（資料3-1）

IPA Better Life with IT

組込みシステム品質確保のための
設計・実装ミスの防止セミナー

組込みソフトウェア開発向け コーディング作法ガイド[C言語版]Ver3.0の活用方法（ミニ演習付き）

2019年1月29日

独立行政法人情報処理推進機構（IPA）
社会基盤センター 産業プラットフォーム部 調査役
久野　倫義

 IPA IT Knowledge center

お話すること

1. ESCRとは
2. ESCRの改訂について
   ～セキュアコーディングに向けて～
3. ESCRの活用方法

1. ESCRとは？

ESCRとは

組込みシステムの信頼性向上をミッションとして、2004年に開始されたIPA活動の成果物として作成

目的	：	実用的なコーディング規約の策定・運用の促進
対象言語	：	C言語、C++言語
対象読者	：	規約の作成者、プログラマー、レビューアー

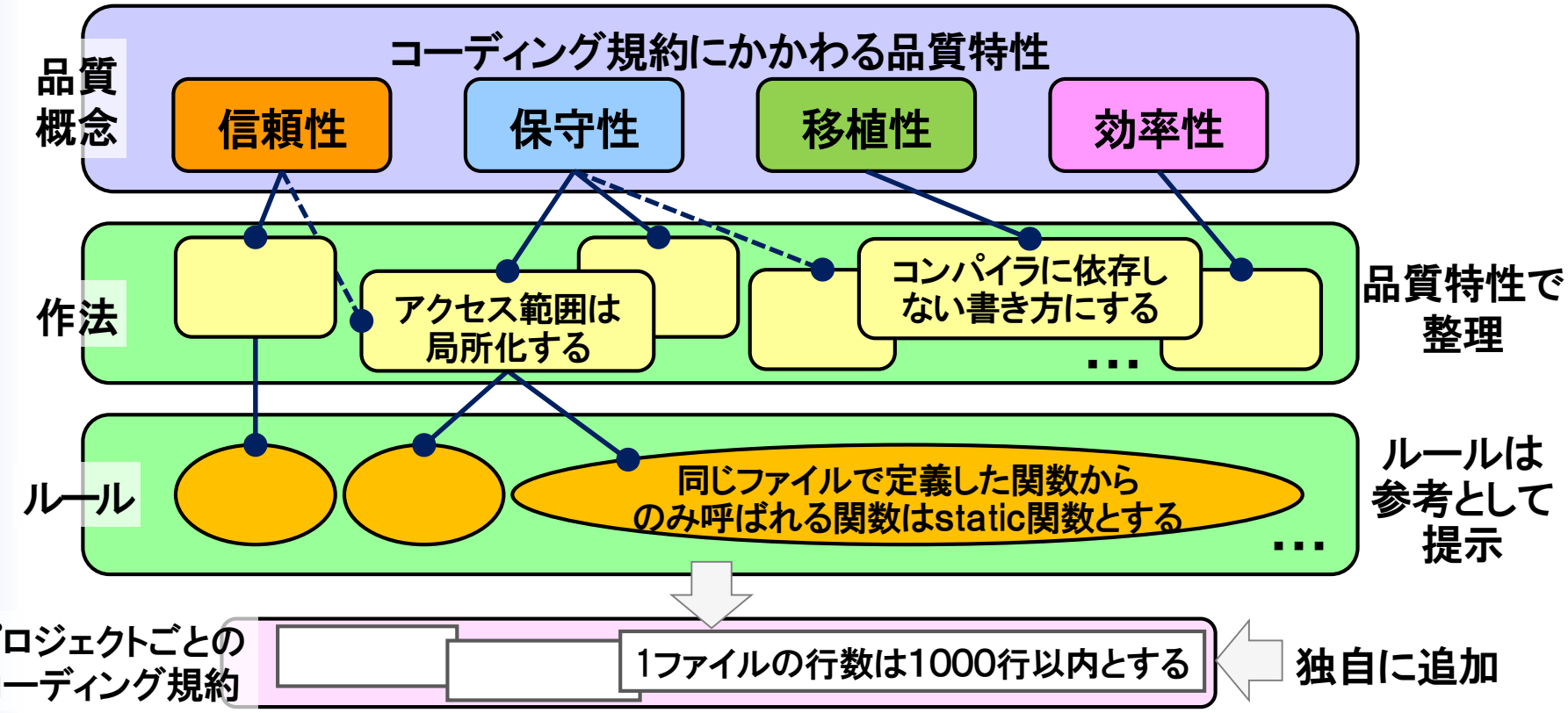
ESCRの特徴

- 品質特性により、ルールをトップダウンに体系化
  - 信頼性、保守性、移植性、効率性 といった品質特性で整理し、理解促進
- すぐに使えるルールのリファレンス
  - 作法に対応し、MISRAや、GNU、参加企業のコーディングルールなどを提示
- ルールの特性を提示
  - 対応する品質特性を保つために重要なルール、など
- 初級者にもわかり易い表現

コーディング規約を分かりやすくするために

- ソフトウェアの品質特性をもとに、作法とルールを体系化する

