# Dealing with SOUP using the LDRA Tool Suite

レガシーコード に代表されるSOUPに対する実践的なテストについて紹介

http://www.fuji-setsu.co.jp/files/LDRA_Dr.Dobb%27s_SOUP.pdf

# SOUP とは？

- 開発経路が未知なソフトウエア

  **Software of Unknown** (or **Uncertain**) **Pedigree** (or **Provenance**)

  再利用　派生、差分開発されるレガシーコード、グラフィックライブラリ、制御系など自動生成されるコード、
  JavaやAdaのランタイム環境、OS など

- 国際スタンダードを意図して開発されていない

- 課題：
  SOUP を管理可能な開発のベースとして
  ベストプラクティスでシステムを開発すること

- *さて、どうしましょう、、？*

# LDRA Lifecycle Automation Components

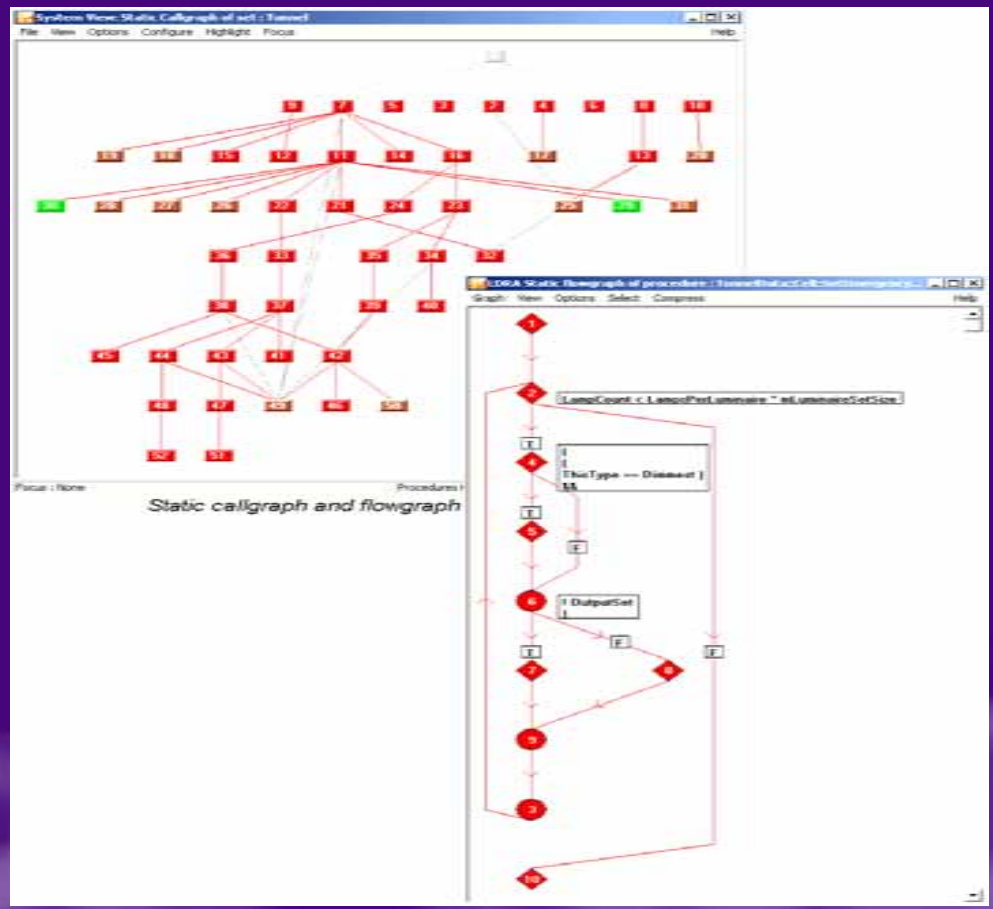

SOUP をベースにするような開発にも活用できるのでしょうか？

# LDRA ツールの実践的活用で可能になるのは、

- 既存コードを理解すること
- プログラミングスタンダードへの対応
- 機能拡張への対処
- 変更の影響を受ける箇所をテストすること
- 適切なコードカバレッジレベルの確保
- 改訂コードとSOUPが同等であることの証明
- SOUP は要件を満たしているか

