



Test Project Management : CoDeC

Ajikumar TN
Ganesh Arunachalam

Wipro Technologies

Agenda

- Test Life Cycle
 - Methodologies
 - CoDeC
- System Complexity Estimation SCE
- System Change Impact Matrix SCIM
- Design Structure Matrix DSM
- Integrated Tool – CoDeC

TEST LIFE CYCLE



Sl. No.	Test Phase
1	Test Requirement Gathering
2	System Study and Effort Estimation
3	Test Sequencing
4	Test Suite Optimization and Creation
5	Resource Leveling and Scheduling
6	Test Execution
7	Defect Tracking and Closure
8	Reliability Estimation and Release
9	Maintenance Testing

TEST PHASES Vs METHODOLOGIES

Sl. No.	Test Phase	Techniques / Tools
1	Test Requirement Gathering	Eg. Voice of Customer (VoC)
2	System Study and Effort Estimation	Eg. System Complexity Estimator (SCE)
		Eg. Analytical Hierarchical Program (AHP)
3	Test Sequencing	Eg. Dependency Structure Matrix (DSM)
4	Test Suite Optimization and Creation	Eg. Orthogonal Array (OA)
5	Resource Leveling and Scheduling	Resource Leveling
6	Test Execution	Testing by Suite
		Testing by Exploration
		Testing by Pairs
		Test Execution Tracker
7	Defect Tracking and Closure	Defect Tracker
8	Reliability Estimation and Release	Reliability Estimation
9	Maintenance Testing	Eg. System Change Impact Matrix (SCIM)

Wipro's ROBUST TESTING METHODOLOGY

CoDeC

- CoDeC is a combination of 3 methodologies used for
 - Effort estimation / distribution
 - Test Sequencing
 - System / Factor Impact Analysis

- SCE
 - Effort estimation / distribution
- DSM
 - Test Sequencing
- SCIM
 - Change Impact Analysis

- These methodologies will assist the Project Manager in proper decision making in each of the applicable phases.

System Complexity Estimation SCE

- Agenda
 - Challenge addressed
 - Tool Structure
 - Feature Summary
 - Tool Flow
 - Exaple Input
 - Example Output

CHALLENGES Addressed

- We have a product or a system which has different modules or applications interlinked.
- How to distribute testing effort across these modules and / or applications when testing the product or the system

TOOL Structure

- MAIN SCREEN
 - Create Skeleton Input Sheet
 - Estimate System Complexity

The Tool analyses

- System inter-dependencies

And Gives the output

- System Complexity

FEATURE SUMMARY – Outputs

Sl. No.	Features
1	System Complexity
2	Application Dependency on System' Index (ADSI)
3	System Dependency on Application' Index (SDAI)
4	Application Contributon to System Complexity % (ACSC)
5	Application Complexity (AC)

% ACSC helps us in determining the distribution of testing effort across different modules

TOOL Flow

- Create Skeleton Input Sheet
 - Input no. of modules
 - Give file name to be saved as
- Enter Inputs
 - Open input file
 - Give the interdependency information
 - Save the file
- Estimate System Complexity
 - Give the input file
 - Give the output file name to be saved as
 - The tool will analyse the input sheet and save the output file

AN EXAMPLE – Output



System Complexity Estimator (SCE)

No of Modules	6
System Complexity	11.3

Module Name	Module No.	ADSI (%)	SDAI (%)	ACSC (%)	AC
	1	13.9	17.1	15.5	1.7
	2	13.9	16.7	15.3	1.7
	3	11.1	22.2	16.7	1.9
	4	21.3	22.6	22	2.5
	5	17.6	13.1	15.3	1.7
	6	22.2	8.3	15.3	1.7

ADSI	Application Dependency on System Index
SDAI	System Dependency on Application Index
ACSC	Application Contribution to System Complexity
AC	Application Contribution

SCE Demo



System Change Impact Matrix SCIM

- Agenda
 - Challenge Addressed
 - Tool Structure
 - Feature Summary
 - Tool Flow
 - Example Input
 - Example Output

CHALLENGES Addressed

- A complex product/system with several of its applications/modules interlinked.
- How do we determine the impact of change on the system due to any new requirements / field reports

TOOL Structure

- MAIN SCREEN
 - Create Skeleton Input Sheet
 - Estimate System Change Impact

The Tool analyses

- Dependencies between Modules and Factors

And Gives the output

- Module Impact Analysis
- Factor Impact Analysis

FEATURE SUMMARY – Outputs



Sl. No.	Features
1	Module Impact Analysis Report
2	Factor Impact Analysis Report

Module Impact Analysis report : Helps in distributing the testing effort effectively between the Modules.

Factor Impact Analysis report : Helps in distributing the testing effort effectively between the Factors.

