



innovations  
for high  
performance  
microelectronics

---

# Tool-supported Composition of Software Modules for Safe and Secure Wireless Sensor Networks

**Steffen Peter**

**IHP  
Im Technologiepark 25  
15236 Frankfurt (Oder)  
Germany**

- **Introduction IHP & Related Projects**
- **Motivation**
- **Configuration Tool Approach**
- **Module Selection**
- **Security Assessment**
- **Conclusion**

## New Institute & Cleanroom

---



## **The Institute**

- Founded in 1991; successor institution to the former institute of the East German Academy with extensive experience in silicon microelectronics
- 200 employees from 16 countries
- Member of the Gottfried Wilhelm Leibniz Society (WGL)

## **Mission**

- Strengthen the competitive position of the German and European microelectronic and communication research
- Act as an innovation center, leading research results towards prototypes
- Enhance the attractiveness of the region as location for high technology

## **Facilities**

- Complete innovation chain from materials to systems, including class-1 cleanroom, 0.13  $\mu\text{m}$  capable pilotline

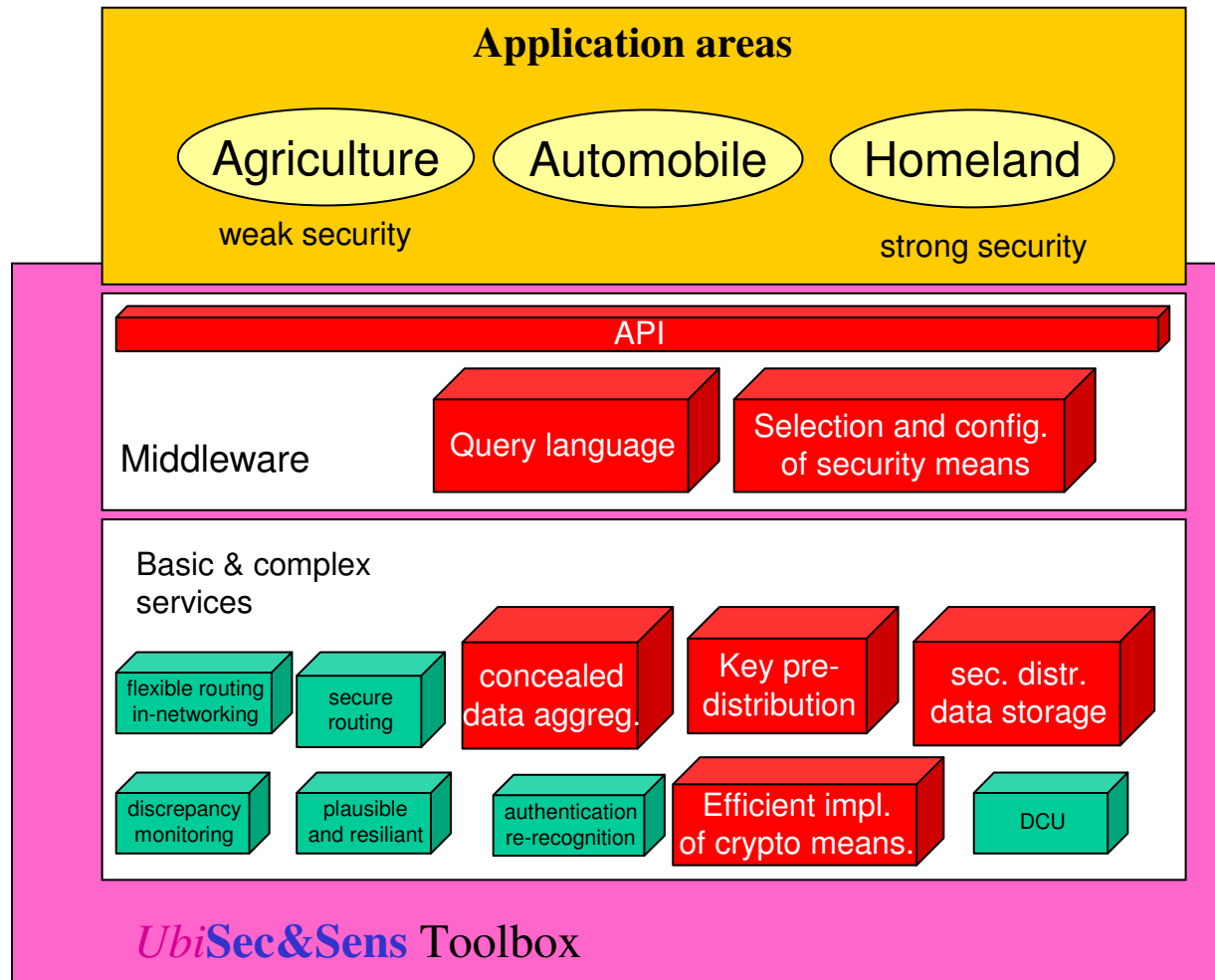
## **Competencies**

- Systems for wireless communication
- RF circuit design
- Extension of silicon CMOS technologies
- Materials for microelectronic technology

## **Strategy**

- Create value through innovation
- Focus on solutions for wireless & broadband communications
- Development of forward-looking technologies and system-level prototypes
- Strategic partnerships

# UbiSec&Sens Project (2006-2008) Overview





# UbiSec&Sens Vineyard Scenario (2008)



## The setting

- Commercially run vineyard “Weingut Georg Naegele” in Neustadt, Germany
- Deployment in operational part of the vineyard, no special arrangements
- Requirements collected together with the proprietors

## Key requirements

- Diverse sensing capabilities together with geographic coverage
- Resilience to faulty data and component failures
- Storage of data to the network as well as remote access either synchronously or asynchronously
- Long deployment times; self-organization

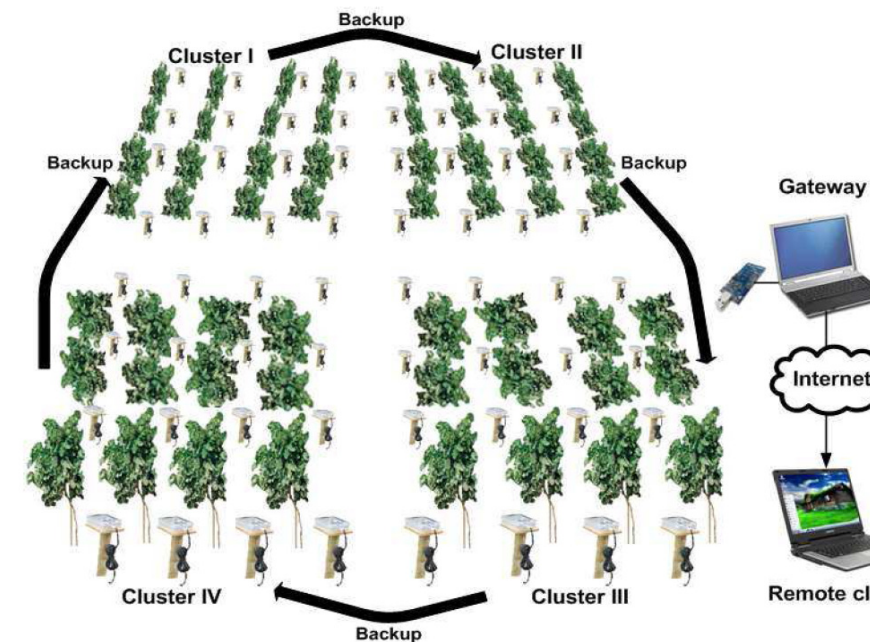


## Encountered problems: (like in LOFAR-Argo Project)

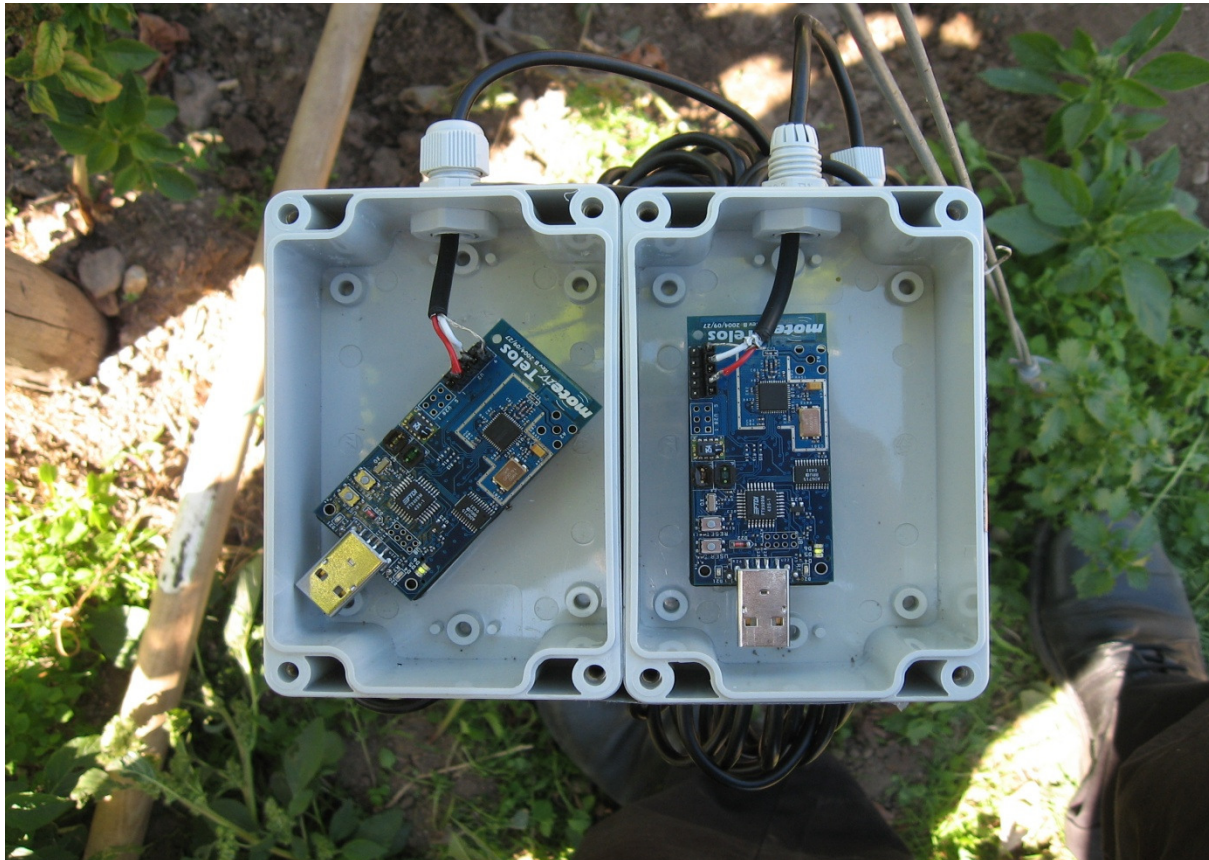
- Insufficient software engineering
- Incompatible modules
- Unforeseen side effects between modules

## Additional problems:

- Harvester does not like sensor nodes
- Metal container influence wireless communication



## Sensor nodes



IHP Im Technologiepark 25 15236 Frankfurt (Oder) Germany

[www.ihp-microelectronics.com](http://www.ihp-microelectronics.com)