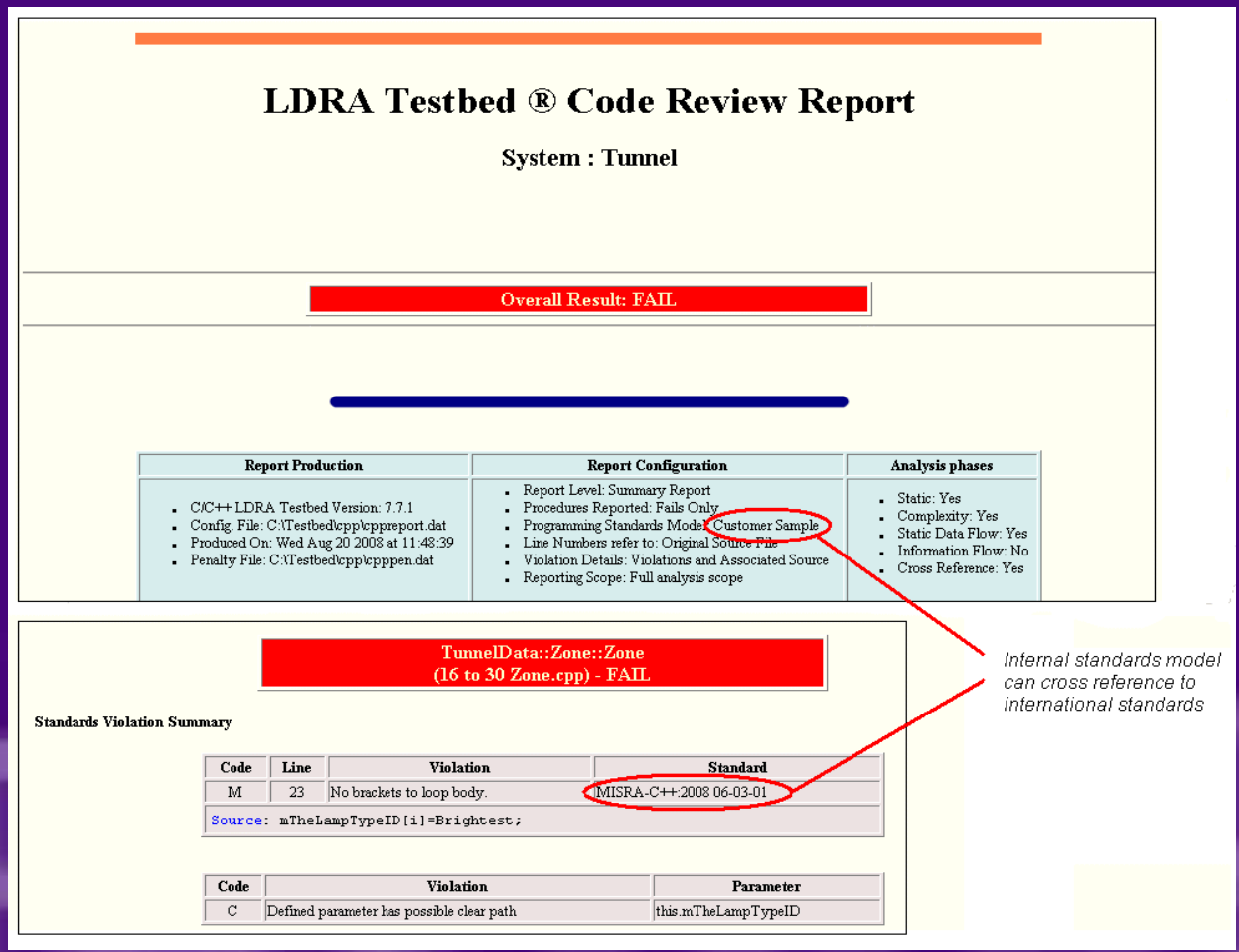
# SOUP ベース開発 LDRA ツールの実践活用

## コールグラフやフローグラフで
## 既存コードを視覚的に理解すること



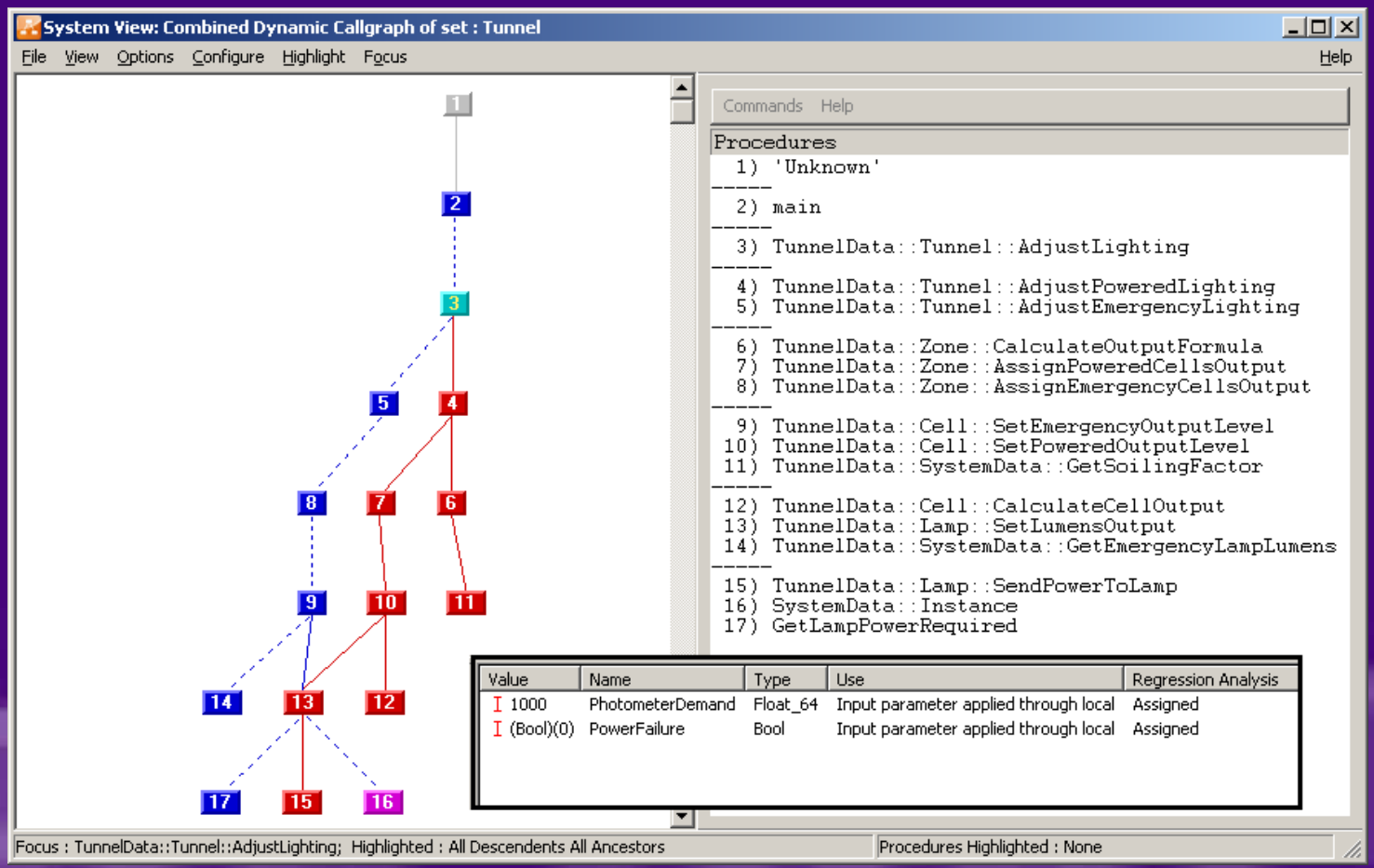
# SOUP ベース開発 LDRA ツールの実践活用

