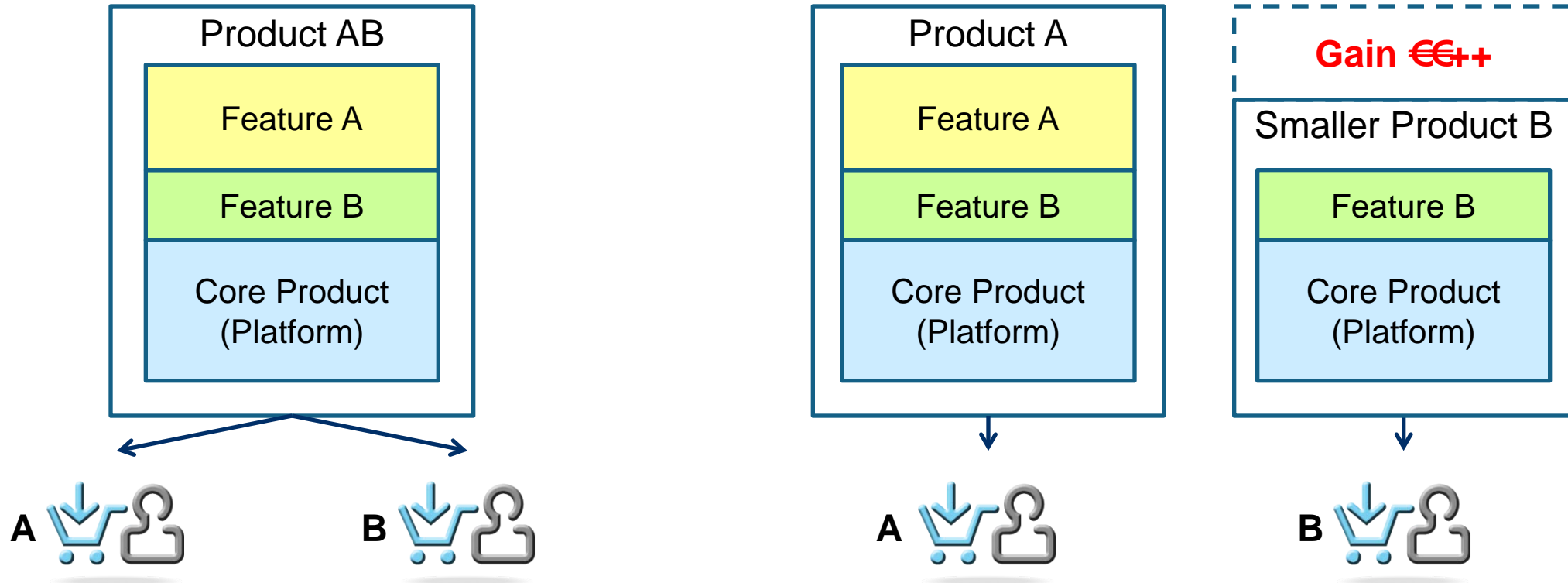


Managing Product Configuration Complexity in CC Evaluations

Dr. Karsten Klohs

/ 14th ICCC, Orlando, September 2013 /

Motivation: Tailoring Products for Customers



→ Fixed Size ROM Mask

→ Multiple Masks not Economic

→ => Single Evaluation

→ Multiple Product Variants on Flash

→ Fitting Size == Cheaper Products!

→ => Multiple Evaluations?! €€-

Problem of Configuration Complexity: Example German eHC

→ Security Service Options (SFR Packages):

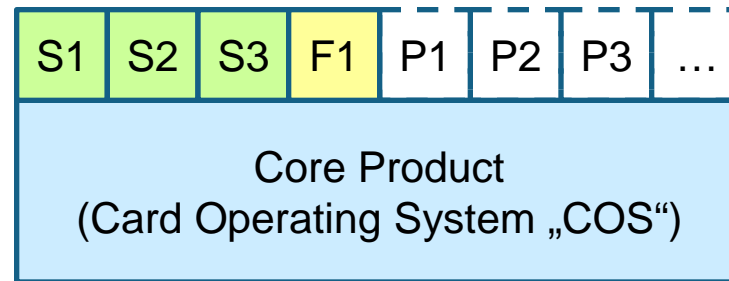
- S1: Multi-Channel Support (needed for HPC, SMC only)
- S2: Contactless Interface (PACE)
- S3: Crypto-Box (Transcription etc...)

→ Functional Options

- F1: USB Interface Support

→ ... and Proprietary Extensions?!

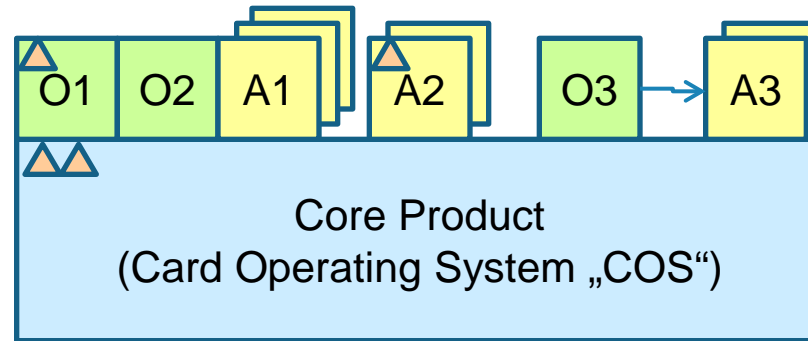
- P1, P2, P3,...: Data Objects, SCP02, Voltage Class C, ...



Standard Options: $2 * 2 * 2 * 2 = 16$
Manageable Evaluations?

