Bull CP8

New Methodologies in Smart Card Security Design

Y.GRESSUS

Methodology and Secure ASIC development manager, Bull CP8

Summary

- Trends
- Opportunities
- New methodologies
  - □ Concurrent Secure development
  - □ Top down methodology
  - □ Concurrent Secure development
  - □ Design for reuse
  - □ Intellectual Property IP development
  - □ IP integration process
  - □ Formal methods
- Conclusion

Trends

- □ **Productivity**
  - **Decrease the development time frame. (“time to market”)**
  - **Rapid prototyping for customer demonstration**
  - **Rapid adaptation to customer needs**
- □ **Flexibility and adaptability**
  - **Generic platform development**
  - **Multi application support (applications and OS independence)**
  - **Hardware and software independence**
- □ **Openness and Security**
  - **open and secure**
  - **Multiple partners for R&D tasks sharing and/or business model**
  - **Without compromising the security requirements**

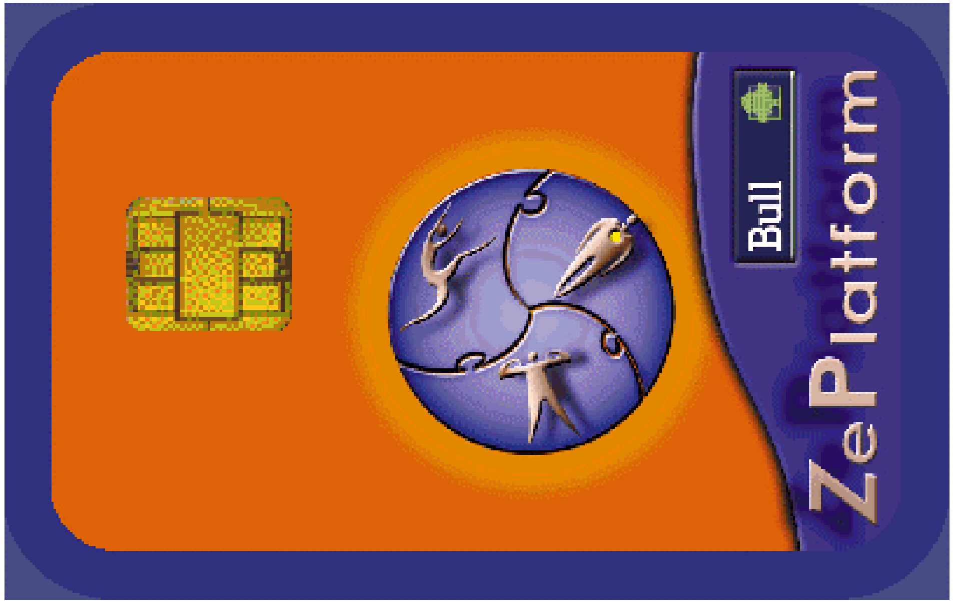
Opportunities

- □ **Design to cost**
  - ● **More integrated function for the same price**
- □ **More power and capacity**
  - ● **1Mips up to 25-30Mips on less than 25 mm2 design**
  - ● **Increase memory size**
- □ **More integration**
  - ● **Take into account the technology evolution**
  - ● **Hardware IP integration (7816,USB, RF…)**
- □ **New HW and SW architecture**
  - ● **16/32bit CISC/RISC processor**
  - ● **Security model and Multi-layers architecture**
  - ● **IP based development for both HW and SW**