## プログラミングスタンダードへの対応には徐々に社内ルールを進化させること



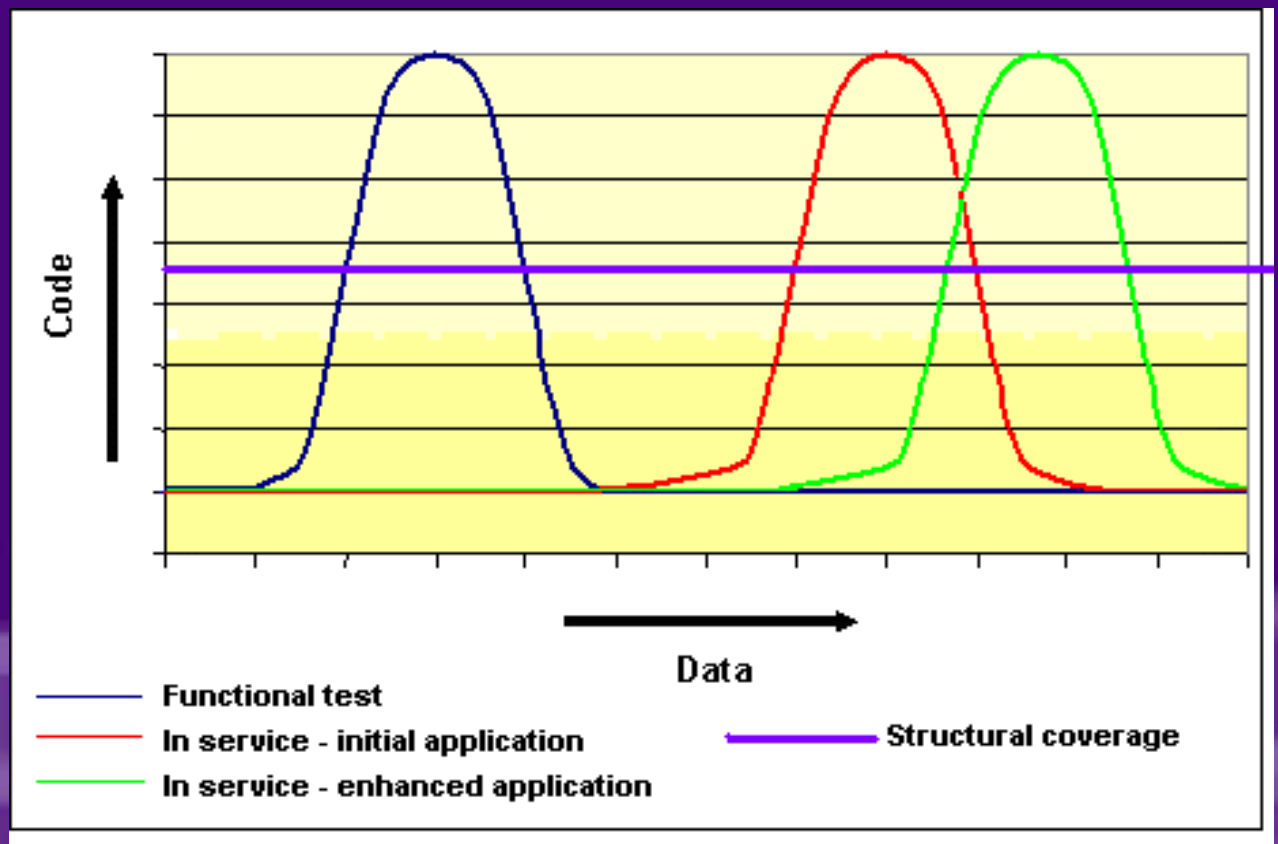
# SOUP ベース開発 LDRA ツールの実践活用

- 機能拡張への対処
- 変更の影響を受ける箇所をテストすること



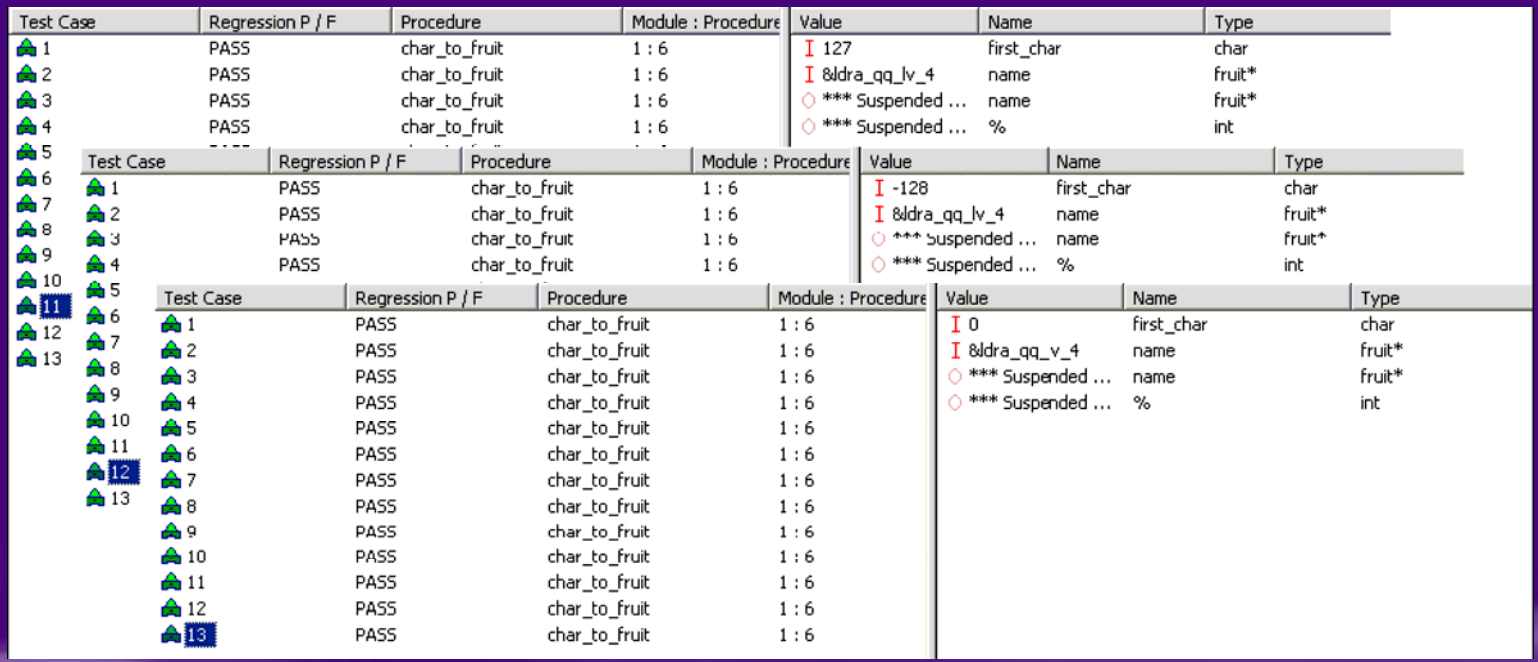
# 適切なコードカバレッジレベルの確保

構造化カバレッジを活用すれば
入力データ値で偏ることの無いテストを行える



# SOUP ベース開発 LDRA ツールの実践活用

## 改訂されたコードとSOUPが同等の機能であることの証明



| Test Case | Regression P / F | Procedure | Module : Procedure |
|---|---|---|---|
| 1 | PASS | char_to_fruit | 1 : 6 |
| 2 | PASS | char_to_fruit | 1 : 6 |
| 3 | PASS | char_to_fruit | 1 : 6 |
| 4 | PASS | char_to_fruit | 1 : 6 |