## Realflex project (2008-2010)



### Water works



Standards,  
existing architektur

### Biogas facility



large distance,  
local intelligence

### Roboter cell

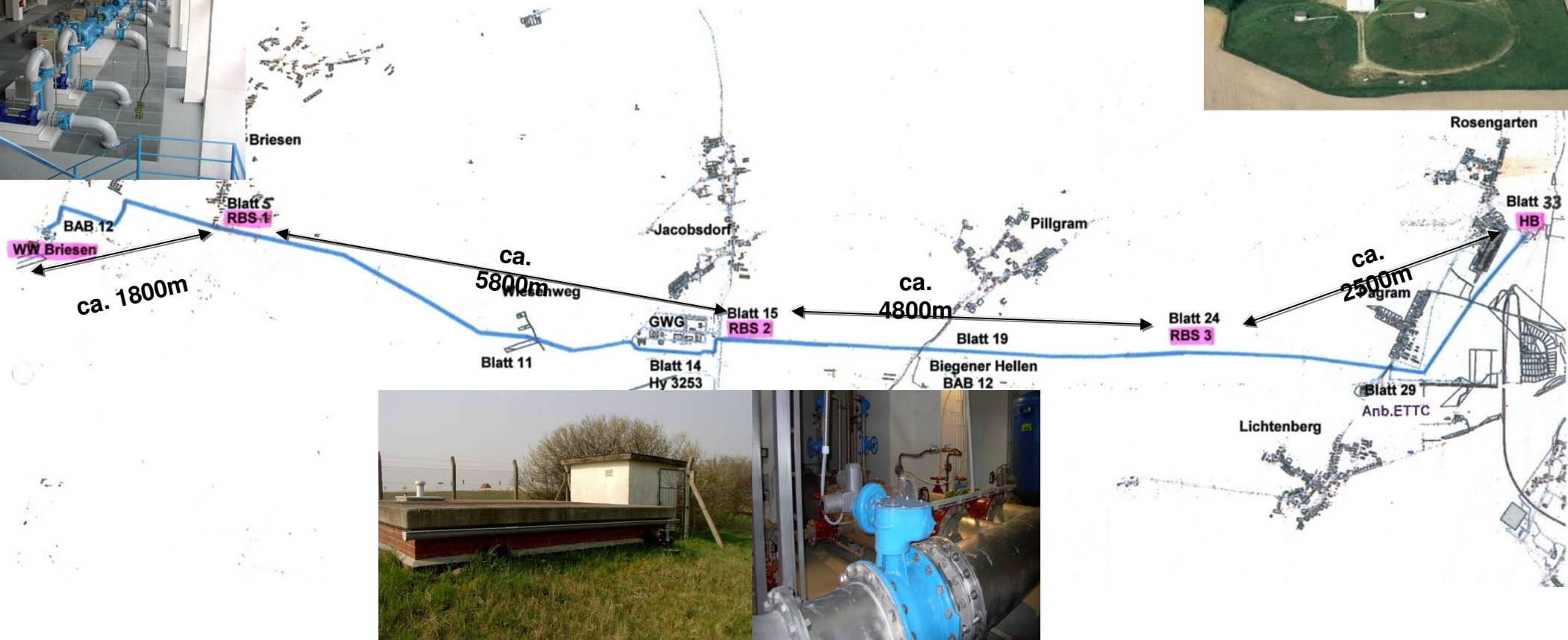


Small latency,  
dependability

wireless architecture for industrial automation



## Scheme of water mains

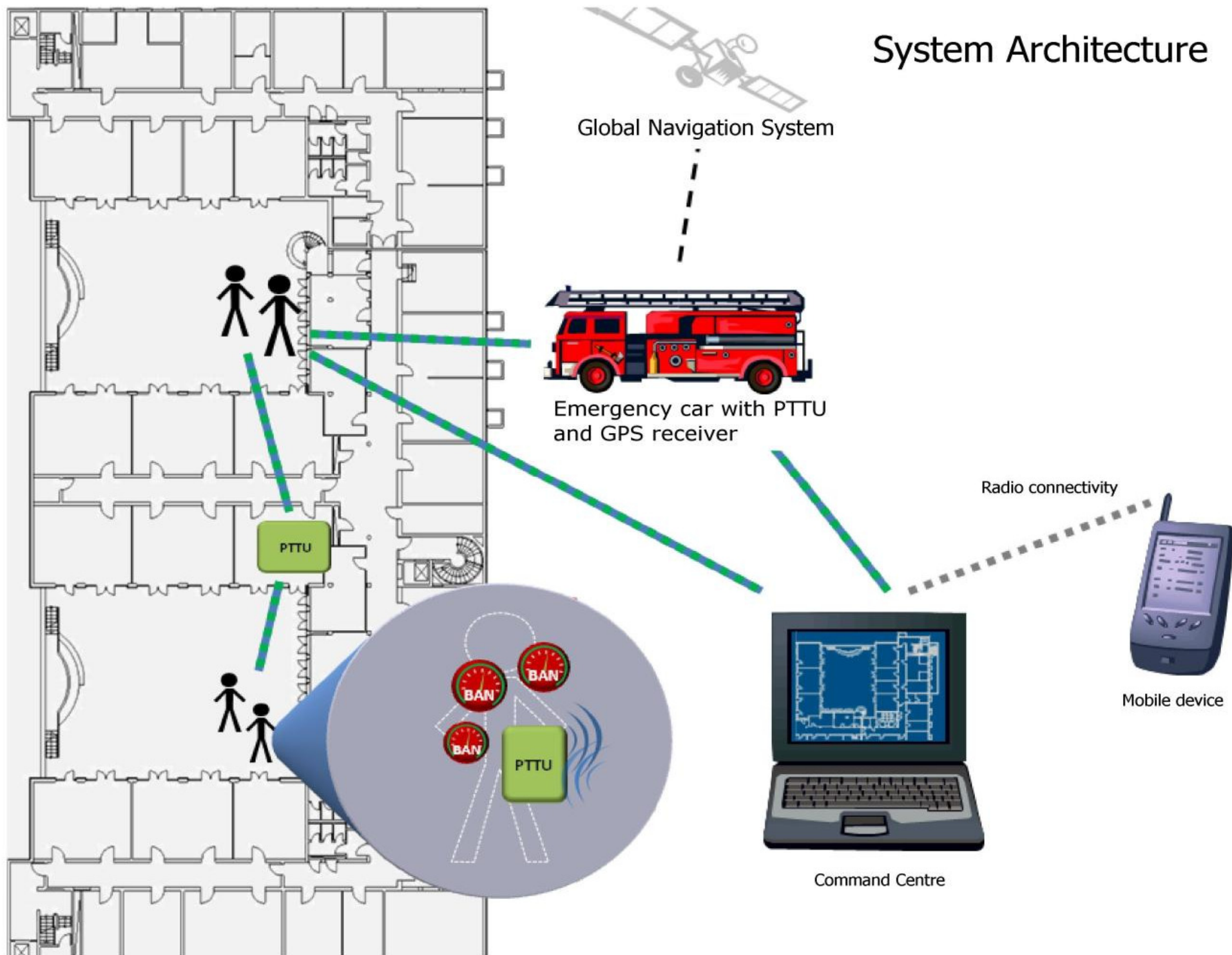


## Projekt: FeuerWhere





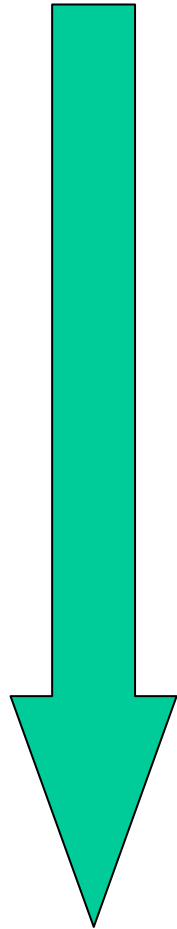
# System Architecture





## Timeline of Sensor Network Research

---



**2000 -**

- **Studies, basic research, small demonstrations**

**2006 -**

- **Prototypical real world applications**
  - **Developed by specialized expert groups**
  - **Very expensive (several person years)**

**???**

- **Broad spectrum of reusable application frameworks**
  - **Programmable by domain experts**
  - **Easy and cheap software and hardware setup**

## Motivation

---

**Huge future market for Wireless Sensor Networks:**

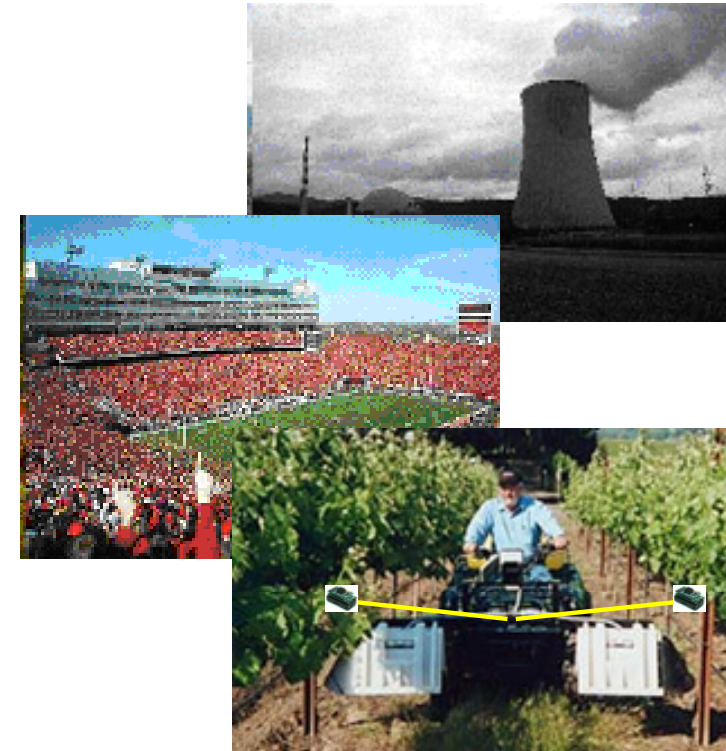
- environmental and structural health monitoring
  - military and homeland security applications
  - control in offices and private households
- Strong security, safety and privacy requirements

**Problem: How to realize and manage this security?**

**Sensor Nodes have severely scarce resources:**

- Energy
- Processing power
- Memory

**Problem: Trade-Off: Security  $\leftrightarrow$  Performance**

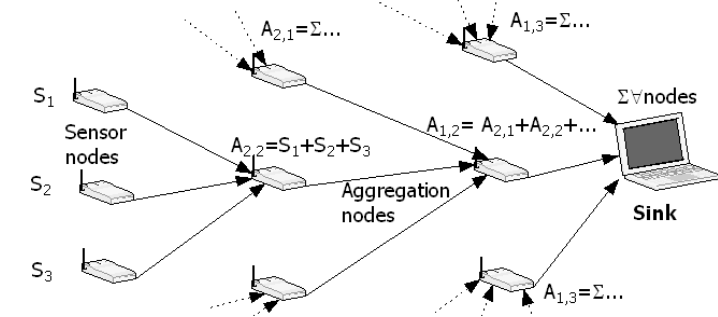


## Motivation (2)



There are a lot of solutions for isolated problems:

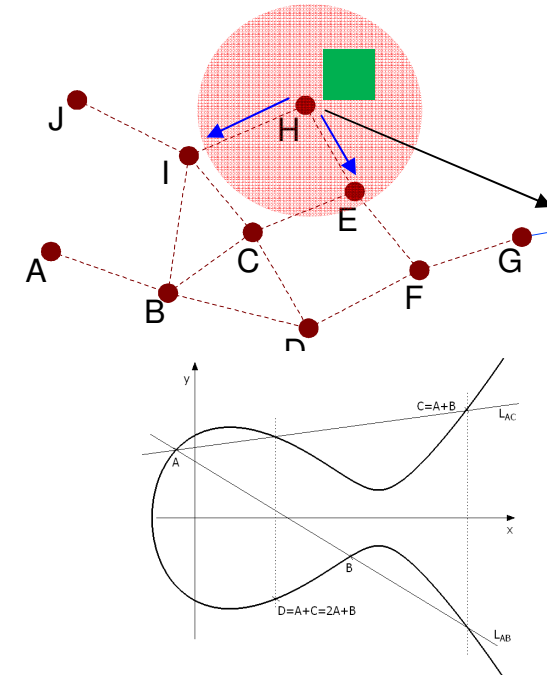
- Secure and dependable routing
- Good encryption
- Dependable and secure distributed data storage



Problem: How to exploit all good solutions in one system?

Implementation of sensor nodes is a lot manual work

- Error-prone
- Not objective
- Needs a lot of time → Expensive



Problem: Secure WSN solutions have to be economically reasonable



# How to design secure and dependable WSNs?

---

- **Today:**

- develop actual application first
- Attach some security stuff later

→no satisfying security & safety  
→Lot of processing overhead

Application

Security Layer

Safety Layer

- **Required:**

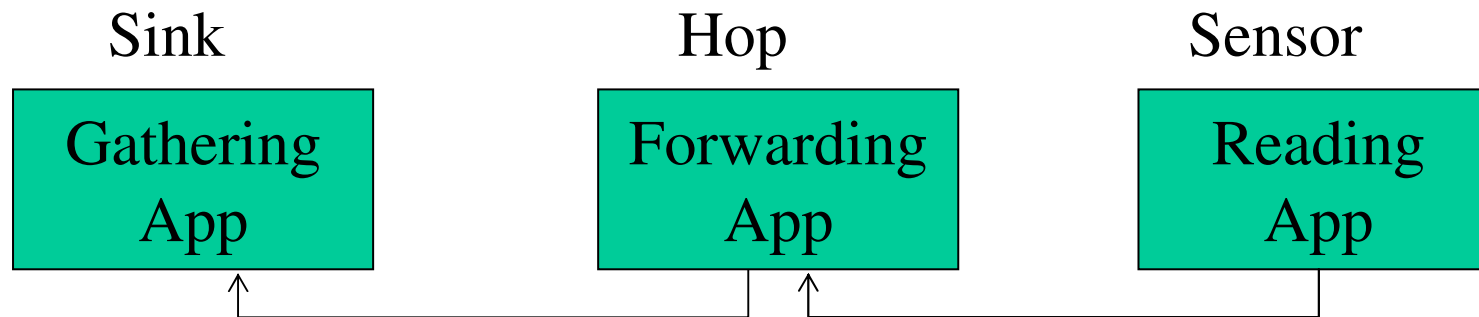
- Development of integrated safe and secure application

