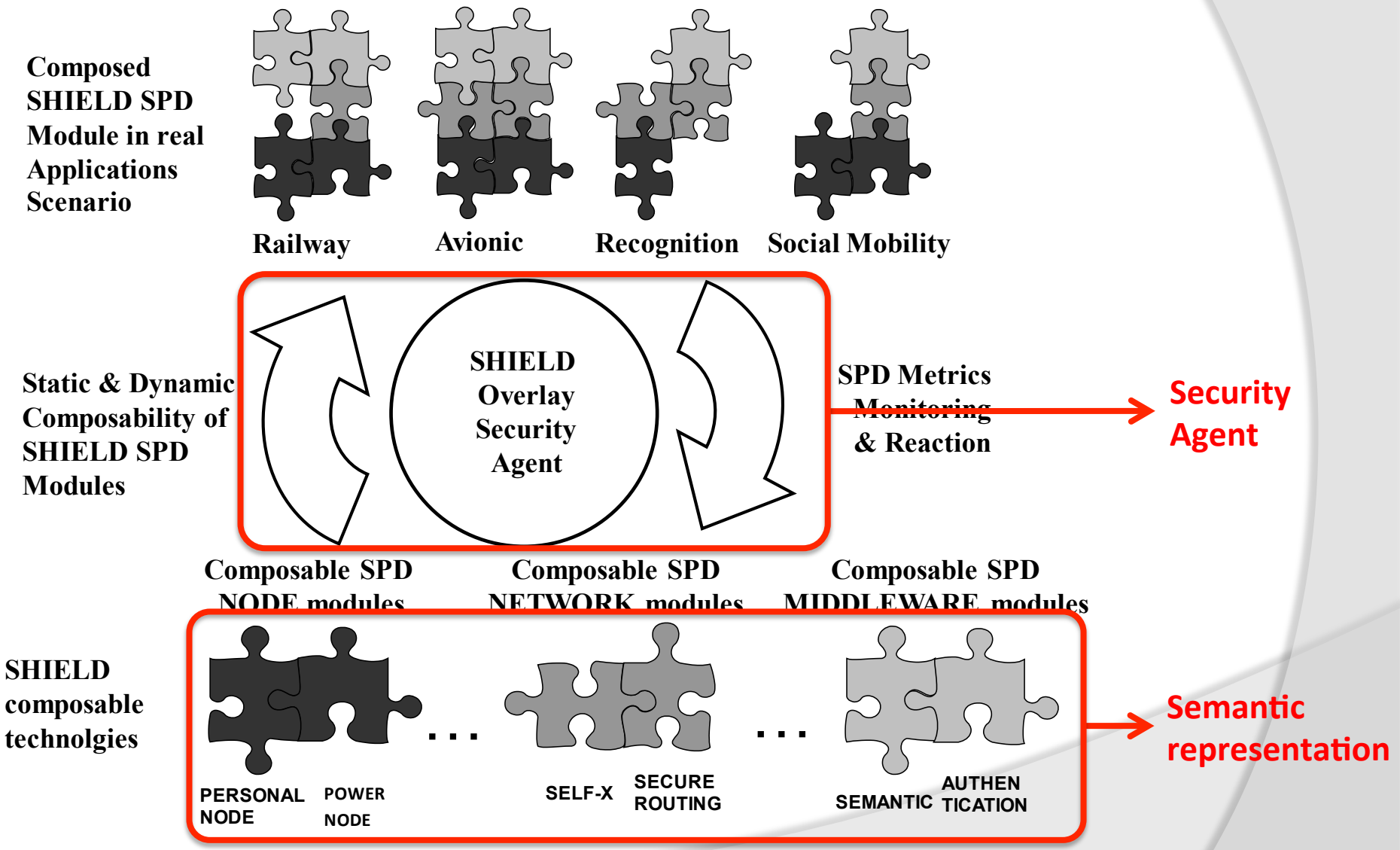
Annual review ROME 2012

Overlay composability – Lesson Learned

Andrea Fiaschetti – University of Rome "La Sapienza"

Overlay Composability: enabling technologies



Semantic representation: lesson learned

Limits of pSHIELD approach to be overcome

- Result valid only for the demonstration purposes (reduced scenario, small set of components)

- Too much queries to go through the whole chain

- Metamodels associated to the physical system are too specific: huge initial effort to translate the description of the various components.

