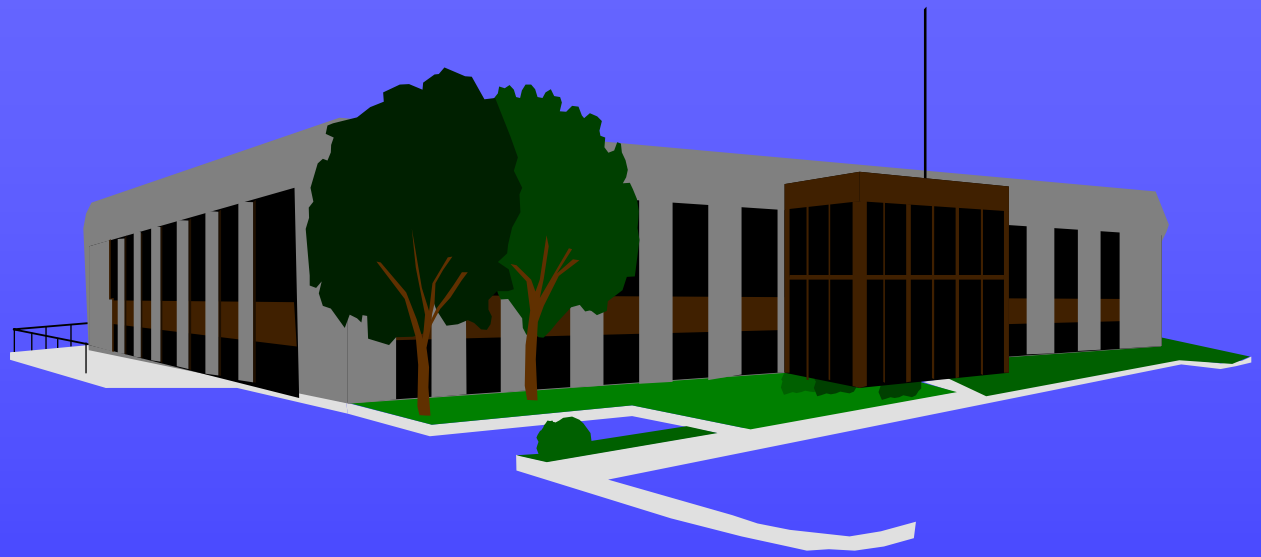
テストシナリオアプローチ
変更影響を分析する



Dr. Suhaimi Ibrahim
Centre For Adv. Soft. Engineering
UTM, City Campus,
Kuala Lumpur

目次

- 回帰テストとは?
- テストケースの選択
- ローケーションというコンセプトの必要性
- 要求のトレース
- 探針（Reconnaissance）アプローチ?
- インストルメントプロセス
- コンパイルプロセス
- テストシナリオ
- 解析
- ケーススタディ – OBA プロジェクト
- 結果と議論
- CATIA アプリケーション
- 結論と今後の計画

What is Regression Testing?

- 回帰テストとは一種のプロセスです。変更が既存のソフトウェアシステムに影響することがないか確認をします。
- 回帰テストは品質管理測定も行います。変更したコードがコンパイルできるか、特定された要求や変更されたいな部分になんら影響ないかを確認する、メインテナンス作業でもあります
- 一般的な回帰テストは、一つ前のバージョンでのテスト結果と修正を入れられたコードが正しくマージされているか確認する

回帰テストとは?

- システムが変更によって影響をうけてないか確認する。理論的には各々のコード修正に対して全てのテストケースを走らせる。実際にはこのような回帰テストは非常にコストのかかるものである。

- 回帰テストは一度だけテストを成功させるという意義より、再テストであり、受け入れ標準である。各々のイテレーションにおいて既存のテストケースの回帰テストを走らせ、そして新しいテスト結果と今までのテスト結果を比較する。