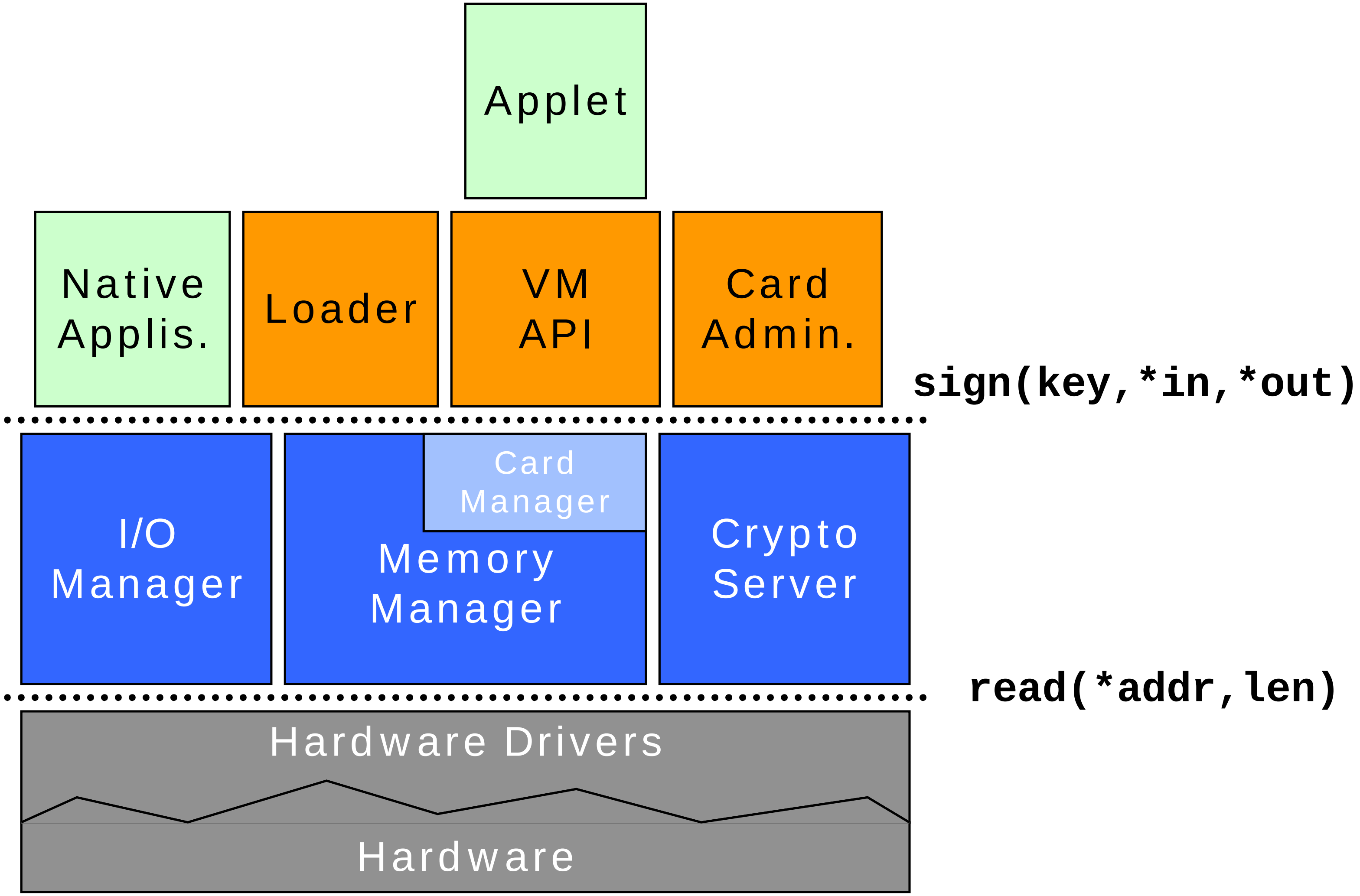
An example:

ZePlatform from Bull/CP8 :

The 32 bit platform



ZePlatform : software concept and modularity



How to develop a modular and secure platform ?

How to reach the goals of productivity, flexibility, openness and security ?

- Apply a new methodology based on:
  - □ Concurrent secure development
  - □ Security and CC assurances
  - □ Intellectual Property (IP), design and evaluation capabilities <u>reuse</u>
  - □ IP protection and test
  - □ Integration process mastery
  - □ Multi developers and /or multi site management