→Less protocol overhead  
→But development overhead

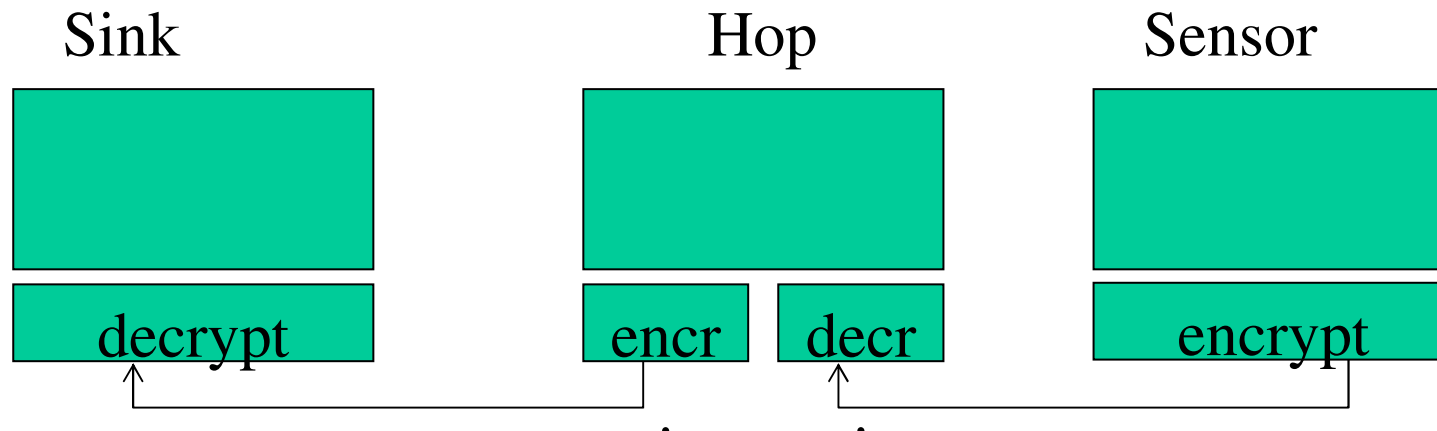
Safe and secure  
Application

## Example for possibly wrong design

Design unprotected application



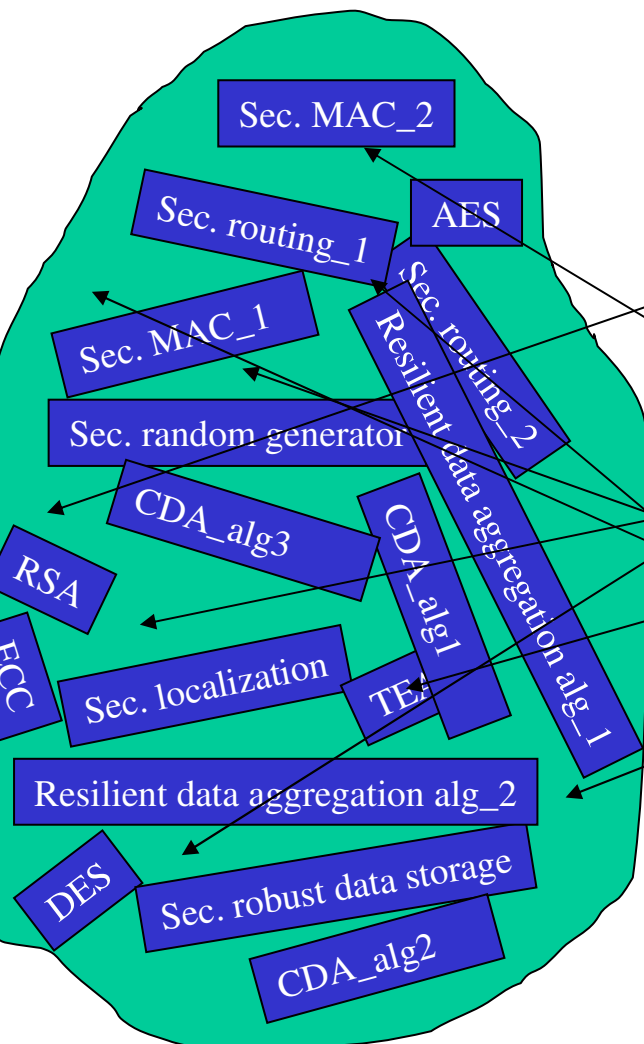
Secure implementation with encryption in MAC



## Goals / Vision



### Bunch of Solutions



Application

Sensor node HW

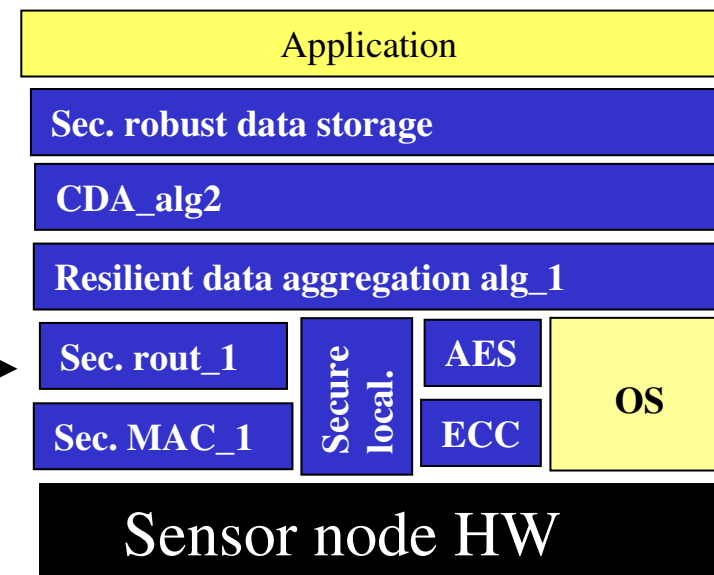
Req.

### Configuration and Management Module

1. Req. vs features of modules
2. Interoperability of modules
3. Security of combination

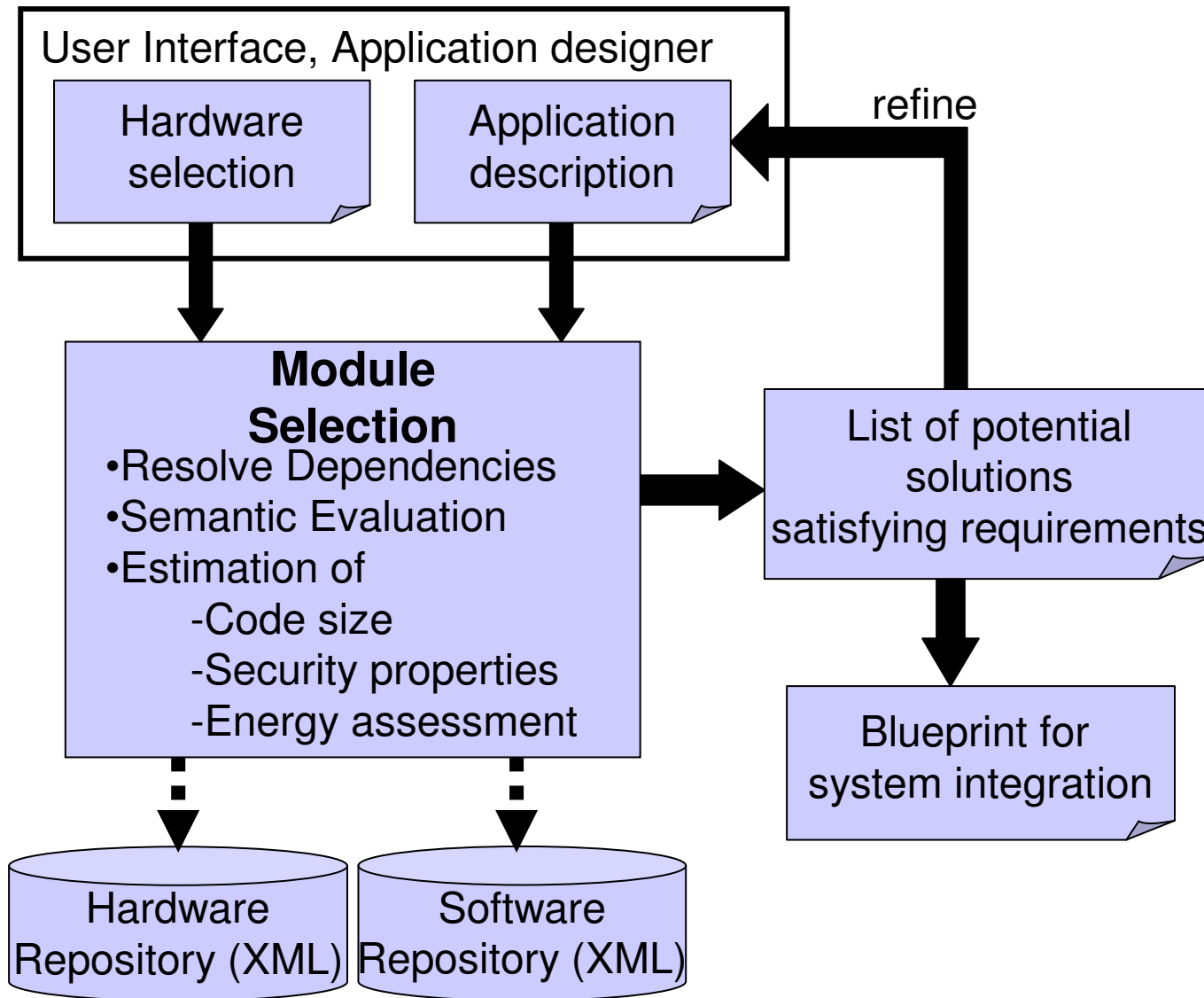
Tailor made security architecture

### System

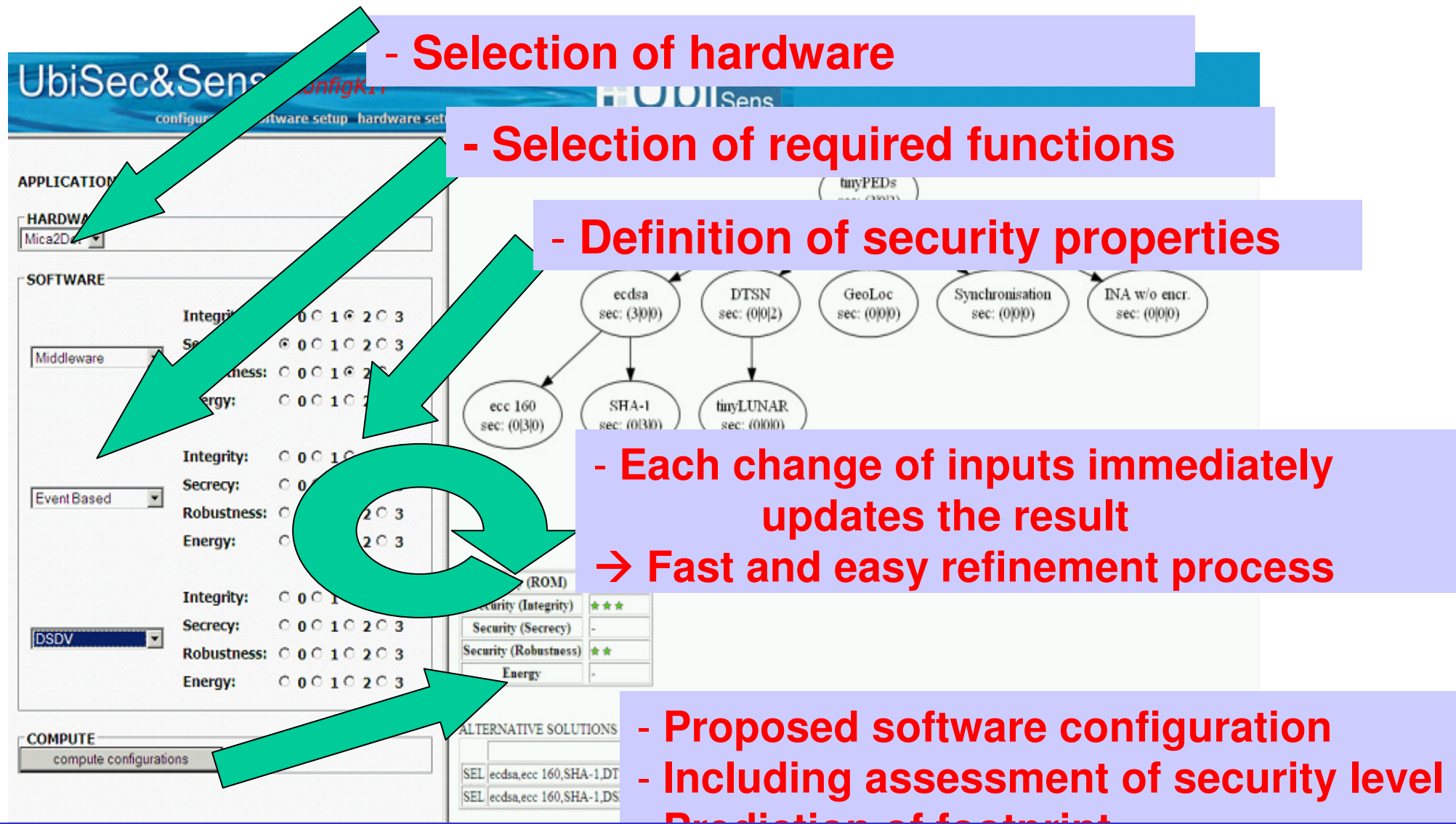




## The configKIT Approach



## configKIT for Application Designers



The screenshot displays the configKIT web interface, which is divided into several sections:

- APPLICATION:** Includes a 'HARDWARE' dropdown menu (currently set to 'Mica2D4') and a 'SOFTWARE' section with various configuration options.
- SOFTWARE:** Contains multiple rows of configuration options for 'Integrity', 'Secrecy', 'Robustness', and 'Energy', each with radio buttons for values 0, 1, 2, and 3. A 'Middleware' dropdown is also present.
- COMPUTE:** Features a 'compute configurations' button.
- Security Properties:** A diagram showing a hierarchy of security properties: 'ecdsa sec: (3|0|0)' and 'DTSN sec: (0|0|2)' at the top, leading to 'ecc 160 sec: (0|3|0)', 'SHA-1 sec: (0|3|0)', and 'tinyLUNAR sec: (0|0|0)' below them. Other properties like 'GeoLoc sec: (0|0|0)', 'Synchronisation sec: (0|0|0)', and 'INA w/o encr. sec: (0|0|0)' are also shown.
- Assessment:** A table showing the security level for different properties:
 