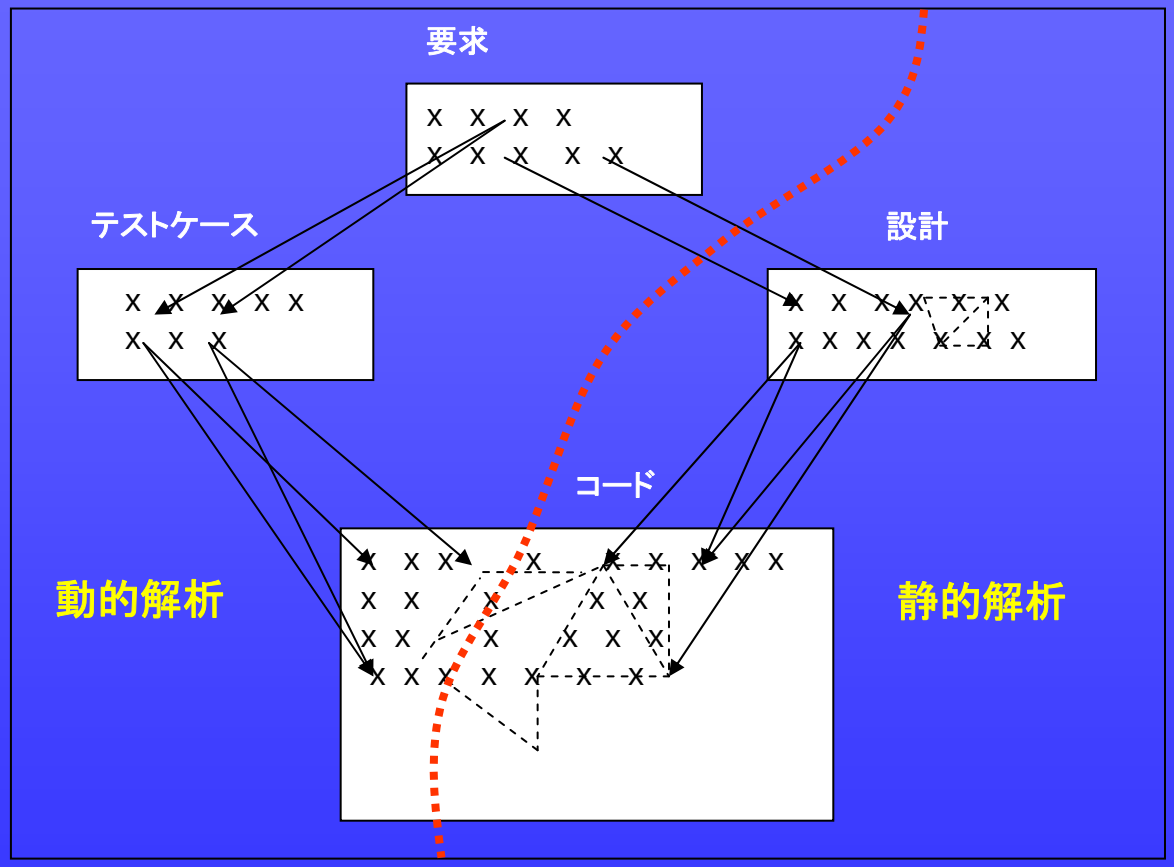
テストケースの選択

- 全てのテストケースの実行のコスト高と、作業時間増加
- 既存のプログラム機能を確認するため、変更影響に対するテストケースだけが実行される
- 最重要テストケースの選択
- 選択テストケースの削減可能性
- どのようにテストケースが選択されるか….