Concurrent Secure development- Objectives

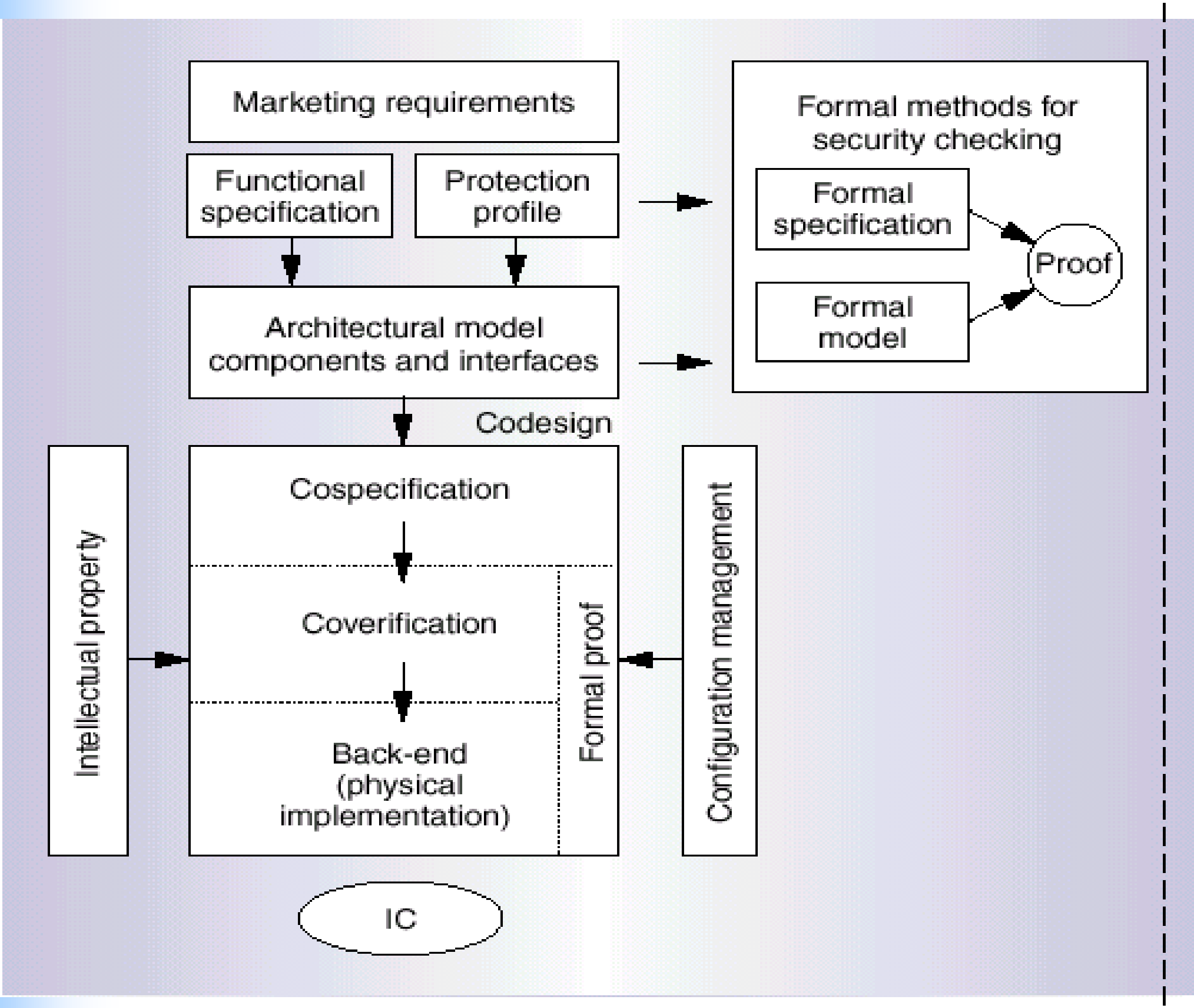
- □ **Decrease the global development time frame by concurrent development tasks**
- □ **Integrate and assure the security requirements and deliverables for evaluation all over the development cycle (Assurance Class)**
- □ **Firewalling between the layers/modules in development to allow multi sites and developers teams**
- □ **Tests development at each stage of the process**
- □ **Control throughout development cycle by using appropriate tools**