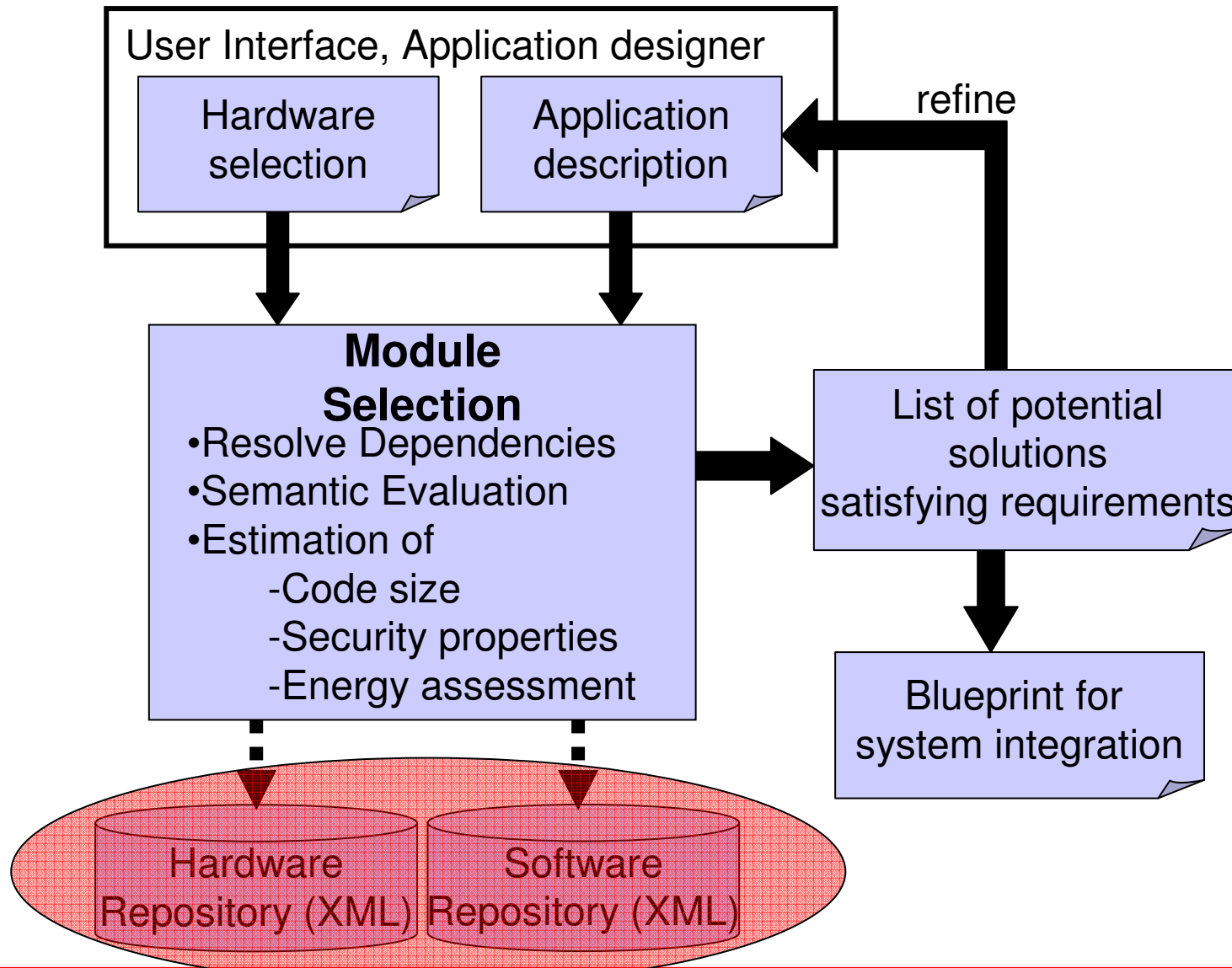
Property	Security (Integrity)	Security (Secrecy)	Security (Robustness)	Energy
Security (Integrity)	★★★	-	★★	-
Security (Secrecy)	-	-	-	-
Security (Robustness)	★★	-	-	-
Energy	-	-	-	-
- ALTERNATIVE SOLUTIONS:** A list of alternative configurations, such as 'SEL ecdsa,ecc 160,SHA-1,DT' and 'SEL ecdsa,ecc 160,SHA-1,DS'.

Four large green arrows point from the text boxes to the corresponding sections in the interface:

- Selection of hardware (points to the HARDWARE dropdown)
- Selection of required functions (points to the SOFTWARE section)
- Definition of security properties (points to the Security Properties diagram)
- Each change of inputs immediately updates the result → Fast and easy refinement process (points to the COMPUTE button)
- Proposed software configuration (points to the ALTERNATIVE SOLUTIONS list)
- Including assessment of security level (points to the Security Assessment table)

Online demo available at: <http://www.ist-ubisecsens.org/downloads/configKIT/configkit.php>

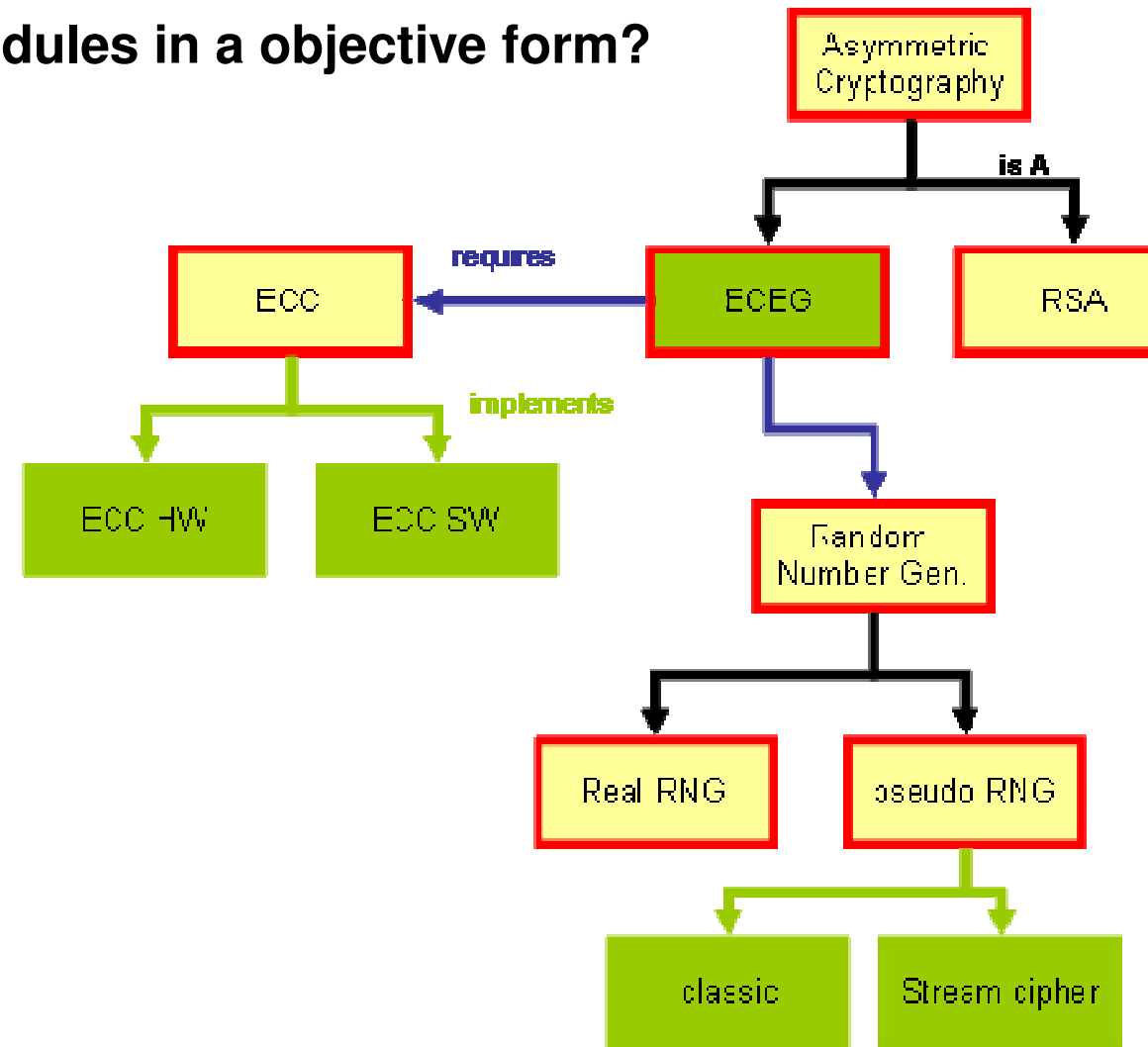
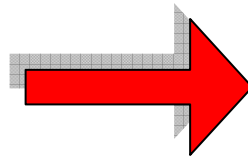
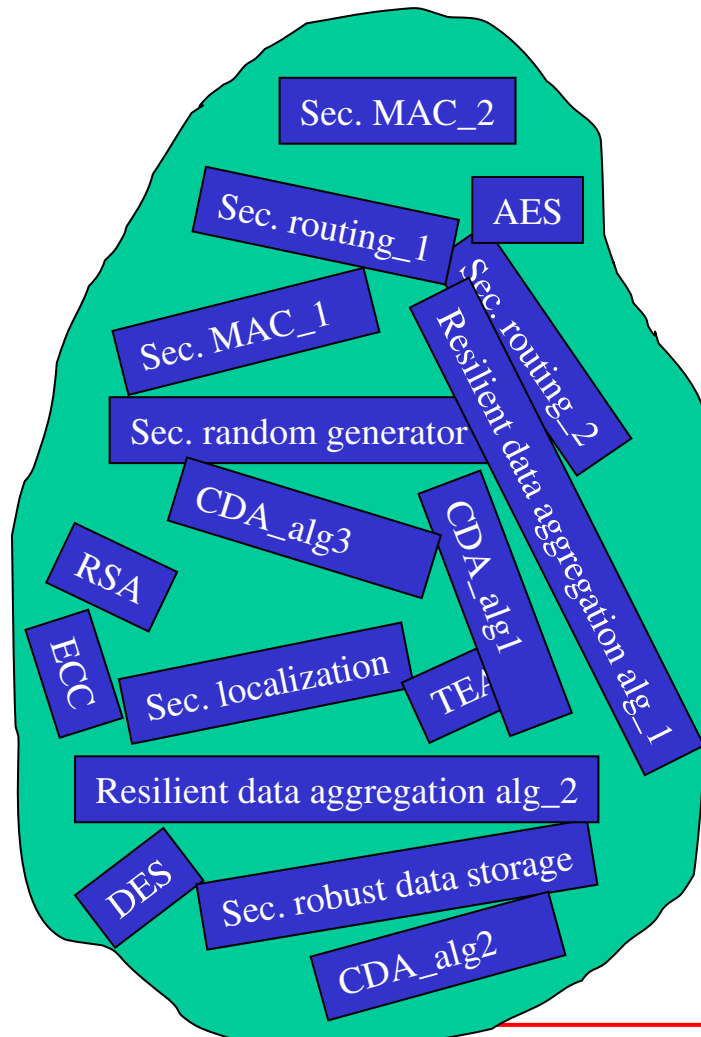
## The configKIT Approach – Setup of Repositories



## Challenge (1)

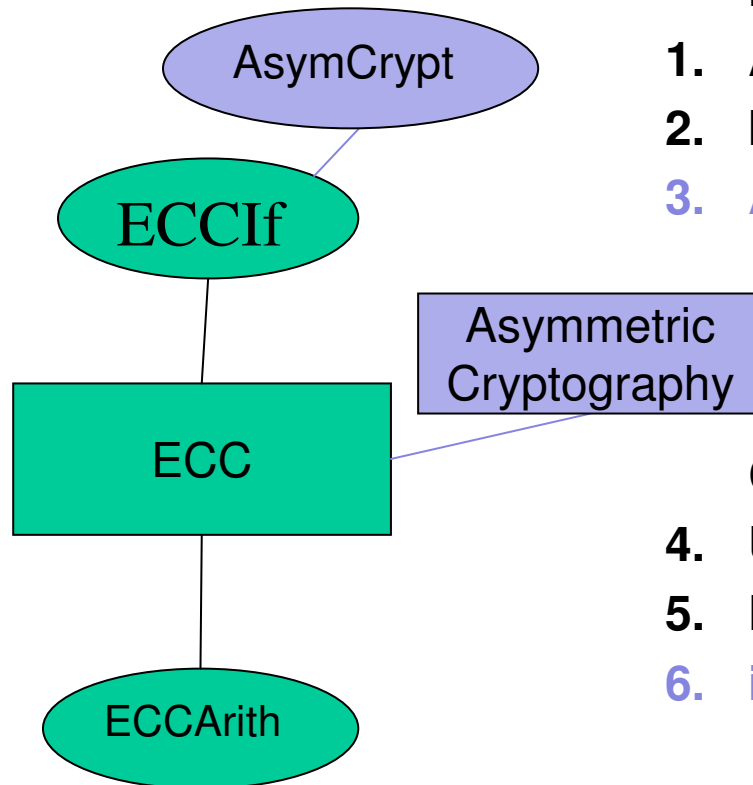


How to present the bunch of modules in a objective form?





# Structure of Software Repository



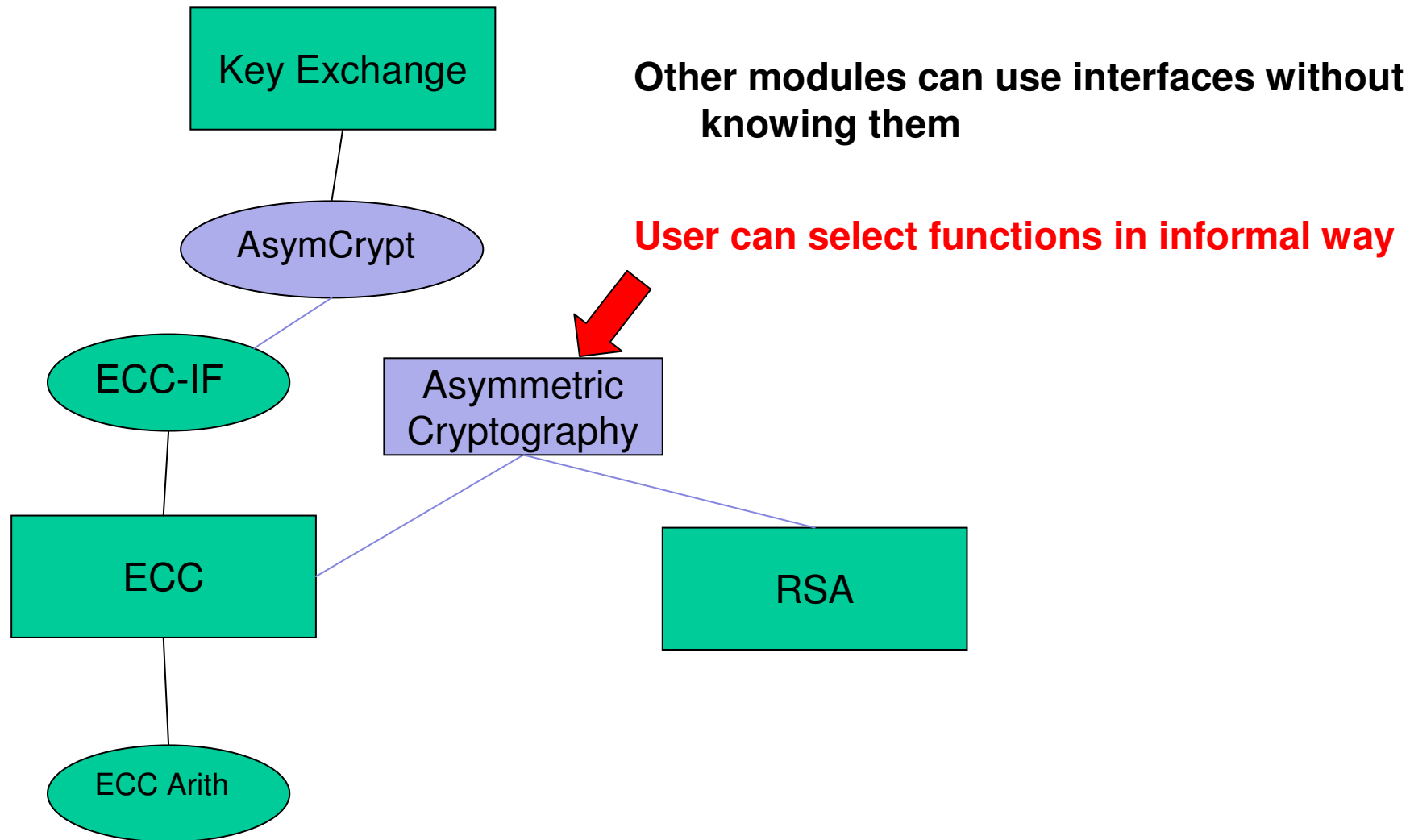
## Nodes:

1. **Actual Module**
2. **Interfaces they use and provide**
3. **Abstract Modules and Interfaces**

## Connections

4. **Uses**
5. **Provides**
6. **isA**

# Structure of Software Repository



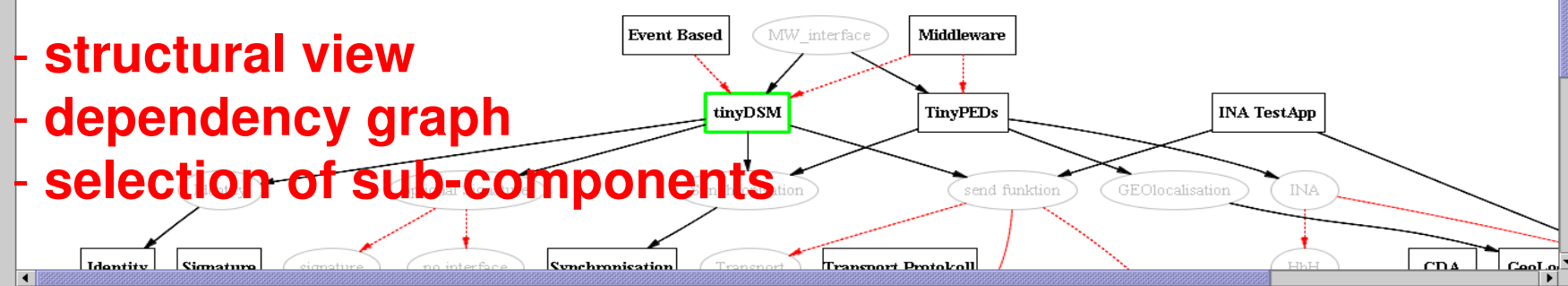
# Setup of Software Repository

Applet Viewer: ckltgui.StartApplet.class

Applet

hardware setup software setup

- structural view
- dependency graph
- selection of sub-components



tinyDSM

tinyDSM

M006

eventbased Middleware

yes

size 9500

IF004 MW\_int...

IF012 Synchronisatic

M025 Event Based

provide

uses

is

IF004 MW\_inte...

IF010 optional signatu...

IF009 send function

IF014 Identity

IF012 Synchronisation

M024 Middleware

M025 Event Based

add module

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ModuleType SYSTEM "module.dtd">
<ModuleType>
  <Module>
    <Name>tinyDSM</Name>
    <ID>M006</ID>
    <Description>eventbased Middleware</Description>
  </Module>
  <ModuleAttribute>
    <Implementation>yes</Implementation>
    <Provides>
      <Provide>IF004</Provide>
    </Provides>
    <Uses>
      <Use>IF010</Use>
      <Use>IF009</Use>
      <Use>IF014</Use>
      <Use>IF012</Use>
    </Uses>
    <Is>
      <is>M024</is>
      <is>M025</is>
    </Is>
    <Size>9500</Size>
  </ModuleAttribute>
</ModuleType>
```

Setting of parameters, dependencies, requirements

Re-usable XML, part of software repository

Applet started.

## Example Module Description

---

```
<SoftwareComponentType
  Description="Elliptic Curve Digital Signature Algorithm"
  IsStatic="true" Name="ECDSA" Version="0.1">
  <EnergyConsumption>0</EnergyConsumption>
  <CodeMemorySize>1468</CodeMemorySize>
  <DataMemorySize>540</DataMemorySize>
  <PersistentMemorySize>0</PersistentMemorySize>
  <Provides>
    <SoftwareInterfaceType Alias="ECDSA" SoftwareInterfaceTypeId="" />
  </Provides>
  <Uses>
    <SoftwareInterfaceType Alias="ECC" SoftwareInterfaceTypeId="" />
  </Uses>
  <SecurityParameter Name="Integrity" Value="4"/>
  <SecurityParameter Name="Concealment" Value="2"/>
  <SecurityParameter Name="Robustness" Value="1"/>
</SoftwareComponentType>
```

# Setup of Hardware Repository

Applet Viewer: ckittgui.StartApplet.class  
Applet

hardware setup software setup

**Structural view/  
selection of  
sub-components**

```

graph TD
    TestNode((TestNode)) --> actuators((actuators))
    TestNode --> wiredcomm((wiredcomm))
    TestNode --> processor((processor))
    TestNode --> wirelesscomm((wirelesscomm))
    TestNode --> memory((memory))
    TestNode --> sensors((sensors))
    sensors --> SHT10((SHT10))
  
```

**Parametrisation**

Name	SHT10
SensorTypeID	SENS00001
Description	Humidity accuracy: +-4.5%RH Temperature accuracy +-0.5°C
DataType	unit8
Unit	%RH / °C

TimeToSample: 100 ms

EnergyConsumptionPerSample: 100 nJ

SamplingRate: 100 Hz

modify ok

**Re-usable XML,  
part of  
hardware repository**

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE SensorType SYSTEM "sensor.dtd">

<SensorType>
  <Sensor>
    <Name>SHT10</Name>
    <SensorTypeID>SENS00001</SensorTypeID>
    <Description>Humidity accuracy: +-4.5%RH Temperature accuracy +-0.5°C</Description>
    <DataType>unit8</DataType>
    <Unit>%RH / °C</Unit>
  </Sensor>
  <SensorAttribute>
    <TimeToSample unit="ms">100</TimeToSample>
    <EnergyConsumptionPerSample unit="nJ">100</EnergyConsumptionPerSample>
    <SamplingRate unit="Hz">100</SamplingRate>
  </SensorAttribute>
</SensorType>
  
```

Applet started.



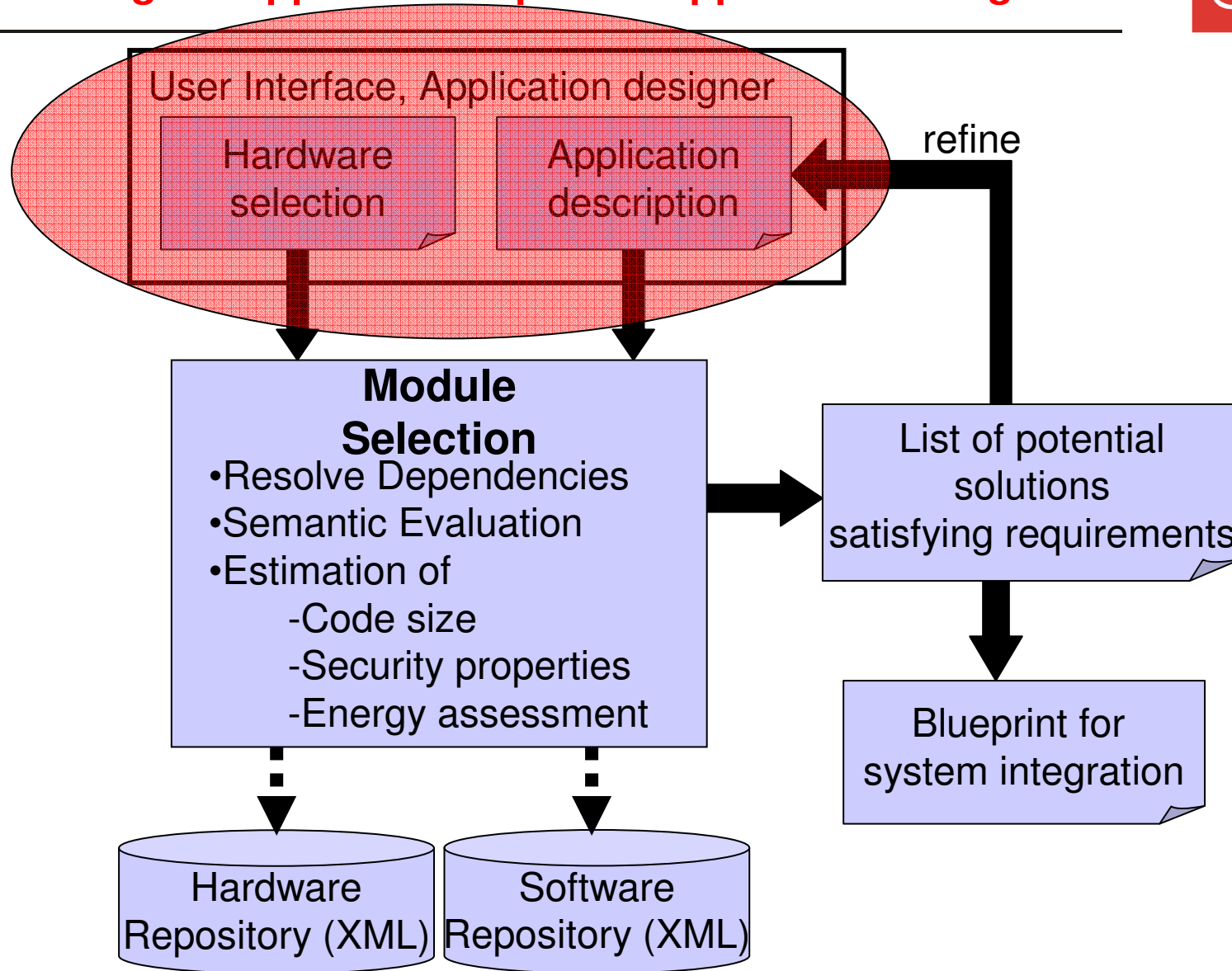
## Example Hardware Description

---



```
<SensorType>
  <Sensor>
    <Name>SHT11</Name>
    <SensorTypeID>10011</SensorTypeID>
    <Description>Humidity accuracy: +-3.0%RH  Temperature accuracy +-0.4°C</Description>
    <DataType>unit8</DataType>
    <Unit>%RH / °C</Unit>
  </Sensor>
  <SensorAttribute>
    <TimeToSample unit="ms">10</TimeToSample>
    <EnergyConsumptionPerSample unit="mJ">10</EnergyConsumptionPerSample>
    <SamplingRate unit="Hz">10</SamplingRate>
  </SensorAttribute>
</SensorType>
```

## The configKIT Approach – Input for Application Designer



## Challenge (2) – App description & Module Selection



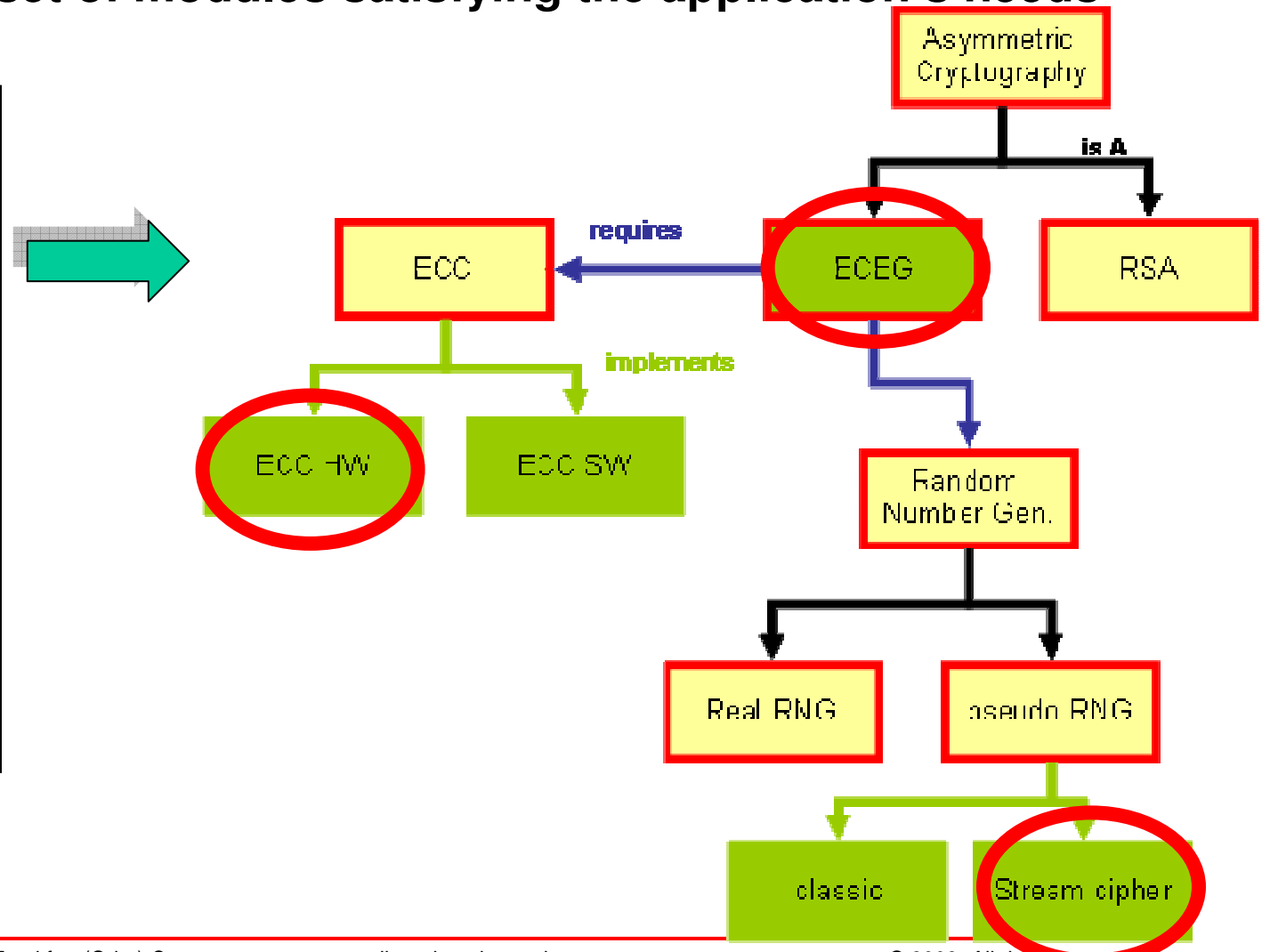
### Selection of the set of modules satisfying the application's needs