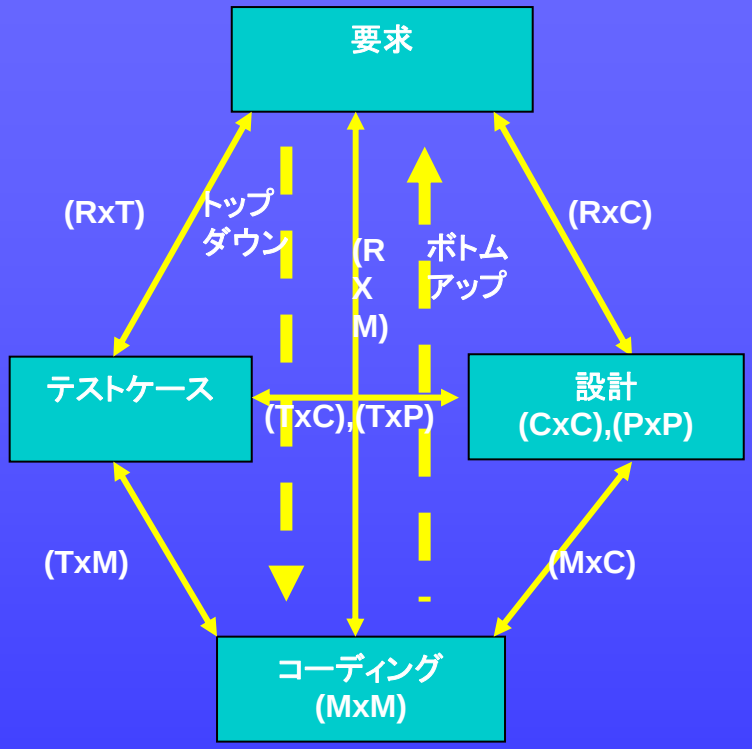
ローケーションというコンセプトの必要性

- ローケーションと言うコンセプトはシステムの中のプロセスの場所をあきらかにする。例えば年齢、年収、購買履歴、クレジットカードシステム、変更要求、要件、テストケース
- 例としては、各変更要求に対して、実際の変更が入る前に変更影響を推定する。
- CCBでの決定は開発者によるコンサルタントが必要になる。
- ある種の自動化ツールが必要になる
- 既存のツールやCASEはソフトウェア開発やその統合作業やコーディネーションに注力している
- ある意味で偵察的テクニックが必要となる ……