Concurrent Secure development - Needs

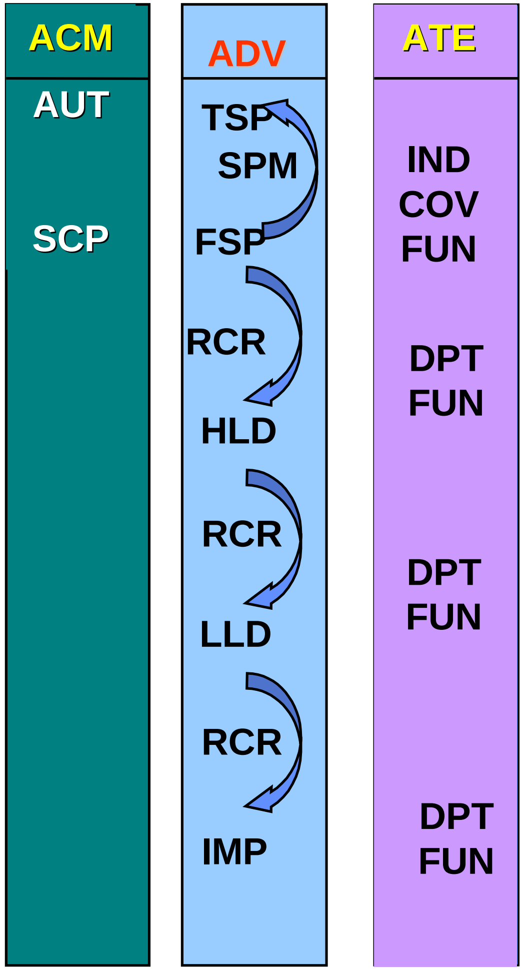
- Structural and modular architecture:
  - Clear interface communication definition
  - IP building process
  - Designed for re-use
- tools for tasks and configuration management.
- Development Life cycle support
- Means and tools to ensure the IP development , test and integration
- Organizational measures to support the development process .

A top-down methodology

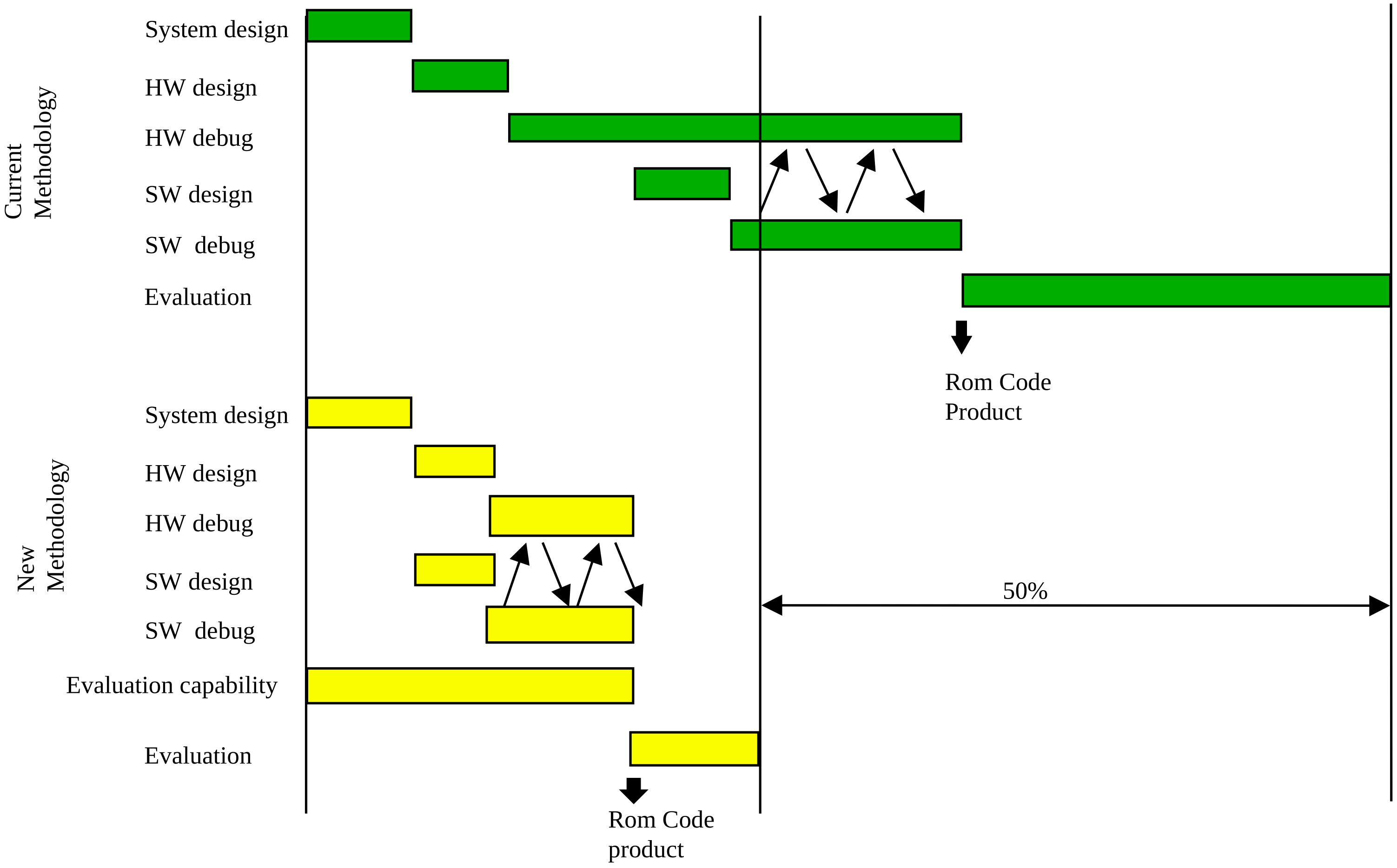
- From market requirements to the IC
- To guarantee an IC and its SW right the first time
- Which takes into account the CC requirements at the first stage of the design
- According to the security level objective
- Based on means and tools to support:
  - Multiple languages (HDL,C/C++,Java,formal..)
  - Multiple IP representations levels including functional and security features
  - Formal methods
  - Life cycle and project management
  - Standard development cycle
  - Multiple development sites