| Value | Name | Type |
|---|---|---|
| 127 | first_char | char |
| &ldra_qq_lv_4 | name | fruit* |
| *** Suspended ... | name | fruit* |
| *** Suspended ... | % | int |

| Test Case | Regression P / F | Procedure | Module : Procedure |
|---|---|---|---|
| 1 | PASS | char_to_fruit | 1 : 6 |
| 2 | PASS | char_to_fruit | 1 : 6 |
| 3 | PASS | char_to_fruit | 1 : 6 |
| 4 | PASS | char_to_fruit | 1 : 6 |

| Value | Name | Type |
|---|---|---|
| -128 | first_char | char |
| &ldra_qq_lv_4 | name | fruit* |
| *** Suspended ... | name | fruit* |
| *** Suspended ... | % | int |

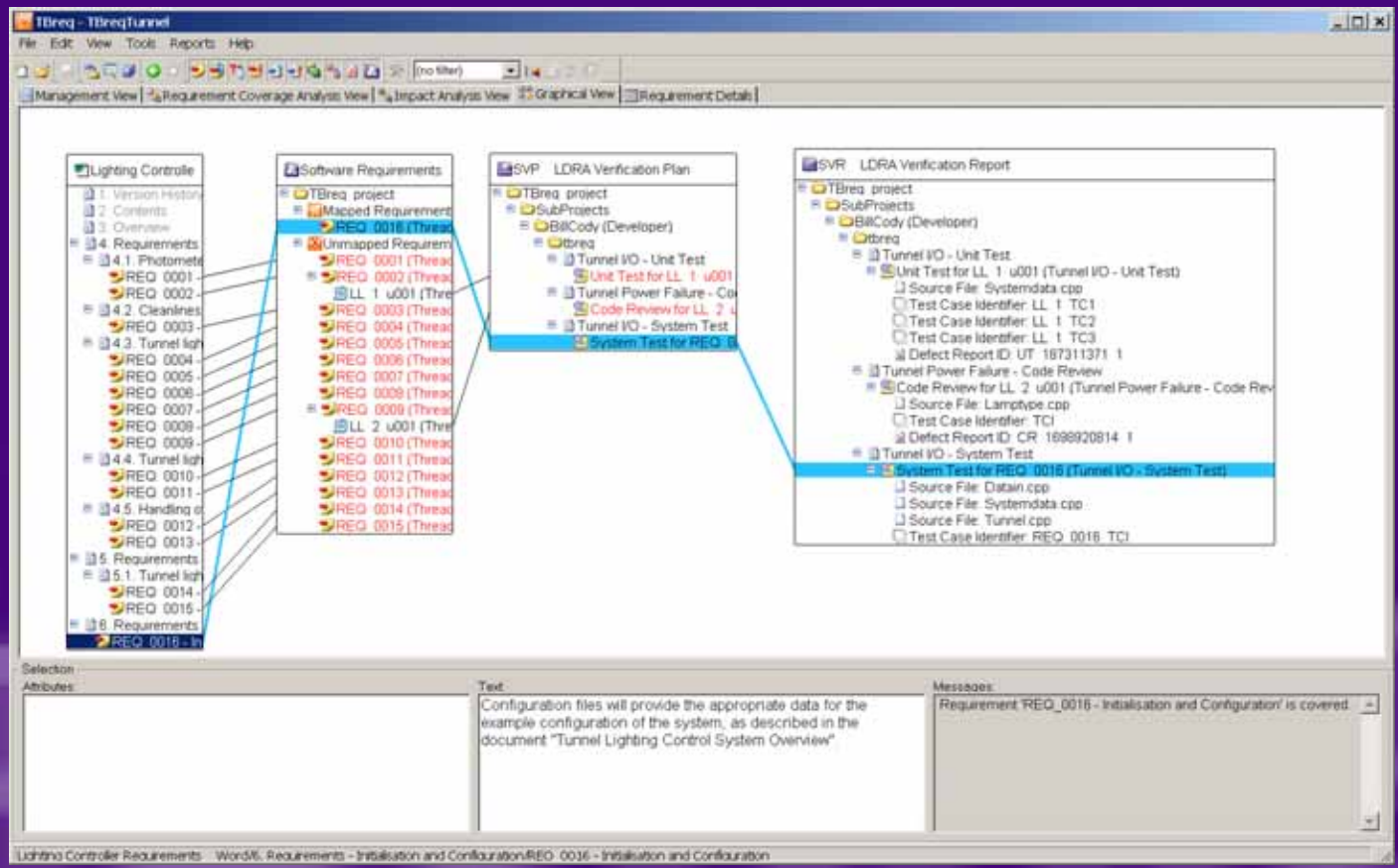
| Test Case | Regression P / F | Procedure | Module : Procedure |
|---|---|---|---|
| 1 | PASS | char_to_fruit | 1 : 6 |
| 2 | PASS | char_to_fruit | 1 : 6 |
| 3 | PASS | char_to_fruit | 1 : 6 |
| 4 | PASS | char_to_fruit | 1 : 6 |
| 5 | PASS | char_to_fruit | 1 : 6 |
| 6 | PASS | char_to_fruit | 1 : 6 |
| 7 | PASS | char_to_fruit | 1 : 6 |
| 8 | PASS | char_to_fruit | 1 : 6 |
| 9 | PASS | char_to_fruit | 1 : 6 |
| 10 | PASS | char_to_fruit | 1 : 6 |
| 11 | PASS | char_to_fruit | 1 : 6 |
| 12 | PASS | char_to_fruit | 1 : 6 |
| 13 | PASS | char_to_fruit | 1 : 6 |