#### Application:

- Need for Public Key Cryptography
- 20 operations per hour
- Life time > 6 months
- Hardware: MicaZ

#### Security Requirements:

- Data secure for 20 years



## Selection: Current State

- Selection of Hardware
- Selection of software modules  
functional description  
or explicit modules  
  
+ required parameters

### APPLICATION NEEDS:

#### HARDWARE

MicaZ

#### SOFTWARE

Event Based

Integrity: ☐ 0 ☐ 1 ☐ 2 ☐ 3

Secrecy: ☐ 0 ☐ 1 ☐ 2 ☐ 3

Robustness: ☐ 0 ☐ 1 ☒ 2 ☐ 3

Energy: ☐ 0 ☐ 1 ☐ 2 ☐ 3

Signature

Integrity: ☐ 0 ☐ 1 ☐ 2 ☒ 3

Secrecy: ☐ 0 ☐ 1 ☐ 2 ☐ 3

Robustness: ☐ 0 ☐ 1 ☐ 2 ☐ 3

Energy: ☐ 0 ☐ 1 ☐ 2 ☐ 3

DSDV

Integrity: ☐ 0 ☐ 1 ☐ 2 ☐ 3

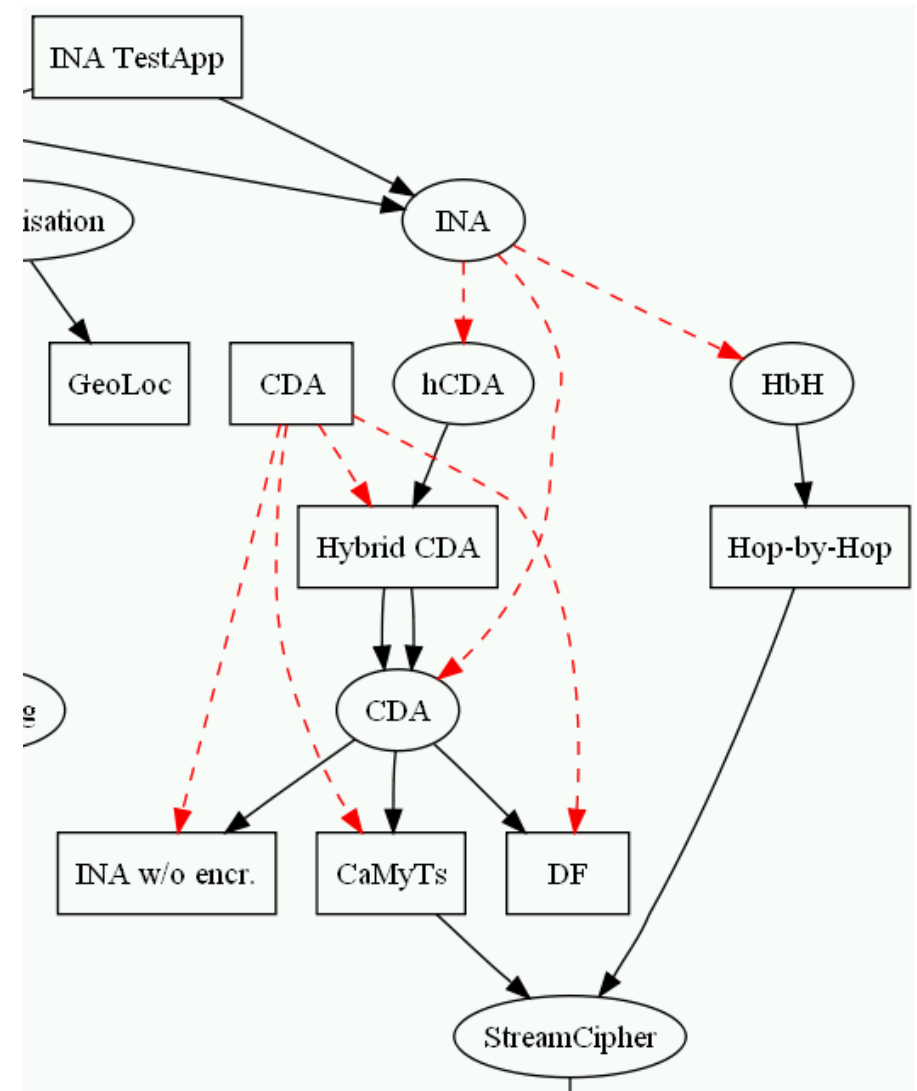
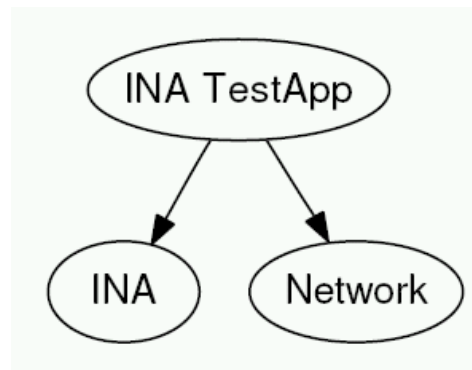
Secrecy: ☐ 0 ☐ 1 ☐ 2 ☐ 3

Robustness: ☐ 0 ☐ 1 ☐ 2 ☐ 3

Energy: ☐ 0 ☐ 1 ☐ 2 ☐ 3

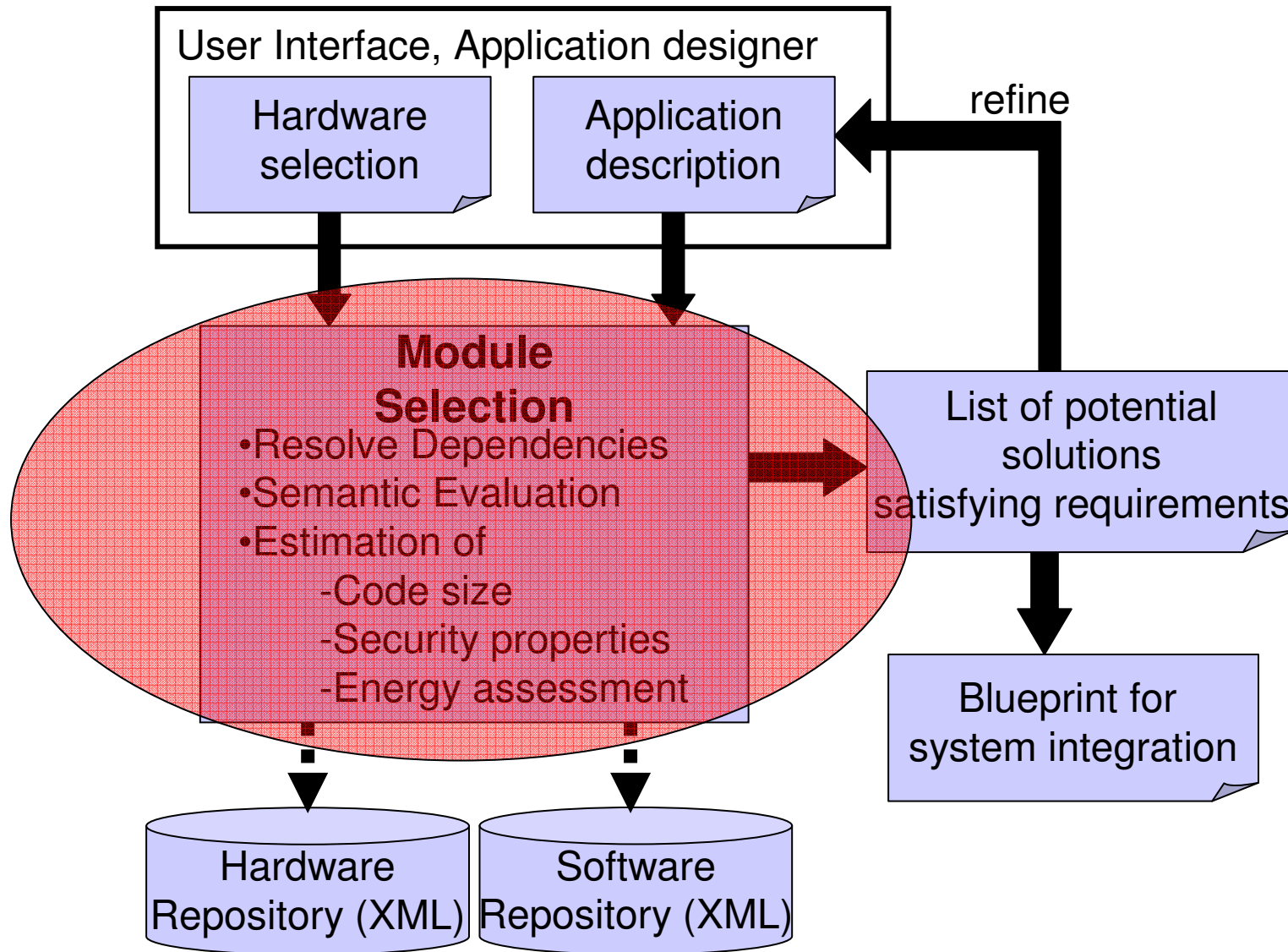
## What's the application anyway?