要求のトレーサビリティ



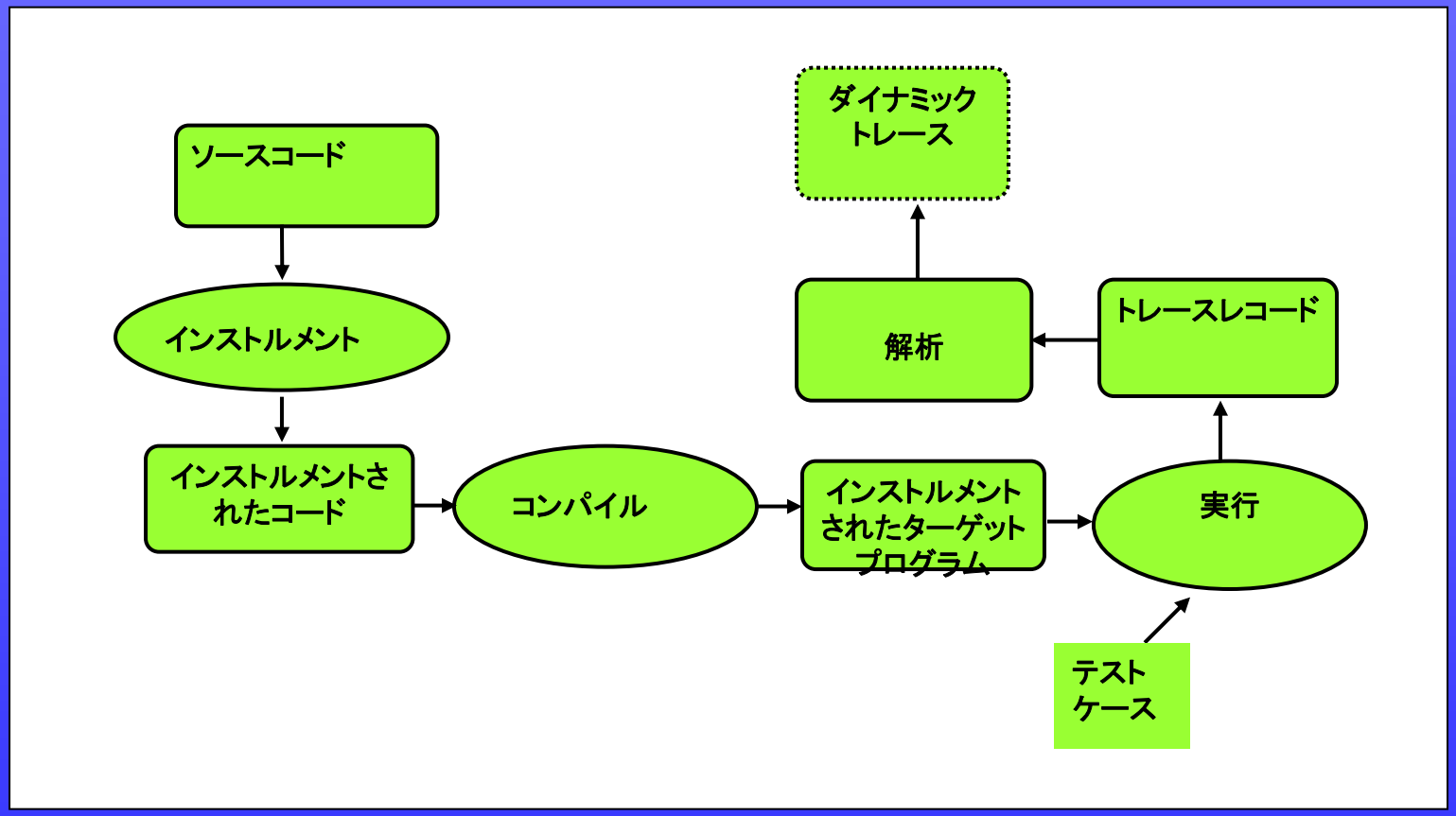
要求のトレース


要求
(RxT)
トップ
ダウン
(R
x
M)
ボトム
アップ
(RxC)
テストケース
(TxC),(TxP)
設計
(CxC),(PxP)
(TxM)
(MxC)
コーディング
(MxM)

探測的アプローチとは?

- テストケースからの動的解析
- インパクトコードのキャプチャー
- インパクトコードの分析
- テストケースと影響モジュールトレーサビリティ
- すべてのプログラム変更はテストケースメインテナンスのためにトレースされる

What is Reconnaissance Approach?



インストルメントのプロセス

- 開発したツールを利用してのインストルメント。*CodeMentor* はいくつまのマーカーをソースコードの中に埋め込む
  - ファンクション/クラスのスタート - e スイッチはファンクション、クラス（main()を含む）エントリーポイントをイントルメントする。
  - リターン - r スイッチは暗示的もしくは明示的にRETURN させるインストメントを埋め込む。
  - 修了 - x スイッチはEXITステートメントを埋め込む
  - 状態 - d スイッチはディシジョンステートを埋め込む

インストルメントコード

```
/* FUNCTION PURPOSE : CCruiseInfo constructor                              */
/*                                                                         */
/* PARAMETER                : None                                         */
/*                                                                         */
/* RETURN VALUE             : None                                         */
/*************************************************************************/

CCruiseInfo::CCruiseInfo()
{
fputs("\n#TRACE# I am Inside Function *CCruiseInfo CCruiseInfo()* Line=55",dst_file1);
fprintf(dst_file1,"^ %d ^",++count_exe);
            bActive = false;
            bAcceleration = false;
            bAccelerationPedal = false;
          if(countIFTrue1 > 0)
           {
fputs("\n#TRACE# If statement Line : 59",dst_file1);
               ++countIFTrue1;
           }
            bResume = false;
            bSuspend = false;
            dPrevCruiseSpeed = 0;
            throttle->controlThrottle(throttleCommand, throttlePosition, 0);
            displayMsg->displayCommonMsg("--------------------"); //display
            led->lightLed(ledShm,1,false);
            led->lightLed(ledShm,2,false);
            led->lightLed(ledShm,3,false);

fputs("\n#TRACE# END OF Function CCruiseInfo CCruiseInfo() Line=67  ",dst_file1);
fprintf(dst_file1,"^ %d ^",++count_exe);
}

/*******************  END OF FUNCTION  ************************/
```

Entry Instrumentation

Decision (IF) Instrumentation

Return Instrumentation

コンパイルプロセス

- インストルメントコードのコンパイル
- ターゲットプログラム用にコード生成
- テストシナリオ用の準備

テストシナリオ

- テストシナリオはテストケースに基づく実行プログラムである
- 各要求は1つかそれ以上のテストケースによってカバーされている
- 各々のテストケースはインパクトコードによって生成され実行される
- インパクトコードコンポーネントは分析のために保存される