- Metamodels associated to the physical system are not scalable

Good heritage

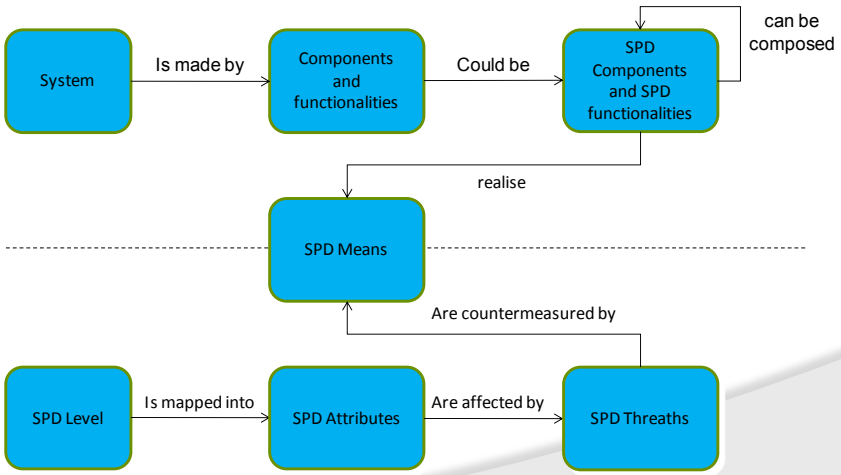
- The modelling procedure derived from the Common Criteria