**Application is just another software  
Module as part of the  
Software Repository**

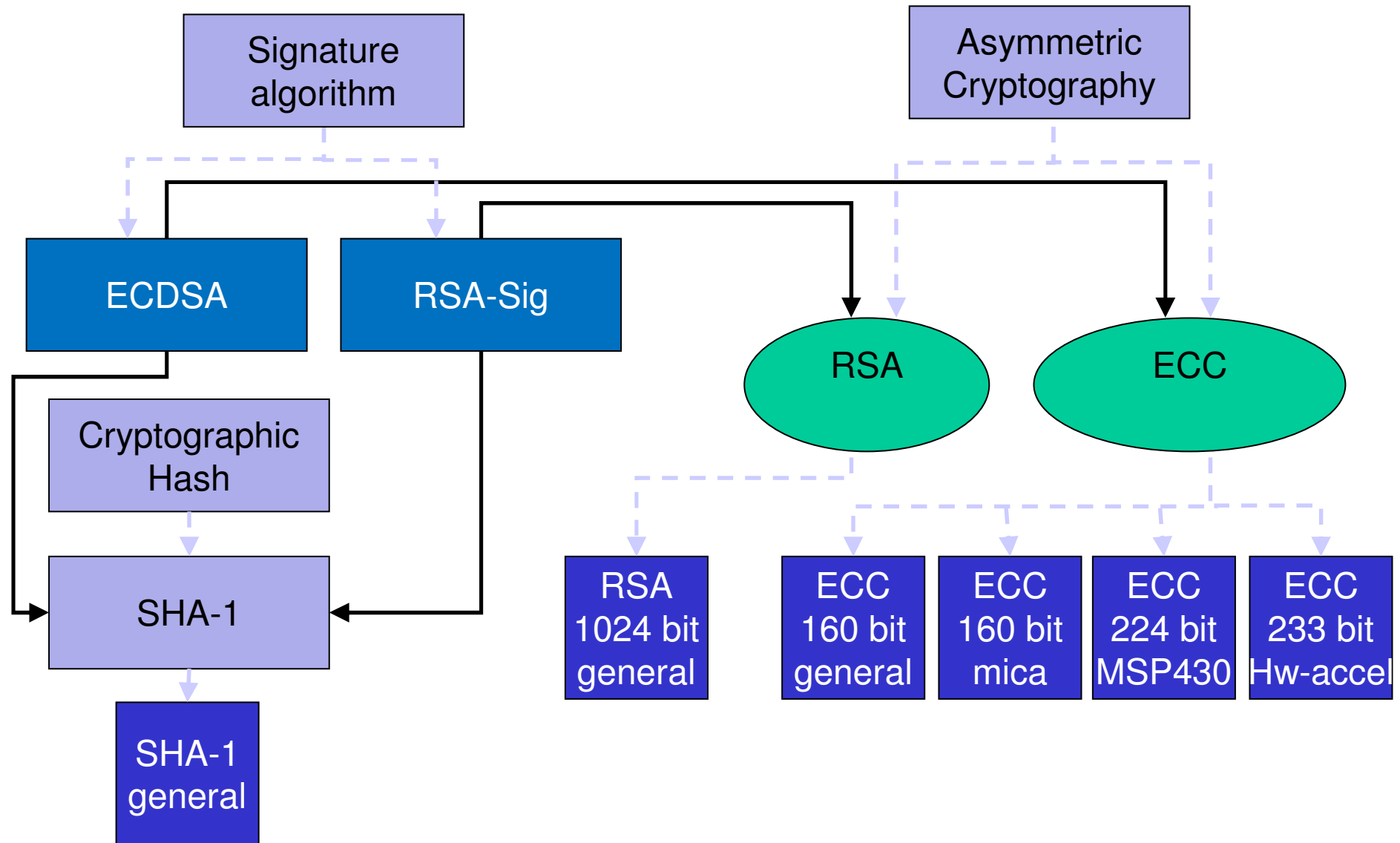




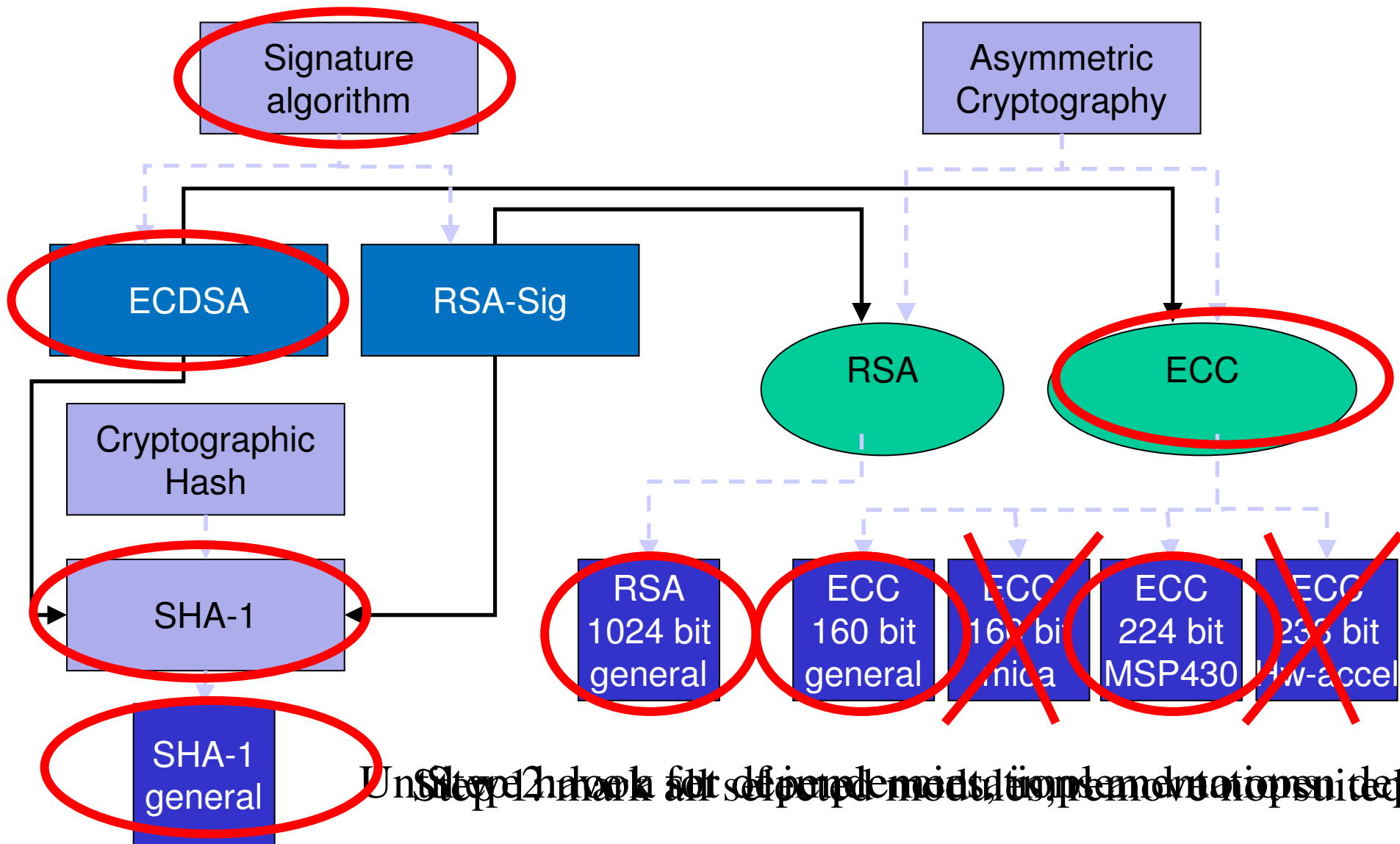
## The configKIT Approach – Module Selection



## Example (1)



## Example (2) – Signature algorithm on MSP430



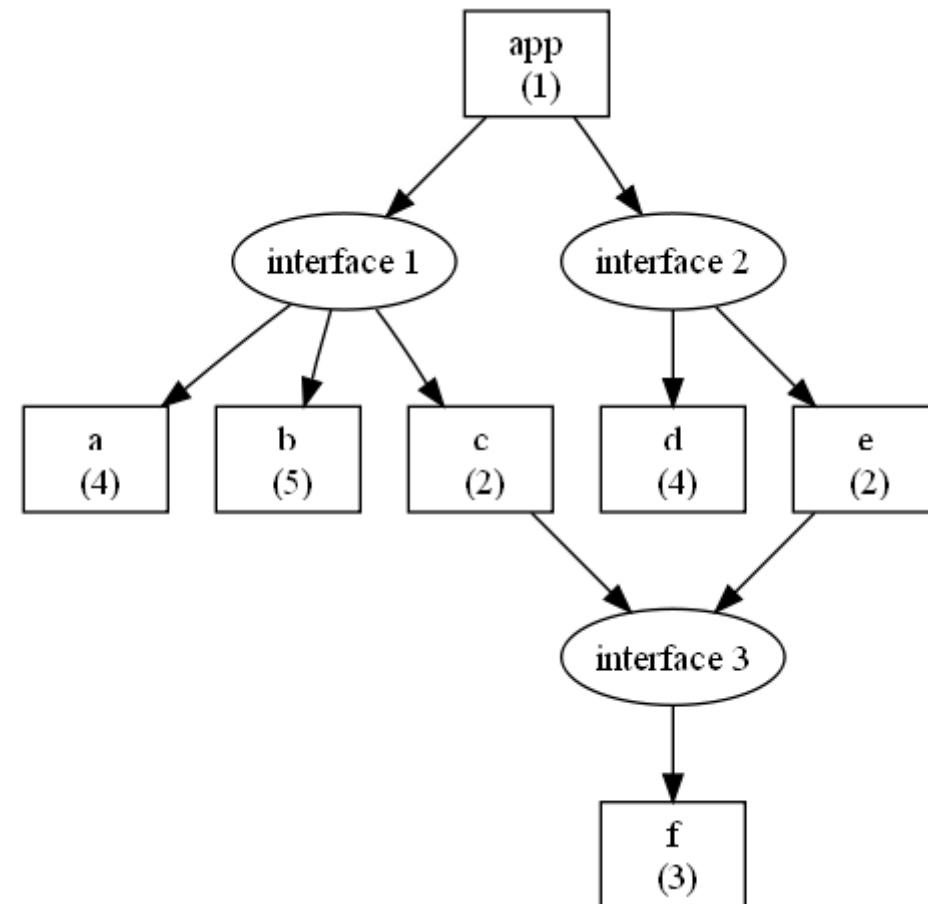
- Original algorithm is NP complete  
→ Optimizations required

## Approach:

- Remember decisions
  - Do not follow unbeneficial sub-trees more than once

## →Problems:

- Re-convergences (no actual tree)
- how make sure unbeneficial sub-tree is not better for another pre-selection?





ActiveMessageAddress

TinyPEDSApp

Alarm

Arbi

TinyPEDS

Boot

StdControl

Dissemination

Sensor

Aggregator

IdleTimer

DisseminationCache

Send

ECCArith

ECElGamal

RANBAR

FromHandler

Socket

SplitControl

Identity

VisualDebug

KeyInfoWriter

Identity

Queue

Pool

PacketAcknowledgements

Init

Timer

Leds

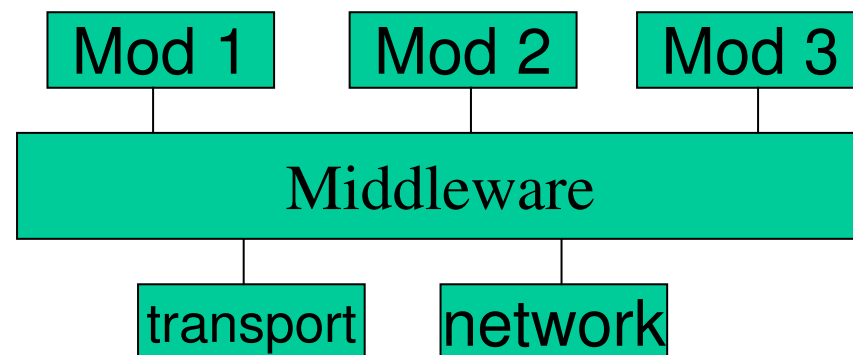
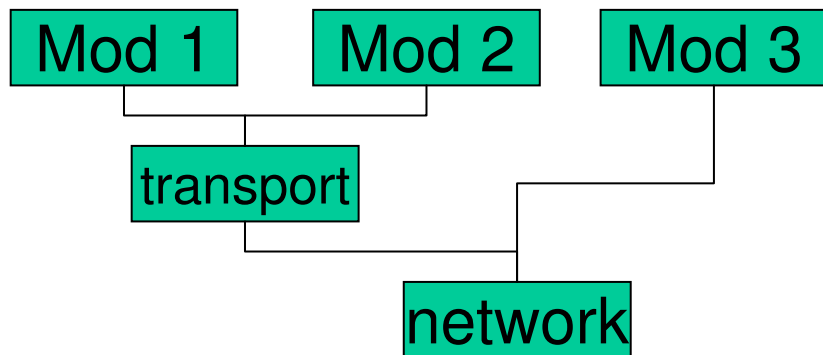
VisualDebug

Leds

## How to solve the model-reality-gap

---

- **Simplify reality**
  - **Unified interfaces**
  - **Backbone operating system/message controller/middleware**



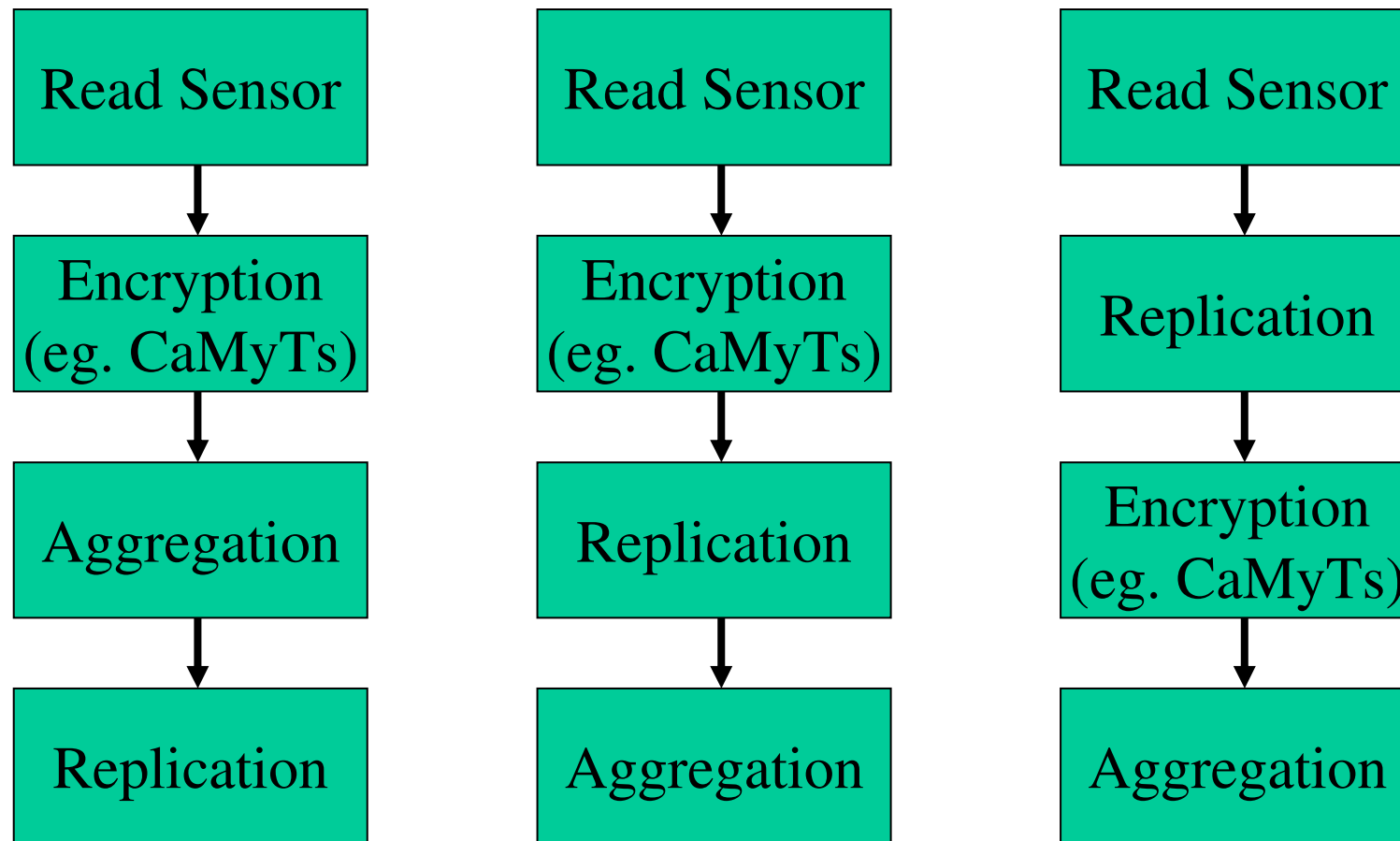
- **Increase complexity of models**

- **Memory:**
  - Addition of single modules
  - Problem:  $\text{size}(A+B) \neq \text{size}(A)+\text{size}(B)$ 
    - Simple addition is rather an approximation (upper bound)
- **Energy:**
  - currently qualitatively (good, medium, bad)
    - Allows comparison of similar protocols
    - Not yet satisfying
  - Required: prediction of actual energy consumption (uJ/op)
    - A lot of issues!
- **Security & Dependability:**

## Challenge (3) – Proof of Security

Which flow is the best and why?