With Extensions: $16 * 2 * 2 * 2 \dots \geq 128$
⇒ Not Manageable
⇒ Killed by Combinatorial Complexity

Additional Aspects of Product Variance



→ „Options“ are the Simplest Case Only

→ Mutually Exclusive „Alternatives“, e.g.:

- Standard Conformant vs. Customer Specific Implementation
- Performance Optimised vs. Memory Optimised Variant
- Optimised Special Case vs. General Implementations

→ Variant Dependencies

- Option O3 depends on an Instance of Alternative A3 ...




→ Build Parameters: ▲

- Buffer Sizes, Error Code Mappings, ... => cannot be modelled by variants



Managing Configuration Complexity

→ Positive Factors

■ General



-  □ Customer's select only a limited Subset from the possible Variance
-  □ Software Engineering Principles cover Software Variance Completely
-  □ All Variants are Available for Evaluation

■ CC-Related



-  □ Variance often out of the SFR-enforcing core of the product
-  □ ALC assessment in the CC approach allows for verifying process security

→ Challenges

■ Market

-  □ Customer Selection Difficult to Predict
-  □ Tight time-to-market Requirements

■ CC-Related (*will be detailed...*)

-  □ „Variance in AVA”
-  □ Solving General Variance Issues in all Assurance classes

→ Suggested Steps towards a Solution

1. Improve the Integration of Software Variance into an Evaluation
2. Cover „Selected Variants“ in a Single Evaluation (fall-back solution, examples available)
3. Use in a Lightweight „Prepared Maintenance/Re-Evaluation“
Objective: 4 weeks max
4. **Certify the Platform implicitly including all of its Variants**

ASE – Security Target

→ Formalise the Treatment of Variance in the Security Target / PP

- Modular PPs (French Scheme), SFR-Packages (EHC-PP)
- Approach may also support evaluations against several PPs



→ Identification Scheme for Product Variants to replace Identification Enumeration



→ Objective: One Security Target for the Platform not for Each Product Variant

ADV – Development Evidence

→ ADV_FSP: Functional Specifications



→ ADV_ARC: Security Architecture



- Shall describe the „Soundness of the Security Architecture under Product Variance“

→ ADV_TDS: TOE Design



- Describe the security implementation in a modular way (cf. ICC3 2012):
 - Side-Channel Resistance (on assets in input, output or system state)
 - Enforcement of Result and System State Correctness

→ ADV_IMP



ATE – Tests / Developer Perspective

→ ATE_FUN: Functional Testing

- Platform Approval vs. Variant Approval
- Issue: Configuration Complexity (next slide)



→ ATE_COV: Coverage

- Issue: SFR Coverage under Variance



→ ATE_DPT: Depth of Testing

- Component Testing can be Corner-Stone of Variant Testing



→ ATE_IND: Independent Testing

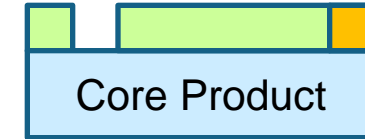
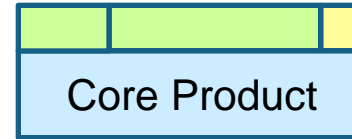
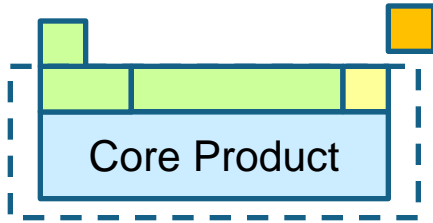
- Should be used for adding Testing Requirements from Variance Assessment in AVA



→ To Be Considered also for Penetration Testing and Side-Channel Analysis!



ATE / AVA - Test Sample Selection Strategies



→ „All Features In“ (1 Sample)

- Works nicely for Options
- Does not Cover Alternatives
- „All Features In“ Product may not fit into Target



→ „Each Alternative at Least Once“ ($|MaxAlt| + X$ Samples)

- Covers Alternatives
- Fits into Target
- May not address (all) Security Relevant Configurations



→ „Each Alternative at Least Once + Security Coverage“ ($|MaxAlt| + X + Y$ Samples)

- Reduce Configuration Coverage on SFR Non-Interfering
- Option for the Evaluator to Add Configurations based on AVA Assessment



AVA – Augmented Vulnerability Analysis

- **Assess the Impact of Variance on System Security based on ADV**
- **Cross-Check the (Functional) Sample/Variance Testing Strategy**
- **Add Security Configurations by ATE_IND (if needed)**

- **Objective: Gain Sufficient Assurance that remaining Configurations will have Similar Security Behaviour**



→ **Key Questions:**

- Acceptance Testing of New Configurations Required?
- Quicker and more Cost-Effective than Re-Evaluations?

ALC – Augmented Life Cycle Assessment

→ ALC_CMx: Configuration Management

- Definition and Management of Variants Should be Integral Part of the Configuration Management



→ ALC_TAT: Tools and Techniques

- Automated / Tool Supported Build of Variants



→ *Objective: Enforce that Variants are Always Constructed Correctly*

→ ALC_LCD: Life-Cycle Definition

- Shall consider the Variant Generation and Approval Process



→ ALC_DEL: Delivery

→ *Objective: Ensure that the Product Variants are Always Shipped Correctly*

Summary

→ Situation

- Flash Memory and Open Platforms enable Customer Tailored Products
- Decreasing Time-to-Market Requirements
- Increasing Number of Customer Configurations to Handle



→ Objective:

- Replace Product Evaluations by a Platform or Product Family Evaluation



→ Solution:

- Evaluate Prepared Product Variance of the Platform
- Evaluate Product Variant Creation and Approval Procedures
- Run a fast “Product Variant Acceptance” Evaluation or Grant Product Family Certificates