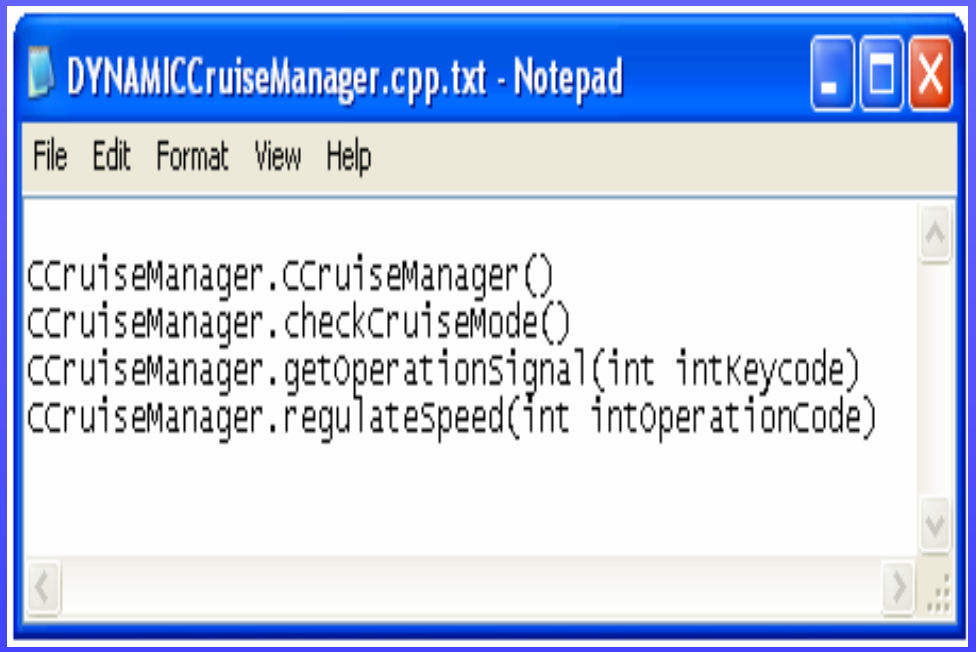
トレースのスナップショット

```
#TRACE# I am Inside Function *CCruiseInfo.CCruiseInfo()* Line=55^ 1 ^
#TRACE# END OF Function CCruiseInfo.CCruiseInfo() Line=67  ^ 2 ^
#TRACE# I am Inside Function *CCruiseInfo.isActive()* Line=99^ 3 ^
#TRACE# exit from Function CCruiseInfo.isActive() by return( ) statement Line=100 ^ 4
^
#TRACE# I am Inside Function *CCruiseInfo.isActive()* Line=99^ 5 ^
#TRACE# exit from Function CCruiseInfo.isActive() by return( ) statement Line=100 ^ 6
^
#TRACE# I am Inside Function *CCruiseInfo.isActive()* Line=99^ 7 ^
#TRACE# exit from Function CCruiseInfo.isActive() by return( ) statement Line=100 ^ 8
^
#TRACE# I am Inside Function *CCruiseInfo.isActive()* Line=99^ 9 ^
#TRACE# exit from Function CCruiseInfo.isActive() by return( ) statement Line=100 ^
10 ^
#TRACE# I am Inside Function *CCruiseInfo.isActive()* Line=99^ 11 ^
#TRACE# exit from Function CCruiseInfo.isActive() by return( ) statement Line=100 ^
12 ^
#TRACE# I am Inside Function *CCruiseInfo.isActive()* Line=99^ 13 ^
#TRACE# exit from Function CCruiseInfo.isActive() by return( ) statement Line=100 ^
14 ^
#TRACE# I am Inside Function *CCruiseInfo.setCruiseStatus(char operationState, bool
bStatus)* Line=133^ 15 ^
#TRACE# If statement Line : 135
#TRACE# If statement Line : 137
#TRACE# END OF Function CCruiseInfo.setCruiseStatus(char operationState, bool
bStatus) Line=182  ^ 16 ^
#TRACE# I am Inside Function *CCruiseInfo.isActive()* Line=99^ 17 ^
#TRACE# exit from Function CCruiseInfo.isActive() by return( ) statement Line=100 ^
18 ^
#TRACE# I am Inside Function *CCruiseInfo.setCruiseStatus(char operationState, bool
bStatus)* Line=133^ 19 ^
#TRACE# If statement Line : 135
#TRACE# END OF Function CCruiseInfo.setCruiseStatus(char operationState, bool
bStatus) Line=182  ^ 20 ^
```

