組込みソフトウェア開発向け「開発技術リファレンス」および「コーディング作法ガイド[C言語版]Ver3.0」の紹介

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

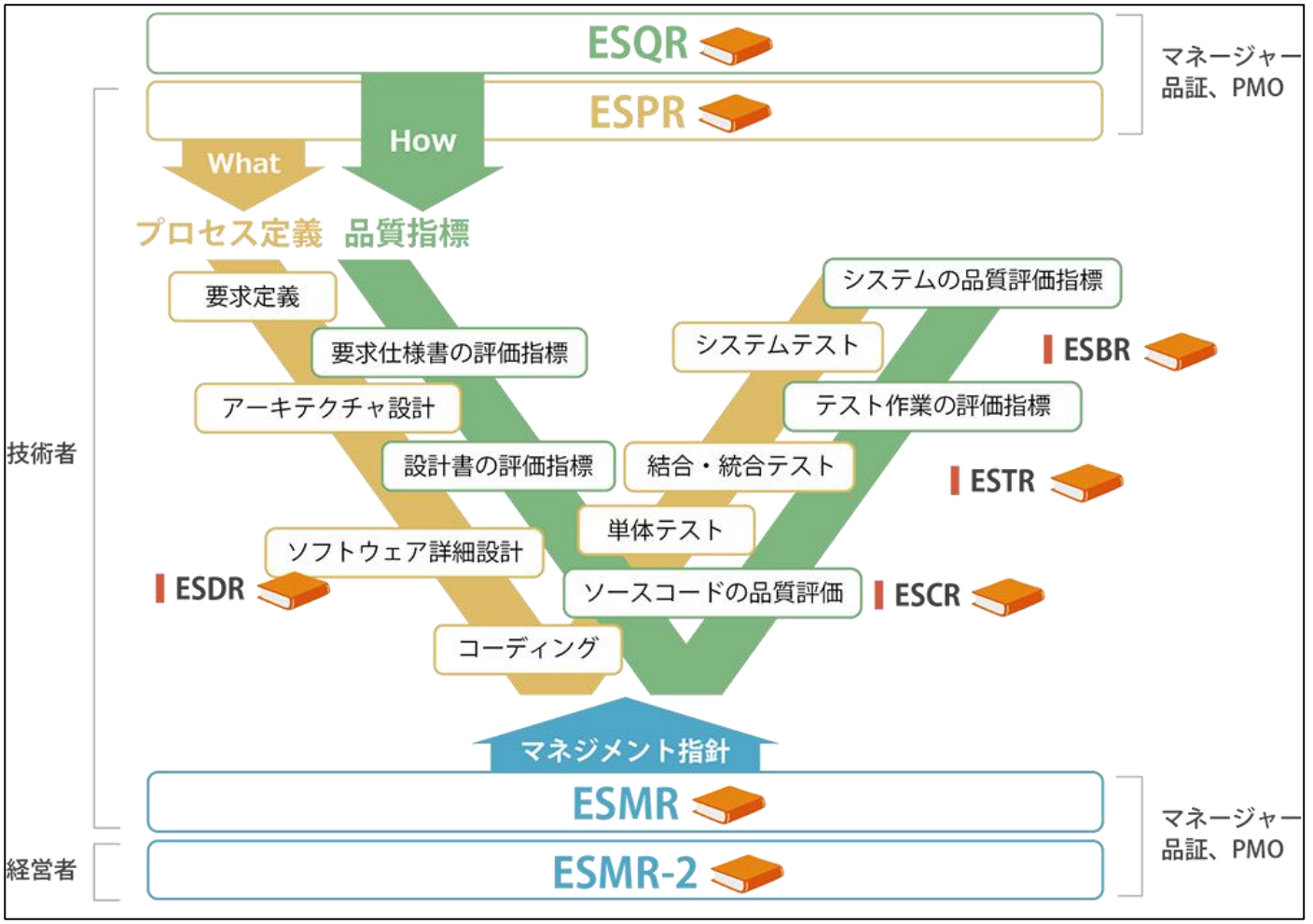
お話すること

1. 開発技術リファレンスとは
2. ESCRとは
3. ESCRの概要
   ～セキュアコーディング対応～
4. ESCRの活用方法
5. 活動状況と今後について