作　法　：　品質向上のために守るべき具体的な実装の考え方
ルール　：　言語依存性を考慮した具体的なコーディングの決め事



2. ESCRの改訂について
（セキュアコーディングに向けて）

脆弱性に関する状況

米ICS-CERTより

- Emerson社の設備管理システムに不適切な特権管理等の脆弱性
  https://ics-cert.us-cert.gov/advisories/ICSA-18-270-01
  CVSSv3では基本値「10.0」となっています。

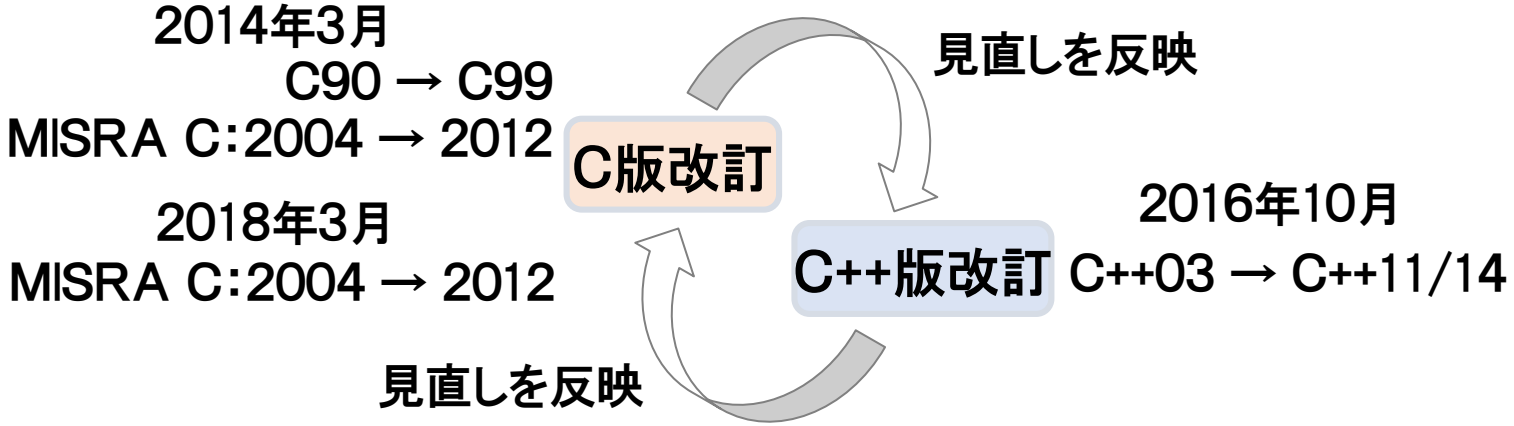
- General Electric社のデジタルメーターにヒープベースのバッファオーバーフローの脆弱性
  https://ics-cert.us-cert.gov/advisories/ICSA-18-275-02
  CVSSv3では基本値「7.6」となっています。

- WECON Technology社のPI Studioにスタックベースのバッファオーバーフロー等の脆弱性
  https://ics-cert.us-cert.gov/advisories/ICSA-18-277-01
  CVSSv3では基本値「9.8」となっています。

※CVSS(Common Vulnerability Scoring System)v3の深刻度
緊急:9.0～10.0、重要:7.0～8.9、警告:4.0～6.9注意:0.1～3.9、なし:0
https://www.ipa.go.jp/security/vuln/CVSSv3.html