分析

- TestAnalyzerを用いて影響のあるコンポーネントのトレース
- 同じコンポーネントの排除
- インパクトクラスやメソッドを生成

クラスとそのインパクトメソッド

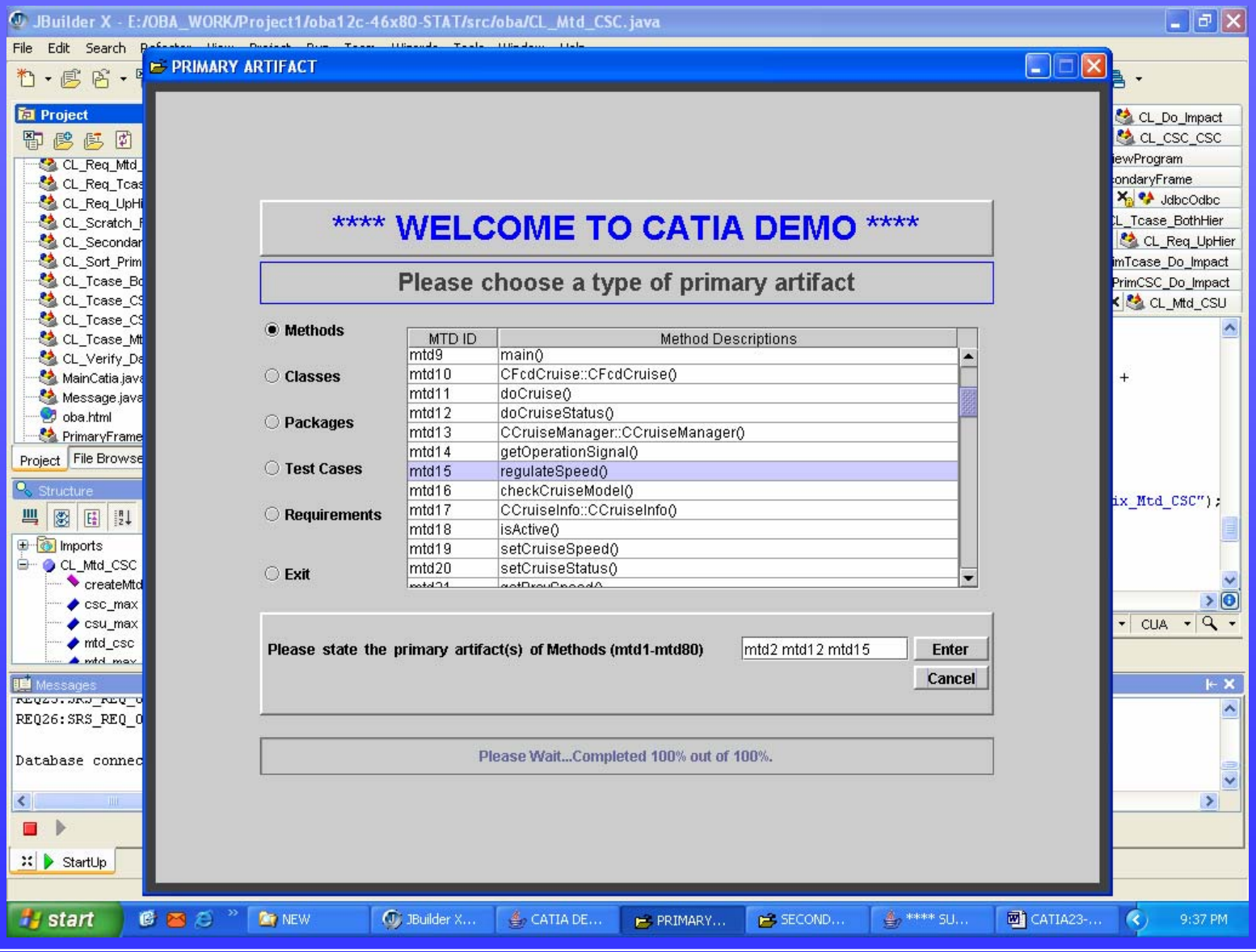


ケーススタディ
- OBA Project

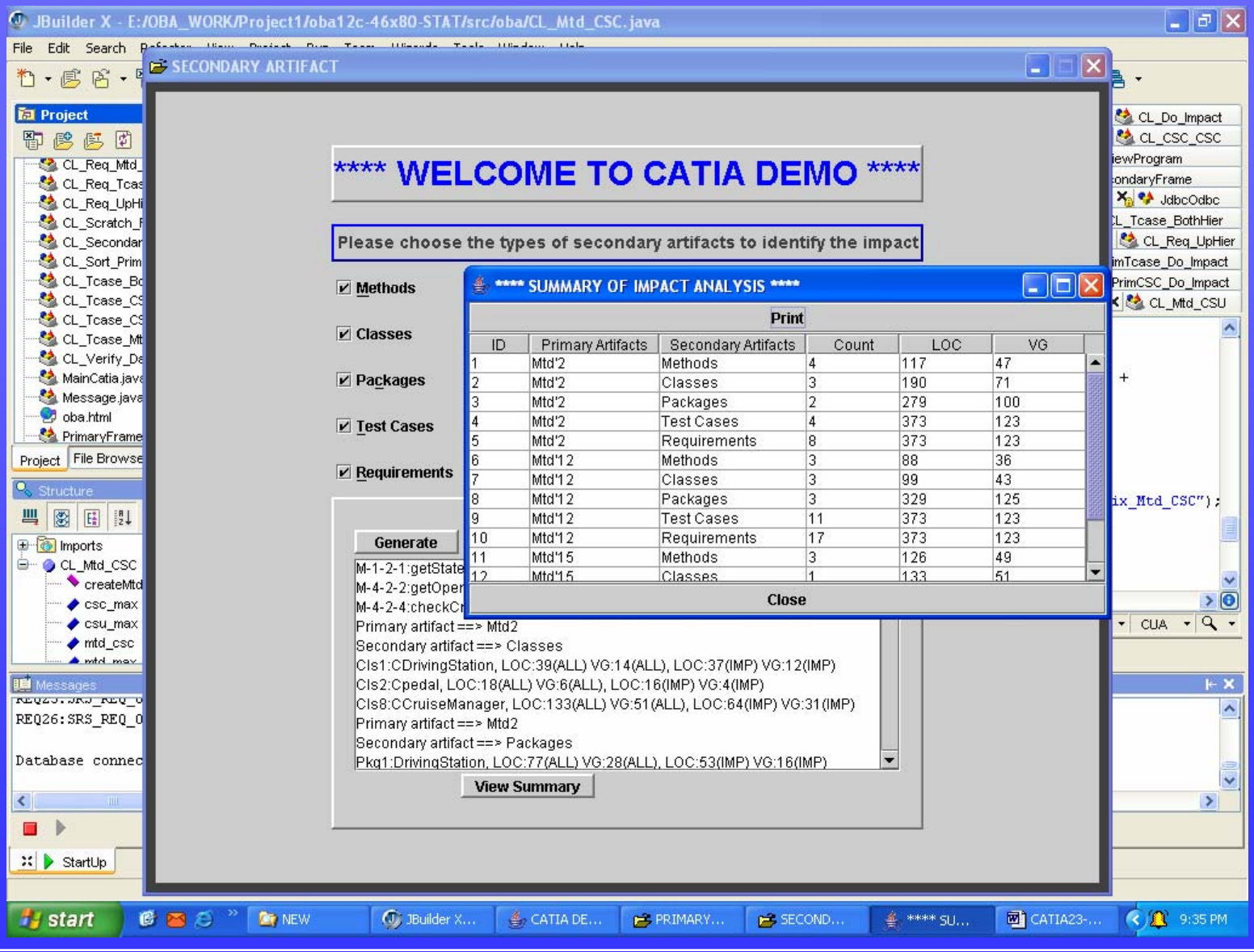
- OBAのプロジェクトは完全なプロジェクトマネジメントとDODスタンダードと MIL-STD-498に準拠した
- UML スペックと設計スタンダード
- 480ページからなるシステムドキュメント
- 4kのLOC
- OBA プロジェクトは以下の成果物から構成
  - 12 パッケージ
  - 23 クラス
  - 80 メソッド
  - 34 テストケース
  - 46 要求

	T1	T2	T3	T4	T5	T6	Tn
C1	1	1	1	1	1	1	..
C2	0	1	0	0	0	0	..
C3	1	1	1	1	1	1	..
C4	0	1	0	0	0	0	..
C5	1	1	1	1	1	1	..
C6	1	1	1	1	1	1	..
C7	1	1	1	0	0	0	..
C8	1	1	1	0	0	0	..
C9	1	1	1	0	0	0	..
C10	0	1	0	0	0	0	..
C11	1	1	1	1	1	1	..
C12	1	1	1	1	1	1	..
C13	1	1	0	0	0	0	..
C14	1	0	0	1	1	1	..
C15	1	0	0	0	1	1	..
C16	1	0	0	1	0	0	..
C17	1	0	0	0	1	1	..
C18	1	0	0	1	1	0	..
C19	1	0	0	0	0	1	..
C20	1	0	0	0	0	1	..
C21	1	0	0	0	0	1	..
C22	0	1	1	1	1	1	..
C23	1	1	1	1	1	1	..

