組込みC言語のレビュー・ポイント

株式会社東陽テクニカ
ソフトウェア・ソリューション

プログラム構造に起因する問題

- プログラムの構造に起因する問題は
  - 単体テストでは、発見が難しい。
    - スタブやテストドライバとの組み合わせでは検出されないことがある。
  - システムテストでは、再現が難しい。
    - 実機、フィールドでの問題となり、手戻り、回収に発展する可能性をはらんでいる。
- プログラムの構造に着目したレビューは
  - 問題の検出に有効である。
    - 方法の確立や工数の確保が心配である。

プログラムの構造に着目したレビュー方法の確立

- リバースエンジニアリング技術を利用する
  - グローバル変数の分析
    - 関数からの書き込み、読み出しをトラッキングする。
  - フローチャートの分析
    - 変数へのアクセスについて順序を収集する。
  - 関数のコールツリーの分析
    - 関数の呼び出し順序を収集する。
- よく知られている情報から方法を確立する
  - 使いやすく、理解しやすい。
- 単純なルールで構成する
  - 形式化しやすく、レビューが容易である。

デッドリンクの検出

- グローバル変数のアクセスをグラフにすると、全てのグローバル変数は、関数からの読み出し（入力）、書き込み（出力）の線により、結ばれているはずである。
  - 入力しかないもの、出力しかないものは、デッドリンクである。

デッドリンクの検出
読み込みのみの変数

- 考えられるリスク
  - 初期化されていない可能性がある。
  - 値が変更されない変数を参照している可能性がある。
  - センサーなどのハードウェアの故障や電源投入時・リセット時に想定外の結果が格納されている可能性がある。
  - 該当のモジュールのみが流用され、付属のモジュールが流用されなかった可能性がある。
  - 流用モジュールの実行に必要な手続きが不足している可能性がある。

デッドリンクの検出
読み込みのみの変数

- レビューポイント
  - 「値が参照されているにもかかわらず、値が更新されない変数」をリストアップする。
- レビューの観点
  - 読み込みアクセスのみしか存在しないのは、意図的かどうかを検証する。
  - 初期化が必要でない理由が、明確になっているかどうかを検証する。

デッドリンクの検出
書き込みのみの変数

- 考えられるリスク
  - 使用されていない可能性がある。
  - 書き込んでいるモジュールは、リンクされているが呼び出されていない可能性がある。
  - 使用されていないソースコードが、プロジェクトに混入している可能性がある。
  - 参照時に変数名を間違った可能性がある。

デッドリンクの検出
書き込みのみの変数

- レビューポイント
  - 「値が更新されているにもかかわらず、値が参照されない変数」をリストアップする。
- レビューの観点
  - 書き込みアクセスのみしか存在しないのは、意図的かどうかを検証する。
  - 参照されていない理由が、明確になっているかどうかを検証する。

タスク間共有変数

- 組み込みソフトウェアでは、グローバル変数の使用が多い。それらの中には、無意識にタスクや割り込みルーチン間で共有してしまっていることがある。
  - タスク間で共有されているグローバル変数を確認する。
- タスク間でグローバル変数が共有されていると勝手に値が更新され、予期していない値を参照する場合がある。
  - 制御するには、排他処理が必須である。

タスク間共有変数
複数タスクから書き込まれる共有変数

- 考えられるリスク
  - タスク間で共有されているグローバル変数の値を読み込むとき、その値は自ら書き込んだ値を期待していても、実は、他のタスクから値が更新される可能性がある。

タスク間共有変数
複数タスクから書き込まれる共有変数

- レビューポイント
  - 「複数のタスク／割り込みルーチンから値を書き込まれる変数」をリストアップする。
- レビューの観点
  - 割り込み許可／禁止やセマフォの取得／開放の排他処理が行われているかどうかを検証する。
  - タスク／割り込みルーチンが並行して実行される場合があるかどうかを検証する。

タスク間共有変数
複数回読み込まれる共有変数

- 考えられるリスク
  - 同一タスク内で、同一のグローバル変数を複数回、読み込むとき、読み込んだ値は、同じ値であることを期待しても、実は、他のタスクから値が更新されている可能性がある。