- Dependability?
- Concealment?



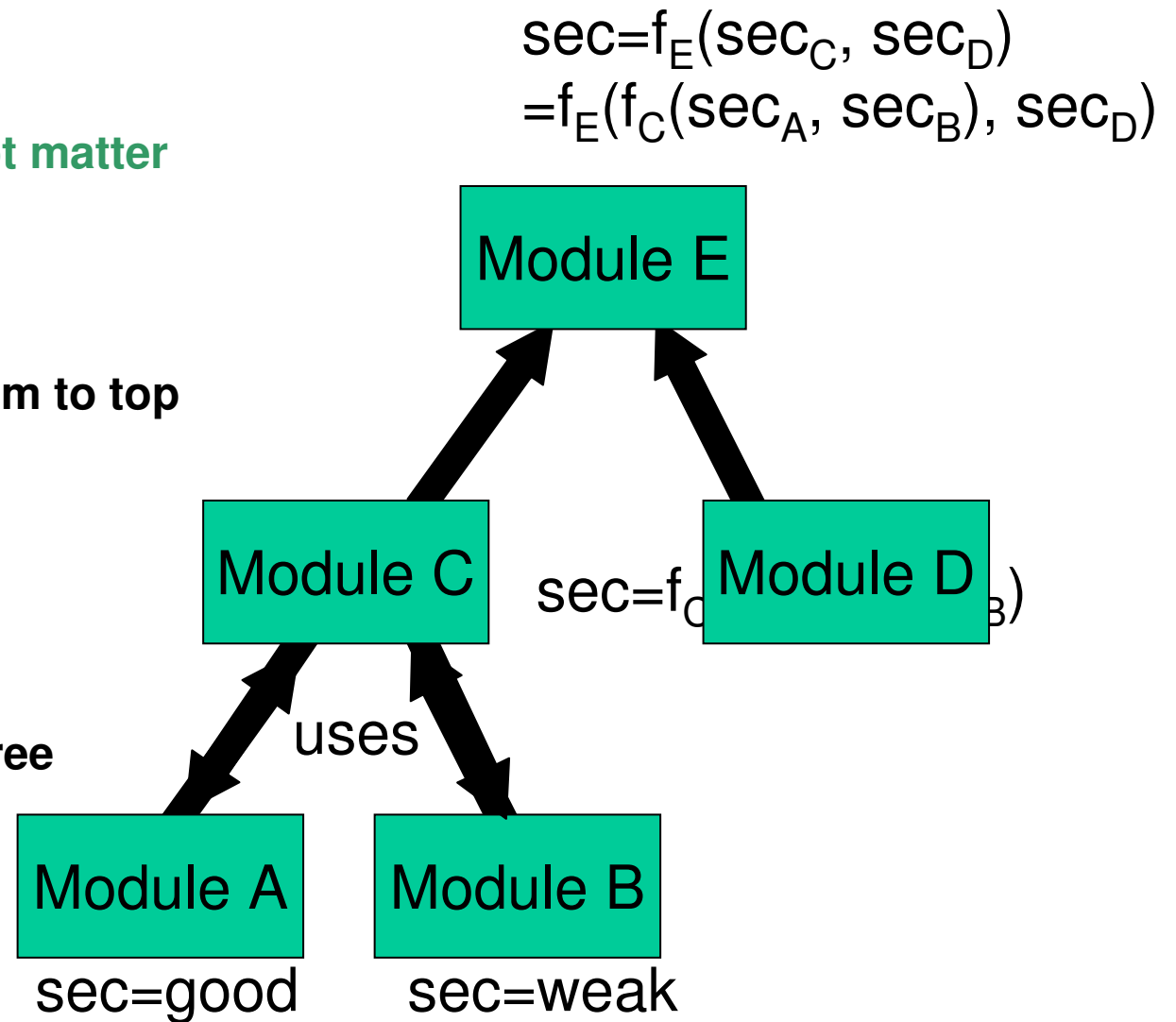
## Proof of Security (2) – How it is currently done

---



- straightforward logic: (probabilistic security?)  
secure module + not secure module = ??
  - For concealment:  
secure + not secure = not secure (the weakest module)
  - For robustness (replication):  
secure + not secure = secure (one replication is ok)
  - For integrity:  
secure + not secure = secure (one proof of integrity is ok)
- NOT REALLY SUFFICIENT → how to do it better?

- **Security metric:**
  - 0 = no security/does not matter
  - 1 = weak
  - 2 = good
  - 3 = strong
- **Propagation from bottom to top of system**
- **Security properties propagate through the dependency graph**
- **Currently support of three security properties:**
  - Secrecy
  - Integrity
  - Robustness





## Security Metric

---

Security class	Attacker	Attacker tools	Budget
0	No security	attack can be succeed 'by accident'	
1	curious hacker	common tools	< 10,000\$
2	organized attacker (academic, crime)	special tools	< 100,000\$
3	large organized attacker (crime, government)	highly specialized tools, laboratory	> 100,000\$

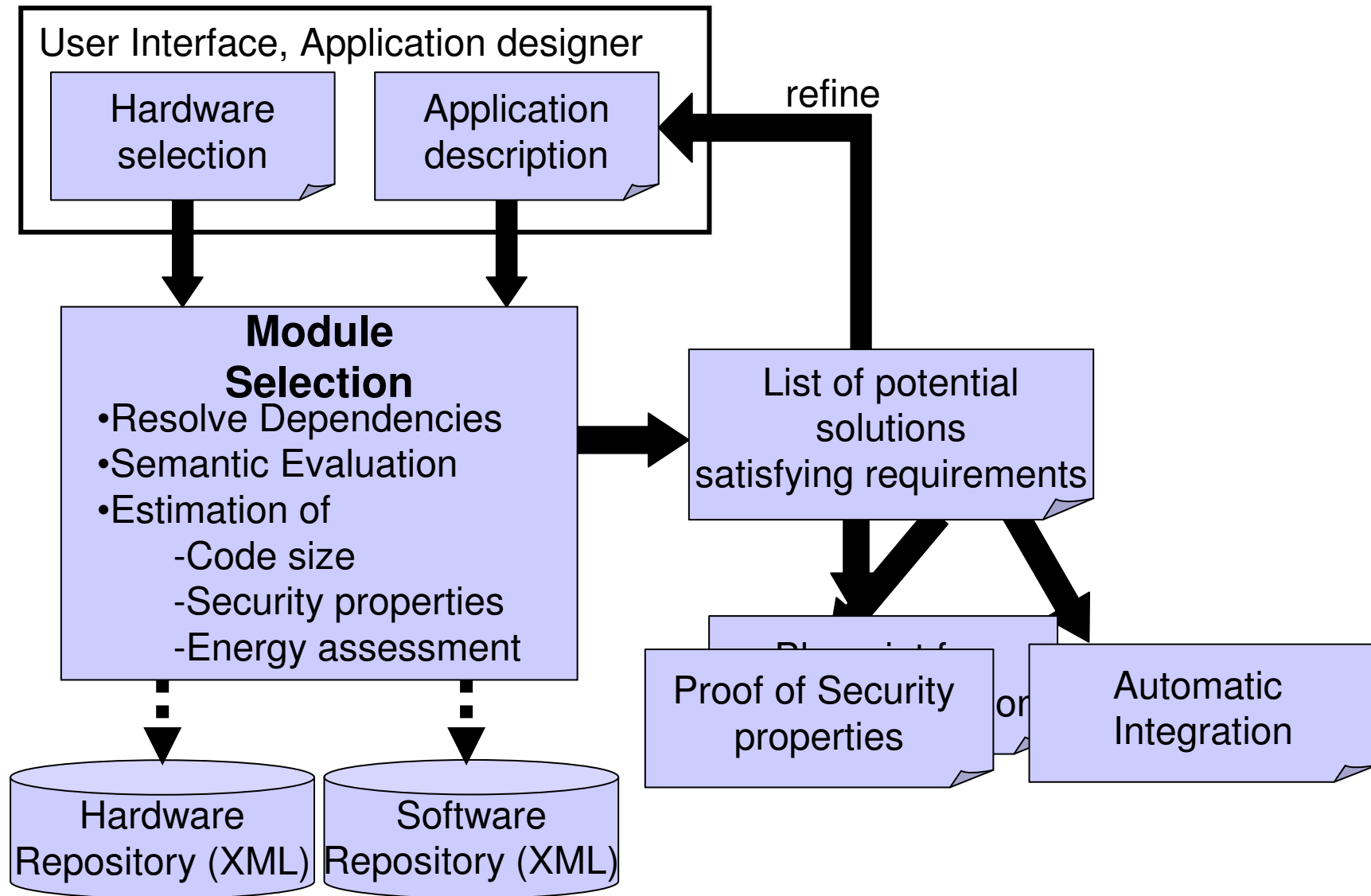
An algorithm belongs to class  $c$  if  
it resists all attacks from attacker groups smaller than  $c$ .

## Similar metrics for other properties?

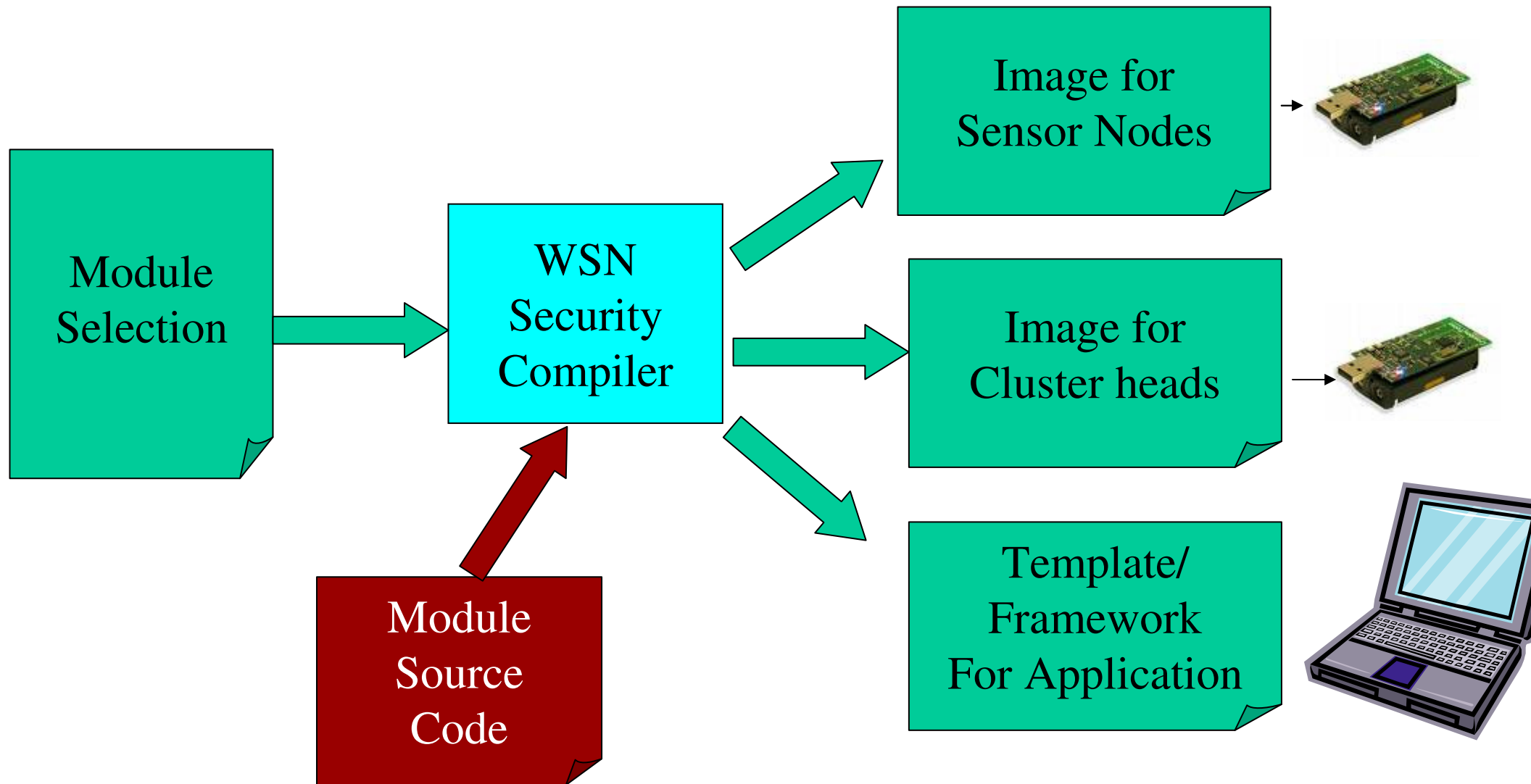
---

- **Dependability / Safety?**
- **Maintainability?**
- **Energy consumption?**
- **Memory consumption?**

## The configKIT Approach – further work



## Challenge (4) – Automatic Integration



## Conclusions

---

- **Wireless Sensor (and Actuator) Networks are needed!**
- **Design of software is too difficult and expensive**
- **What's missing is a unified middleware or engineering approach**
- **configKIT approach can help**
- **Done:**
  - way of module description
  - Selection algorithm
- **ToDo:**
  - Find better metrics for estimation of properties
  - Find a way to verify security and safety properties
  - Automatic integration

# Thank You

---



## Questions?

[peter@ihp-microelectronics.com](mailto:peter@ihp-microelectronics.com)