CC Assurance Class correspondence



Development process: Reduce Time to market



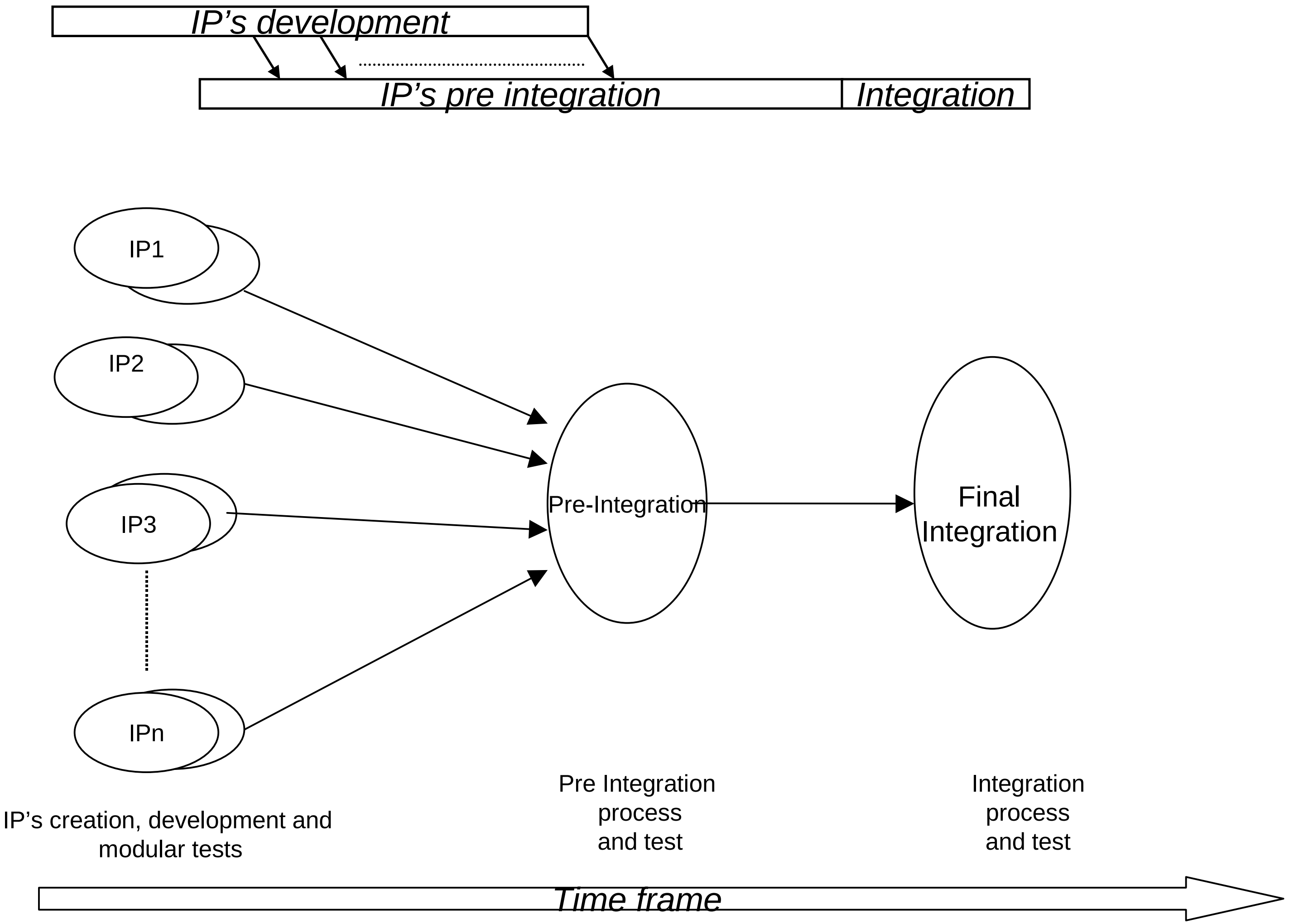
Design for Re-use

- New challenge for a design team
- Implies a robust and correct design based on
  - Structural approach architecture
  - Good specification (complete and stable)
  - Clear interfaces definition for re usability
- Designed for portability
  - Use of standard languages (HDL, ANSI-C….)
- Designed to be verified into a variety of verification tools. HW platform independent.
- Well verified before integration by using appropriate test benches
- Protected for exchange and integration

IP development

- Re use capabilities concepts to be integrated at the design partitioning stage. Associated to the modularity
- Support abstractions levels (e.g.CC: ADV class FSP, HLD,LLD,IMP representations)
- Unique description for functional and security features for model levels representation
- Modular test strategy according to the needs of security test (e.g. CC: ATE class) depending on the targeted EAL level
- IP integration process
- Split in two phases:
  - Pre integration methods and Final integration

From the IP's design to the final integration for product


IP's development
IP's pre integration
Integration
IP1
IP2
IP3
IPn
Pre-Integration
Final Integration
Pre Integration process and test
Integration process and test
IP's creation, development and modular tests
Time frame

