process for external objects

Automating Configuration Management

- Promotion of generated applications
  - Traditional CM tools
- Synchronization of Composer objects to generated applications
  - Queries to compare object/load module timestamps
  - Construction Impact Analysis reports
  - Installation data from target configuration databases on server Implementation Toolsets

Summary

- Identify and define technical environments
- Manage the relationship between Composer models and target environments
- Identify, track, and control Composer and external objects
- Plan and identify scope for a production release
- Define methods to promote a production release and roll out software releases

Configuration Management for Client/Server Applications

Session 540

Rebecca Lawson and Adel Harris
Texas Instruments