ESCRの改訂方針

準拠する言語規格などについての改訂



セキュリティへの対応（品質規格改訂への対応）

- JIS X 0129の後継である 25010 でセキュリティが特性に格上げされた
  → セキュリティは基本的には設計段階の特性と考え、
  セキュリティ特性は採用せず、バッファオーバーフローなど セキュリティに影響するコーディングの説明を追加した。
- セキュリティに配慮したコーディングの重要性の高まり
  → 脆弱性との対応付け、CERT Cコーディングスタンダードとの対応付けを明確化する

ESCRでのセキュリティ対応の強化

- 現状のESCRのルールにはセキュリティの観点から重要なものが含まれる
  - それらとCERT C コーディングスタンダードのルールとの対応付けを行い、**コーディング規約の作成段階からセキュリティを念頭に置くことの重要性を示す**

CERT C ルール		ESCR ルール
EXP34-C	nullポインタを参照しない	R3. 2. 2 ポインタは、ナルポインタでないことを確認してからポインタの指す先を参照する
INT33-C	除算および剰余演算がゼロ除算エラーを引き起こさないことを保証する	R3. 2. 1 除算や剰余算の右辺式は、0 でないことを確認してから演算を行う

✓ 攻撃可能な脆弱性を作り込まない
✓ コードの保守性、移植性の向上

ナルポインタ参照を悪用して任意コードを実行可能。DoS攻撃につながる可能性。

※ CERT C コーディングスタンダード
脆弱性に繋がる恐れのあるコーディング作法や未定義の動作を極力減らすことを目的にまとめられたコーディング規約。C言語を使ってセキュアコーディングを行うためのルールとレコメンデーションを定めたもの　https://www.jpcert.or.jp/sc-rules/

新ルールの例(1)

- 【作法】共有リソースに対して処理を行う際には排他制御を行う。

R3.11.1　並行処理ではvolatileを同期プリミティブとして使用しない。

並行処理や非同期的なシグナル処理では、データの更新結果を他のスレッドに適切に反映する必要がある。 他のスレッドによる更新の不可分性を保証するための目的でvolatileが利用されていることがあるがこれは誤りである。

・・・

なお、同期用プリミティブの取得と解放は、同じモジュール単位内の同一抽象レベルで行うことが望ましい。

例

```
int v = 0;
mtx_t flag; // 排他制御用ミューテックス
 …
mtx_lock(&flag);
v++; // クリティカルセクション内。不可分に処理される
mtx_unlock(&flag);
 …
```

修正の例(1)･･･ 解説の追加

R3.2.2	ポインタは、ナルポインタでないことを確認してからポインタの指すメモリを参照する。

ナルポインタや適正でないメモリを指すポインタを介してメモリにアクセスするとハードウェアトラップやメモリ破壊が発生するため、対策が必要である。

対策の例:

(1)使用済みのポインタにナルポインタを代入する。 ポインタの指す先を参照する前に検査することをルール化することで、メモリの解放後使用や二重解放が回避できる。

(2)近年の組込みOSにはポインタの値の適正を検査するシステムサービスを提供するものがある。 これらのOSを使用する場合は、このシステムサービスを必ず使用する。 ポインタによるメモリアクセスの前にポインタの値が適正であることを確認することで、不当なメモリへのアクセスを回避できる。

コラムによる解説

- ESCRのルールにはしないが重要な項目であるため、「コラム」を設けて解説する
  - 設計に近いと考えられる項目、ライブラリに関連する項目 など
    - 文字列ライブラリの使用、
    - 機密情報の取り扱い、
    - 信頼できない可能性のあるデータの取り扱い
      - 整数値
      - 書式文字列
      - ファイル名やパス名
      - ファイルの特定

    に分けて概説する
  - 対応するCERT Cルールを提示する

3. ESCRの活用方法

3.1 新規コーディング規約の作成／既存コーディング規約の充実

3.2 プログラマの独習、研修の学習教材

3.3 ミニ演習

3.1 新規コーディング規約の作成／既存コーディング規約の充実

1)新規コーディング規約の作成

Step-1 コーディング規約の作成方針を決定
Step-2 決定した作成方針に沿ってルールを選択
Step-3 ルールのプロジェクト依存部分を定義
Step-4 ルール適用除外の手順を決定