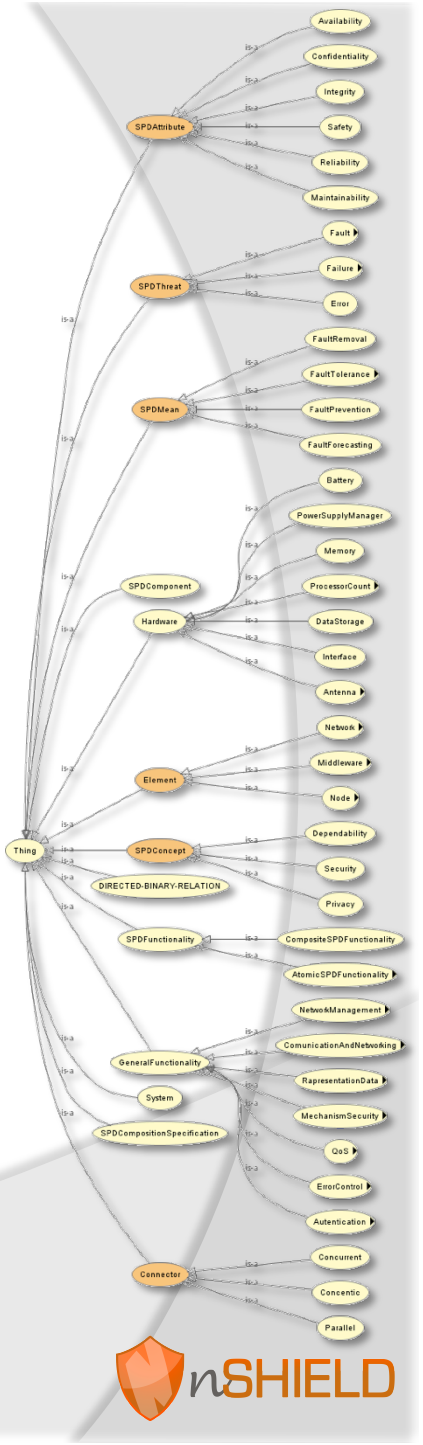
> SPD Level

> Attributes

> Threats

> Means

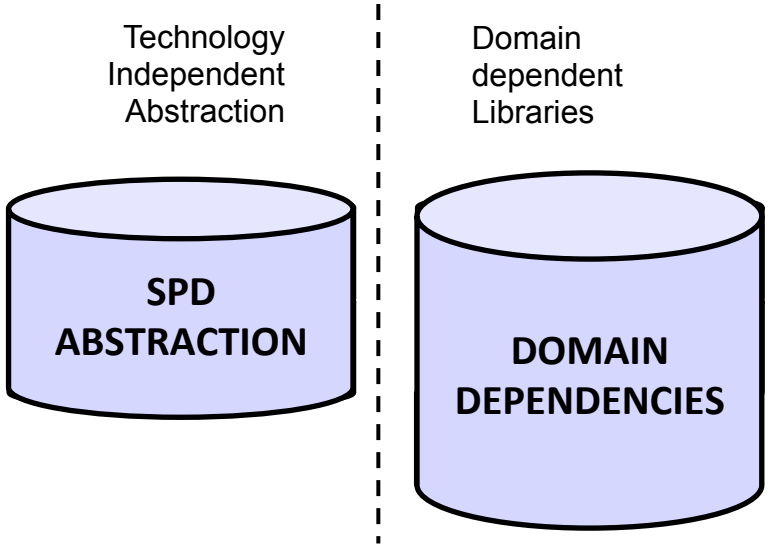




nSHIELD

Semantic representation: nSHIELD achievements 1/4

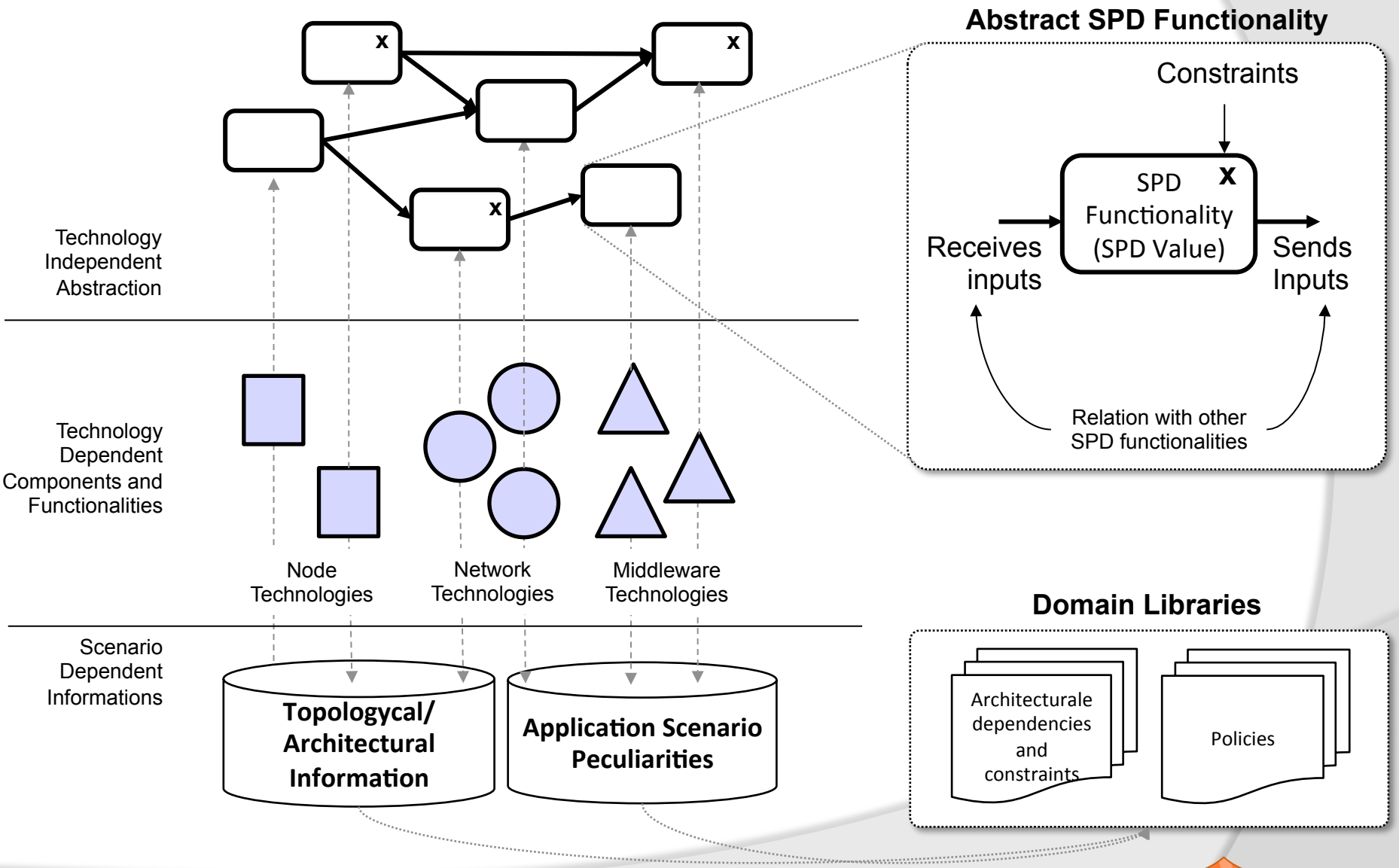
- Axiom: it is not possible to represent the whole system knowledge with a single model
- Objective: balance between expressiveness and flexibility
- Solution: decoupling between
  - Information used to compute the configuration of SPD functionalities and
  - Information used to implement the configuration and tailor it to the scenario needs