| Value | Name | Type |
|---|---|---|
| 0 | first_char | char |
| &ldra_qq_v_4 | name | fruit* |
| *** Suspended ... | name | fruit* |
| *** Suspended ... | % | int |

# SOUP ベース開発 LDRA ツールの実践活用

## SOUP はどの要件を満たすべきなのか

### 要件からコード・検証結果に至るトレーサビリティ



# システムテスト

## SOUP で要件は満たされるか？

This document has been generated by TBreq

### *Software Verification Report*

**1. Table of verification requirements**

This table is based on the "SVR" document of your TBreq project (type: "LDRA Verification Report").

| Verification Task | Verification Group | Project Manager Status | Defect Report ID | Tester | Test Case Identifier | "Source File Name" (Procedure) |
|---|---|---|---|---|---|---|
| System Test for REQ_0016 (Tunnel I/O - System Test) | Tunnel I/O - System Test | Passed | | BillCody (Developer) | REQ_0016_TCI | - "Datain.cpp" (GetData)- "Systemdata.cpp" (InitialiseParams)- "Tunnel.cpp" (InitialiseTunnel) |
| Unit Test for LL_1_u001 (Tunnel I/O - Unit Test) | Tunnel I/O - Unit Test | Failed | UT_187311371_1 | BillCody (Developer) | LL_1_TC1<br>LL_1_TC2<br>LL_1_TC3 | - "Systemdata.cpp" (GetSoilingFactor) |
| Code Review for LL_2_u001 (Tunnel Power Failure - Code Review) | Tunnel Power Failure - Code Review | Failed | CR_1698920814_1 | BillCody (Developer) | TCI | - "Lamptype.cpp" (InitialiseLampType) |

# 100% の要件カバレッジ

- ハイレベル、ローレベル、派生要件を含めて
- システムの全ての機能が検証されることを裏付ける