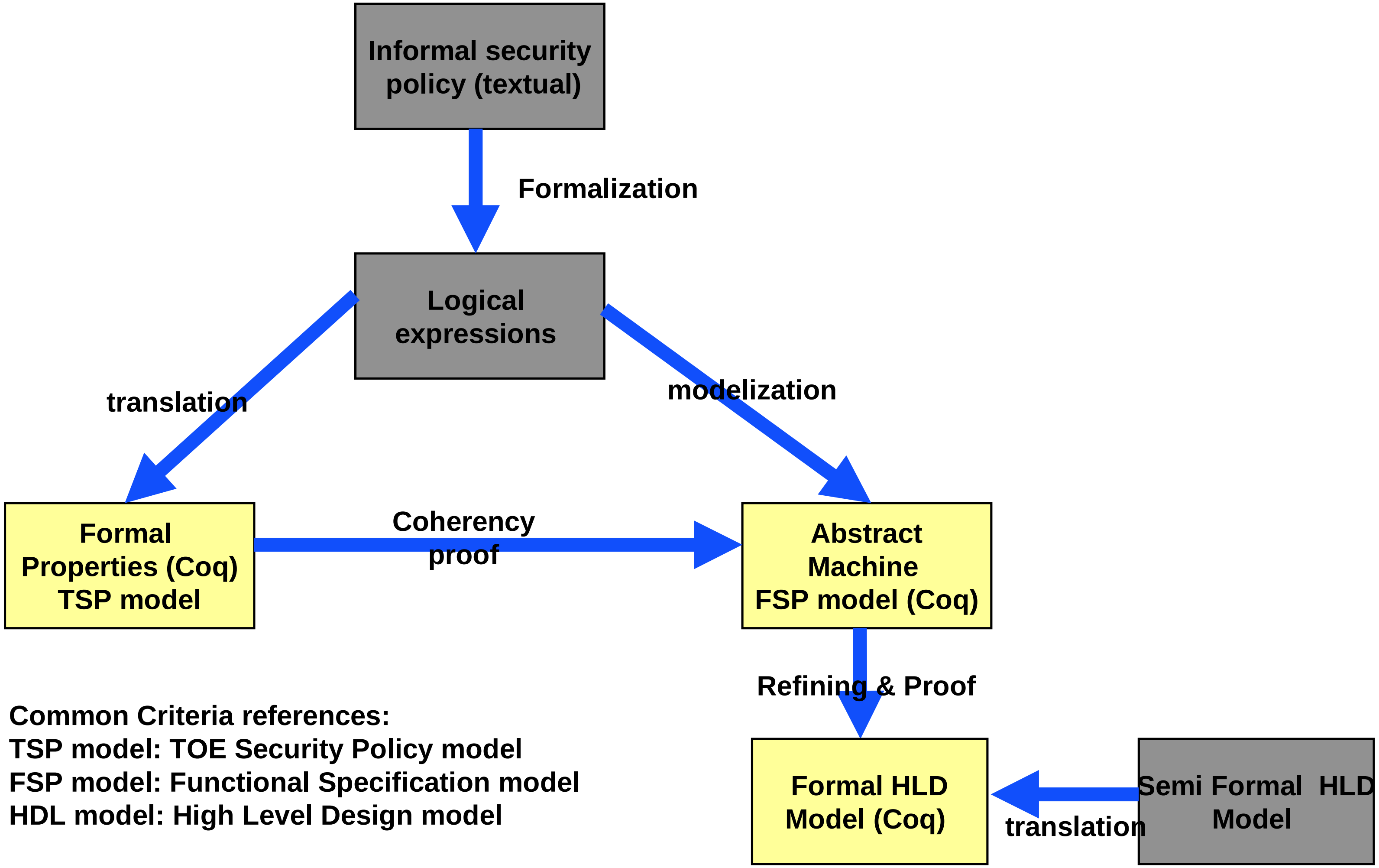
□ IP Pre integration

- Eliminate over the development phase the miss interpretation of the specification and interfaces definition
- Assure an independent development based on the interfaces definition and tests exchanges
- Assure a progressive and bottom up integration of the IP
- Minimize the debug activity by reducing the size of the code to be verified
- Reduce the final integration phase by Putting in place a strategy which allow :
  - ○ To test the IP independently from each others
  - ○ zero knowledge
  - ○ Based on the communication interface specification
- Guarantee and protect the IP integrity

Formal methods

- □ To be compliant with the Assurance Level 5 requirements and more.
- □ Use of the most advanced techniques of semi formal and formal methods:
  - ● Semi-formal/formal methods choice (UML, B ….)
  - ● Define and use a dedicated language (Java, Coq…)
  - ● Define a translation methods from the Functional Security requirements (informal information's) to the TSP model (formal)
  - ● Coherency proof
- □ Started earlier at the development phase

Security Policy formalization - Methodology overview



Common Criteria references:
TSP model: TOE Security Policy model
FSP model: Functional Specification model
HDL model: High Level Design model

Conclusion

- New methods become a reality to
  - Reduce the development time
  - Assure the mandatory correspondence between the standard development flow and the CC requirements for evaluation without extra effort
  - Guarantee a right design the first time
  - Allow the co-design activity
  - Security level guarantee by formal methods.
- These methods have been specified and set up in the European collaborative Project: MASSC (A MEDEA initiative) and now applied for the ZePlatform dev.