TOOL Flow

- Create Skeleton Input Sheet
 - Input no. of modules
 - Input no. of factors
 - Give file name to be saved as
- Enter Inputs
 - Open input file
 - Give
 - ◆ the SCE OUTput file information of the system
 - ◆ the impact information bwteen Modules and Factors
 - Save the file
- Estimate System Change Impact
 - Give the input file
 - Give the output file name to be saved as
 - The tool will analyse the input sheets and save the output file

AN EXAMPLE – Output

SCIM Tool

PROJECT NAME	WE
PRACTITIONER	Ajikumar TN

No. of Modules	6
No. of Factors	4
System Complexity	25.3
Change Impact Inde	19.42

MII	Module Impact Index
FII	Factor Impact Index

Module N	Module N	%MII	MII
a	1	8.91	2.26
b	2	25	6.32
c	3	16.9	4.27
d	4	20.41	5.16
e	5	24.19	6.12
f	6	4.59	1.16

Factor Na	Factor No	%FII	FII
a1	1	33.33	8.43
a2	2	16.67	4.22
a3	3	33.33	8.43
a4	4	16.67	4.22

SCIM Demo



Design Structure Matrix DSM

- Agenda
 - Challenge Addressed
 - Tool Structure
 - Feature Summary
 - Tool Flow
 - Example Input
 - Example Output

CHALLENGES Addressed

- In the normal project scenario, we need to deal with product/system with several of its applications/modules interlinked and inter-dependant.
- The challenge is to decide which module to be taken for testing first and which one next etc.
- Another challenge is to decide which modules can be tested in parallel

TOOL Structure

- MAIN SCREEN
 - Create Skeleton Input Sheet
 - Execute Analysis

The Tool analyses

- Inter-dependencies between Modules

And Gives the output

- AEAP
 - Cyclic Blocks
 - Levels
 - Partition Table
- ALAP
 - Cyclic Blocks
 - Levels
 - Partition Table

FEATURE SUMMARY – Outputs

Sl. No.	Features
1	Cyclic Blocks
2	Levels
3	Partition Table - Compressed
4	Partition Table - Expanded

Cyclic Blocks: Those modules involved in a cyclic dependency is reported separately for the project managers to treat them differently.

Levels: Each of the Modules / Blocks will be bucketized into different Levels and reported. The modules / blocks in level 1 will be taken up for execution before those in Level 2.

Partition Table (Compressed): This table is created to show the dependency sequence between Modules / Blocks (in the post-analysis sequence).

Partition Table (Expanded): This table is created to show the dependency sequence after all the blocks are expanded to show the involved modules. This is a replica of the input information given in the 'input file', but in the post-analysis sequence.

AN EXAMPLE – Input



Design Structure Matrix (DSM) Tool

PROJECT NAME	WE
PRACTITIONER	albert test

Add Module	Add Dependency
Delete Module	Delete Dependency

No Dependency	0 or " "
Dependency	1

	Module Name	Module No.	1	2	3	4	5	6
	a	1	1	1				
	b	2		2	1			
	c	3	1		3		1	
	d	4				4	1	1
	e	5					5	
	f	6						6

AN EXAMPLE – Output

DSM OUTPUT - AEAP As Early As Possible

PROJECT NAME	WE	
PRACTITIONER	albert test	

CYCLIC BLOCKS			
Blocks	Modules	list	-->>>
301	2	1	3

LEVELING			
Level	Modules /	Blocks	
1	5	6	
2	301	4	

PARTITIONING (Compressed)							
Levels	Modules /	5	6	301	4		
1	5	0	0	0	0		
1	6	0	0	0	0		
2	301	1	0	0	0		
2	4	1	1	0	0		

PARTITIONING (Expanded)									
Levels	Block Nos	Module N	Module N	5	6	2	1	3	4
1		e	5	0	0	0	0	0	0
1		f	6	0	0	0	0	0	0
2	301	b	2	0	0	0	0	1	0
2	301	a	1	0	0	1	0	0	0
2	301	c	3	1	0	0	1	0	0
2		d	4	1	1	0	0	0	0

DSM Demo



CoDeC Tool

- CoDeC Tool is an integrated version of SCE, DSM and SCIM
- CoDeC : **C**omplexity, **D**ependency, **C**hange impact

- There are 2 versions of CoDeC
 - Module
 - Activity

- It gives all the output reports similar to SCE, DSM and SCIM
- Extra Outputs
 - Value Thread information
 - ♦ CoDeC will report all the threads of sequence of execution possible
 - ♦ CoDeC-Activity version will analyse the time information available as well

CoDeC OUTPUT – Value Thread Info

Value Thread

CYCLIC BLOCKS			
Blocks	Modules	list -->>>	
301	2	1	3

VALUE THREAD (INFORMATION FLOW)					
Threads					
1	S	7	5	301	E
2	S	8	6	4	E
3	S	9	E		
4	S	10	E		
5	S	11	E		
6	S	12	E		

Value Thread-1 above shows that the execution thread is 'Module 7 -> Module 5 -> Cyclic Block 301'

CoDeC Demo



Thank You