CATIA ツール


JBuilder X - E:/OBA_WORK/Project1/oba12c-46x80-STAT/src/oba/CL_Mtd_CSC.java
File Edit Search
PRIMARY ARTIFACT
**** WELCOME TO CATIA DEMO ****
Please choose a type of primary artifact
Methods
Classes
Packages
Test Cases
Requirements
Exit
MTD ID
Method Descriptions
mtd9 main()
mtd10 CFcdCruise::CFcdCruise()
mtd11 doCruise()
mtd12 doCruiseStatus()
mtd13 CCruiseManager::CCruiseManager()
mtd14 getOperationSignal()
mtd15 regulateSpeed()
mtd16 checkCruiseModel()
mtd17 CCruiseInfo::CCruiseInfo()
mtd18 isActive()
mtd19 setCruiseSpeed()
mtd20 setCruiseStatus()
Please state the primary artifact(s) of Methods (mtd1-mtd80)
mtd2 mtd12 mtd15
Enter
Cancel
Please Wait...Completed 100% out of 100%.
start
9:37 PM

CATIA ツール


JBuilder X - E:/OBA_WORK/Project1/oba12c-46x80-STAT/src/oba/CL_Mtd_CSC.java
SECONDARY ARTIFACT
**** WELCOME TO CATIA DEMO ****
Please choose the types of secondary artifacts to identify the impact
Methods
Classes
Packages
Test Cases
Requirements
Generate
M-1-2-1:getState
M-4-2-2:getOper
M-4-2-4:checkCr
Primary artifact ==> Mtd2
Secondary artifact ==> Classes
Cls1:CDrivingStation, LOC:39(ALL) VG:14(ALL), LOC:37(IMP) VG:12(IMP)
Cls2:Cpedal, LOC:18(ALL) VG:6(ALL), LOC:16(IMP) VG:4(IMP)
Cls8:CCruiseManager, LOC:133(ALL) VG:51(ALL), LOC:64(IMP) VG:31(IMP)
Primary artifact ==> Mtd2
Secondary artifact ==> Packages
Pkg1:DrivingStation, LOC:77(ALL) VG:28(ALL), LOC:53(IMP) VG:16(IMP)
View Summary
**** SUMMARY OF IMPACT ANALYSIS ****
Print
ID | Primary Artifacts | Secondary Artifacts | Count | LOC | VG
1 | Mtd'2 | Methods | 4 | 117 | 47
2 | Mtd'2 | Classes | 3 | 190 | 71
3 | Mtd'2 | Packages | 2 | 279 | 100
4 | Mtd'2 | Test Cases | 4 | 373 | 123
5 | Mtd'2 | Requirements | 8 | 373 | 123
6 | Mtd'12 | Methods | 3 | 88 | 36
7 | Mtd'12 | Classes | 3 | 99 | 43
8 | Mtd'12 | Packages | 3 | 329 | 125
9 | Mtd'12 | Test Cases | 11 | 373 | 123
10 | Mtd'12 | Requirements | 17 | 373 | 123
11 | Mtd'15 | Methods | 3 | 126 | 49
12 | Mtd'15 | Classes | 1 | 133 | 51
Close
Project
Structure
Messages
REQ26:SRS_REQ_0
Database connec
StartUp
start
NEW
JBuilder X...
CATIA DE...
PRIMARY...
SECOND...
**** SU...
CATIA23-...
9:35 PM

結論と今後の研究

- 動的トレースの実現。テストケース、インパクトコード/機能、要求及び設計
- 回帰テストのサポート
- 変更の影響に対しての要求仕様のトレース
- 様々なニーズのサポートのため初期データの提供。例）ビジュアル、コスト予測、スケジュール変更、プログラムの理解

ありがとうございました

Email: suhaimi@citycampus.utm.my