1. 開発技術リファレンスとは

1.開発技術リファレンス【ESxRシリーズ】とは

組込みソフトウェア開発の作業や品質を見える化するために、IPAではソフトウェア開発に関する標準的プロセス、考え方を【ESxRシリーズ】として提供しています。



https://www.ipa.go.jp/sec/softwareengineering/std/emb.html

ESxRシリーズの概要

- ESCR：ソフトウェア実装品質の向上／品質特性を考慮したコーディング作法とルール
- ESDR：ソフトウェア設計品質の向上／ソフトウェア開発現場の設計ノウハウ事例集
- ESTR：組込み製品の品質向上／先進企業で有効性が実証されているテスト事例集
- ESBR：組込み製品の品質確保／日本語として初の体系的バグ管理手法集
- ESPR：開発プロセスの最適化／ソフトウェア開発の標準的プロセスを整理。
  [トレーナー向け(TR)教材あり]
- ESQR：指標を用いた品質コントロールの実現／品質を可視化する手順と指標を整理
- ESMR：効果的なプロジェクトマネジメントを実現／標準的な開発計画書の雛形を整備。
  [TR教材あり]
- ESMG：プロジェクト計画立案手順の詳細を事例とともに説明。[TR教材あり]
- ESMR-2：定量データを活用したマネジメント
- 組込みソフトウェア開発データ白書：製品の特性で変わる組込みソフトウェア開発の生産性・信頼性指標

対象者と効果

- 対象者
  - 経営者/マネージャー/技術者/品質保証/ PMOの方

- 主な効果
  - [経営者] データに基づくプロジェクト管理の徹底 (ESMR-2、データ白書)
  - [マネージャー] 精度の高いプロジェクト計画の作成 (ESMR)
  - [技術者]設計上の工夫点/注意点をノウハウ化し設計品質を向上 (ESDR)
  - [品質保証/PMO] 効果的・効率的な開発プロセスの構築 (ESPR)等