タスク間共有変数
複数回読み込まれる共有変数

- レビューポイント
  - 「1つのタスク／割り込みルーチン内で2回以上、値を読み込まれる共有変数」をリストアップする。
- レビューの観点
  - 割り込み許可／禁止やセマフォの取得／開放の排他処理が行われているかどうかを検証する。
  - タスク／割り込みルーチンが並行して実行される場合があるかどうかを検証する。
  - 常に最新の値を期待しているのか、変わらない値を期待しているのかを確認する。

データフローのループ

- タスク間でグローバル変数を経由してループが構成されてしまうことがある。
- 例えば、タスクAは、グローバル変数V1を読み込み、グローバル変数V2を書き出す。逆に、タスクBは、V2を読み込み、V1を書き出す。(タスクBの出力がタスクAの入力に、タスクAの出力がタスクBの入力になっている。)
- つまり、V1とV2を介して、2つのタスクがフィードバックループを構成している。
  - タスクAとタスクBは、どちらが先に実行されるのか？
  - タスクAとタスクBの優先順位に違いがあるのか？
  - タスクAとタスクBの実行周期に違いがあるのか？

データフローのループ

- 考えられるリスク
  - 例えば、タスクAの方がタスクBより実行周期が2倍長い場合、タスクBが、タスクAによって、まだ、書き込んでいないV2の値を読み込んだり、タスクAが、タスクBによって、2回書き込んだ後のV1の値を読み込んでしまう可能性がある。

データフローのループ

- レビューポイント
  - 「データフローでループができている箇所」をリストアップする。
- レビューの観点
  - タスクが起動される度に読み込まれるデータの値が、周期遅れだったり、周期が進んでいたり、するのは、期待通りかどうかを検証する。

割り込み禁止／許可の対応

- タスク間で共有される変数の排他処理をするために、割り込みやセマフォがよく使用される。
  - 割り込みの禁止と許可
  - セマフォの取得と開放
- これらの排他処理は、禁止と許可、取得と開放が対になっている必要がある。
  - 排他処理を同一の関数内で行っておらず、関数呼び出しのシーケンスの中で行っている場合は、対になっているかどうか、判定しにくい。

割り込み禁止／許可の対応

- 考えられるリスク
  - 割り込み禁止とすべき区間を割り込み許可になっているために、共有変数が書き換わってしまったり、割り込み許可とすべき区間を割り込み禁止になっているために、共有変数に書き込めなかったり、してしまう。
    - コピー／ペーストのミス
    - #ifdef〜#endifの誤り

割り込み禁止／許可の対応

- レビューポイント
  - 「割り込み禁止／許可、セマフォの取得／開放を使っている箇所」をリストアップする。
- レビューの観点
  - 割り込み禁止／許可、セマフォの取得／開放の対応がとれていない箇所が、期待通りかどうかを検証する。

再帰関数

- プログラムが複雑になると、間接的に関数呼び出しの再帰状態が形成されることがある。
- 例えば、関数Aが関数Bを呼び出し、関数Bがさらに関数Cを呼び出す。その関数Cが、最初の関数Aを呼び出すと、関数A→関数B→関数C→関数Aの再帰状態が形成される。
- 意図的でないときは、注意深くレビューする必要がある。

再帰関数

- 考えられるリスク
  - 無意識に無限ループが構築されており、再帰から抜け出せないために、スタックが枯渇したり、ハングアップしたり、する可能性がある。

再帰関数

- レビューポイント
  - 「再帰関数を使っている箇所」をリストアップする。
- レビューの観点
  - 再帰の終了条件が設定されているか、検証する。
    (無限ループになっていないか)
  - 最大再帰回数が設定されているか、検証する。
  - 設定されている回数、再帰したとき、必要なスタック
    などリソースが確保されているかどうかを検証する。

変数の上書き

- プログラムが複雑になると、一つのグローバル変数にいろいろな関数から書き込みが行われる。すると、書き込んだ値を読み込む前に、再び、書き込んでしまうことがある。
- 例えば、グローバル変数Aに代入を行っている関数Aの後で、同じく変数Aに代入を行っている関数Bが続けて呼び出されると、変数Aに対して、関数Aと関数Bが参照を挟まずに、連続で代入を行うことになる。
  - 関数Aと関数Bの単独のレビューでは検出できないが、関数のコールシーケンスを考慮することで、検出ができるようになる。