2)既存コーディング規約の充実

・抜けモレの防止
Step-1 既存コーディング規約のカテゴリ分け(信頼性、保守性等)
Step-2 既存コーディング規約への観点の追加(信頼性3.1等)

・ルールの必要性の明確化
ESCRの作法とルールの適合例を参照し、規則の必要性を認識

3.2 プログラマの独習、研修の学習教材

1)プログラマの独習

ESCRを読み、

- 信頼性を高くするコーディング方法
- バグを作り込まないようにするコーディング方法
- デバッグ・テストがしやすいようにするコーディング方法
- 他人が見て、見やすいようにコーディング方法とその必要性

などを学習できます。

2)研修の学習教材

- ESCRを使って、テストを作る！
- テストを通じて、コーディング作法を周知徹底しよう！
- セキュリティ教育と同様に、継続的に確認すべき！

さあ、どこに問題があるでしょうか？

不適合例

```
void func1(char *cp) {
  size_t x;
  x = sizeof(cp);



void func2(int arg[MAX],size_t n) {
   size_t argsize;
   argsize = sizeof(arg);
```

2)研修の学習教材～問題(上級者／レビューアー向け)～

問1:さあ、どこに問題があるでしょうか？

問2:該当する品質概念は何ですか？

不適合例

```
void func1(char *cp) {
  size_t x;
  x = sizeof(cp);



void func2(int arg[MAX],size_t n) {
   size_t argsize;
   argsize = sizeof(arg);
```

2)研修の学習教材～問題(トレーナー向け)～

問1:さあ、どこに問題があるでしょうか？
問2:該当する品質概念は何ですか？
問3:本例の品質特性、品質副特性(JIS X 25010)と「コード品質」の関係を説明してください。

不適合例

```
void func1(char *cp) {
  size_t x;
  x = sizeof(cp);



void func2(int arg[MAX],size_t n) {
   size_t argsize;
   argsize = sizeof(arg);
```

3.3 ミニ演習

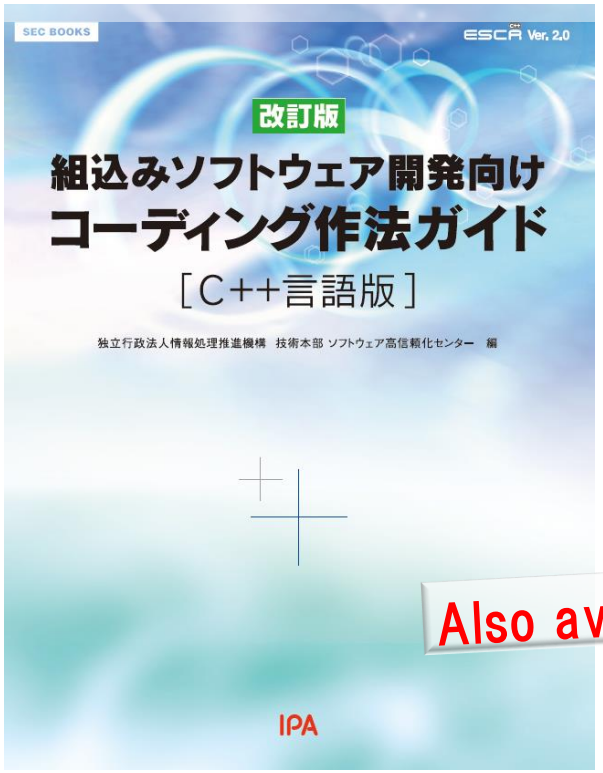
1. グループで意見交換をお願いします（10分）
   ・コーディング規約に関する問題点は、何ですか？
   ・問題に対する解決策は、何ですか？

2. 個人毎にミニ演習（別紙）を実施ください。（10分）
   可能な範囲で実施してください。
   回答は、「ESCR」にありますので、参照頂いて結構です。

3. コーディング規約の定着について、意見交換してください。（10分）

ご清聴ありがとうございました

今後とも ESCR C/C++ のご活用をよろしくお願いします



Also available in English

トレーナーズ向け教材もあります。
https://sec.ipa.go.jp/reports/20120910.html