今後もブラッシュアップ予定です。
ご活用ください。

2. 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

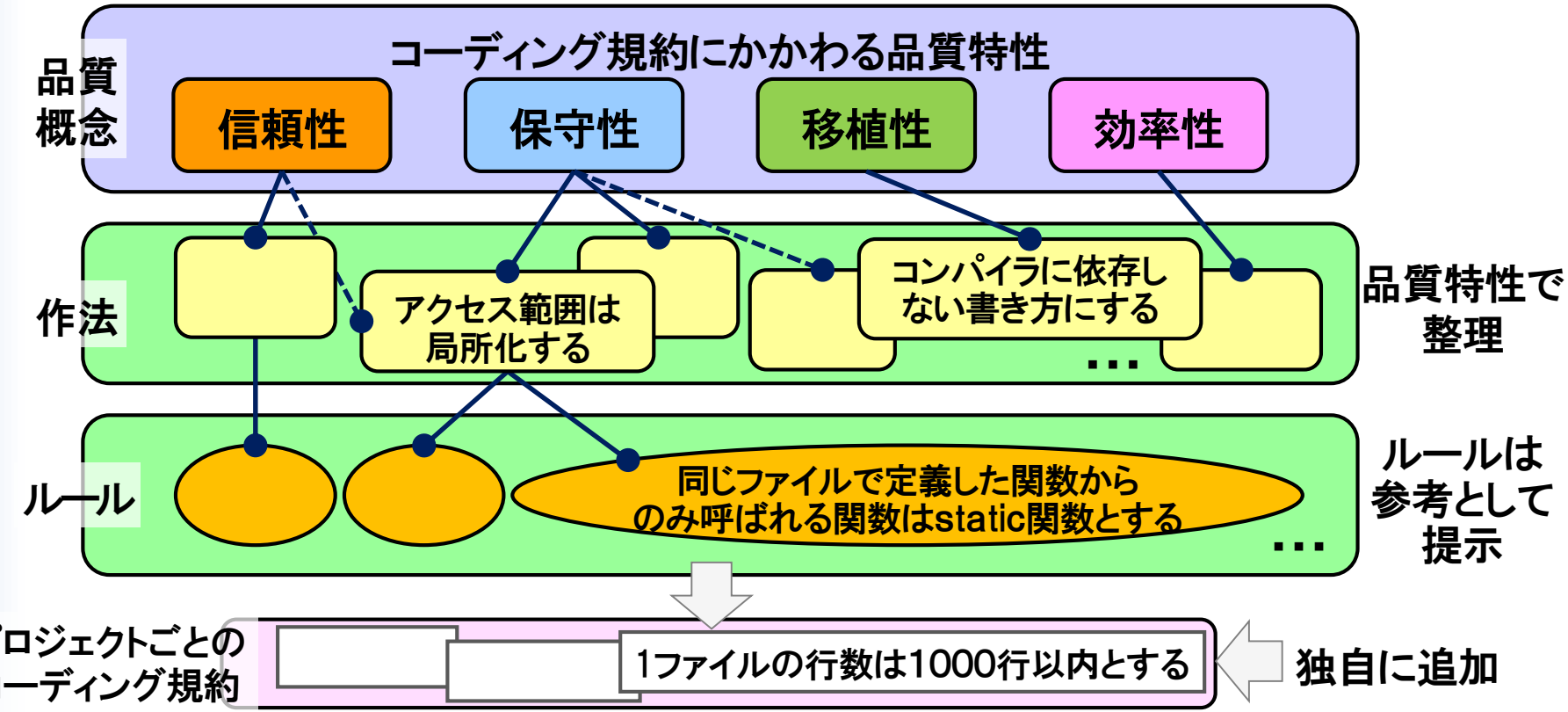
ESCRの特徴

- 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



3. ESCRの概要

～セキュアコーディング対応～

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

※ 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/

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。

4. ESCRの活用方法

4.1　プログラマの独習として利活用

4.2　研修の学習教材として利活用

4.1 プログラマの独習として利活用

プログラマが、

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。

4.2 研修の学習教材として利活用

研修の学習教材として、

- 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);
```

「組込みシステム品質確保のための設計・実装ミスの防止（ミニ演習付き）」(10/21)で、21問を実施。
今後も、同様なセミナーを予定。

5. 活動状況と今後について

ESCR策定の沿革と今後

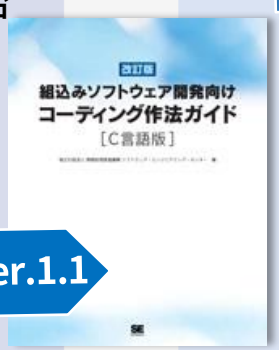
’04 ’05 ’06 ’07 ’08 ’09 ’10 ’11 ’12 ’13 ’14 ’15 ’16 ’17 ’18

コーディング作法ガイド
WG活動開始

ESCR C 作成開始

ESCR C Ver.1.0

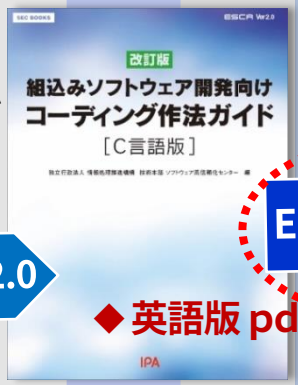
ESCR C Ver.1.1



ESCRのJIS化
(JIS X 0180)

ESCR C 改訂開始

ESCR C Ver.2.0



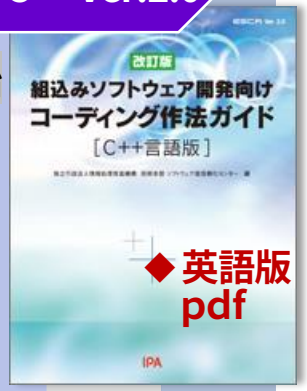
ESCR C Ver.3.0

ESCR C++ 対応開始

ESCR C++ Ver.1.0

ESCR C++ 改訂開始

ESCR C++ Ver.2.0

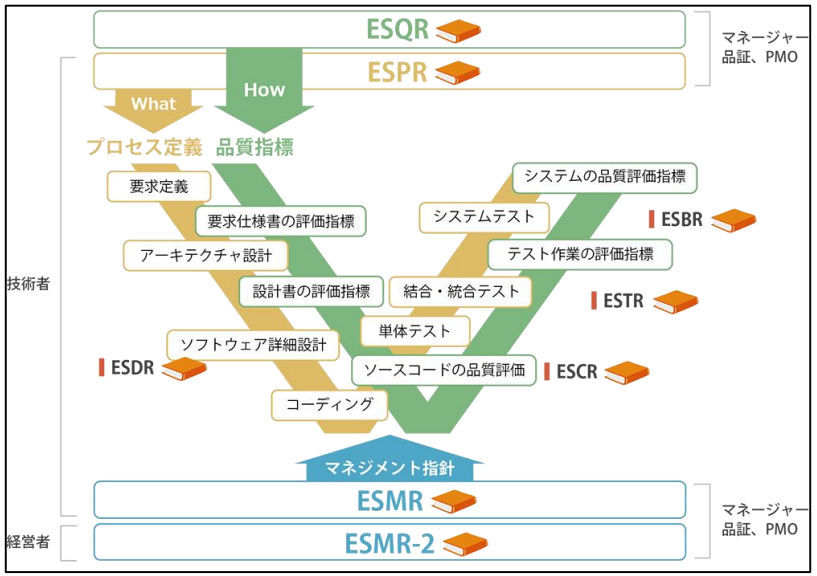


MISRA C++ 改版への対応（今後）

- 品質と生産性の向上に有用との評価
- 書籍および pdf のダウンロード 3万部以上

ご清聴ありがとうございました

今後ともESxR、ESCR C/C++のご活用をよろしくお願いします



https://www.ipa.go.jp/sec/softwareengineering/std/emb.html

https://www.ipa.go.jp/sec/publish/tn16-007.html

https://www.ipa.go.jp/sec/publish/tn18-004.html