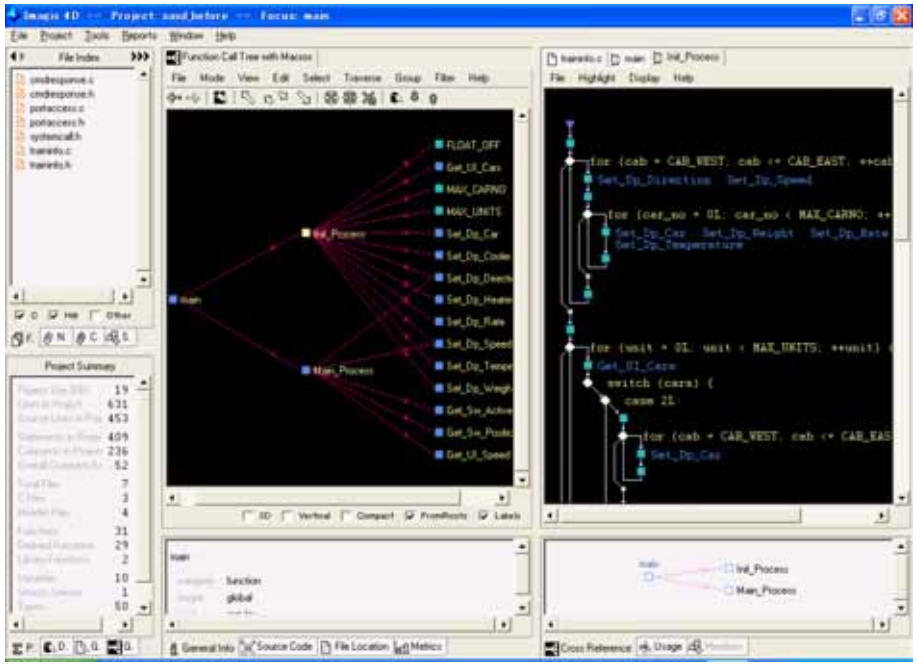
変数の上書き

- 考えられるリスク
  - 最初に行われた代入の結果は利用されない。
  - 2回目の代入の値は、想定しているものと異なる可能性がある。
  - 同じ値を2重に書き込んでいたり、不要な初期化を毎回行っている可能性がある。
  - 実は、代入が連続しているのではなく、参照し忘れている可能性がある。

変数の上書き

- レビューポイント
  - 「同一の変数に対して複数の代入式があり、それらの代入式の間で、一度もその変数（代入式の左辺）を参照していない箇所」をリストアップする。
- レビューの観点
  - 連続した複数回の代入文で値を上書きされたグローバル変数は、最後の代入結果を除いて、値が無視されるが、それが期待通りかどうかを検証する。

ソースコード構造解析ツール
Imagix 4D



- **主な機能**
  - **組み込みC言語用構造レビュー・レポート（チェックリスト）の作成**
  - **コールツリー、フローチャート、クロスリファレンスの表示**
  - **関数と変数の関係を表示**
  - **レビュー用HTMLドキュメントの作成**
  - C/C++/Java**の解析が可能**

ソースコード構造解析ツール
Imagix 4D

- 9種類の構造レビュー・レポートを作成
  - 読み込まれるだけの変数
  - 書き込まれるだけの変数
  - 意味のない代入をしている変数
  - 再帰関数
  - 多重書き込み変数
  - リエントラント関数
  - 割り込みの禁止・許可の対応
  - 複数タスクから共有されている変数
  - 1周期前の値を使用している可能性のある変数(Z変数)

ソースコード構造解析ツール
Imagix 4D

- **お問合せ先:**
  - **株式会社東陽テクニカ**
  - **ソフトウェア・ソリューション**
  - TEL: 03-3245-1248
  - E-mail: ss_sales@toyo.co.jp

参考資料

- 2004年2月13日 東陽テクニカ開催
  「ソースコード・レビューのポイント」
  菊池光彦様 ご講演資料