- SPD Abstraction is natively present into a SHIELD component
- Domain dependencies are manually configured during system deployment

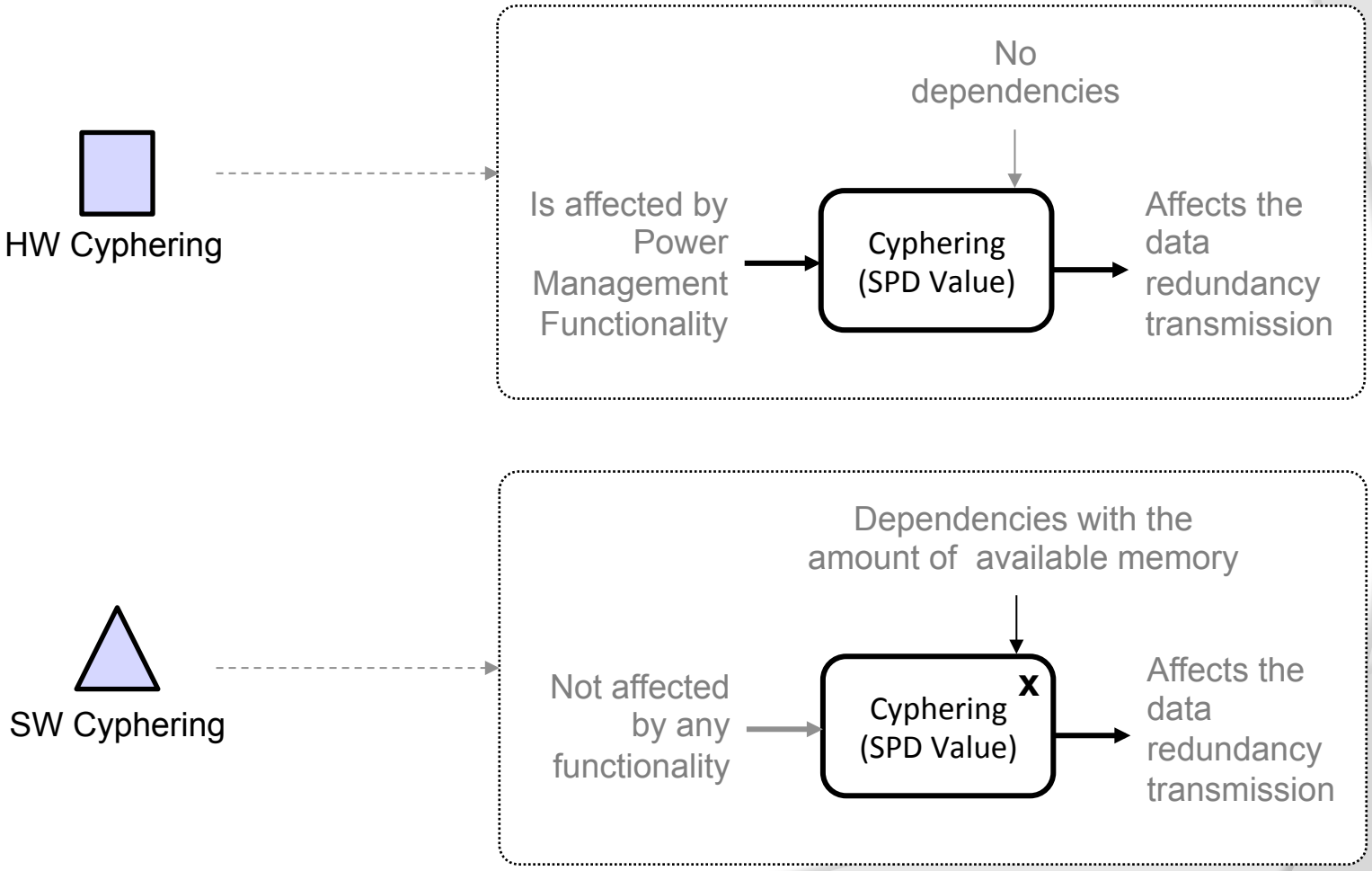
nSHIELD

Semantic representation: nSHIELD achievements 2/4



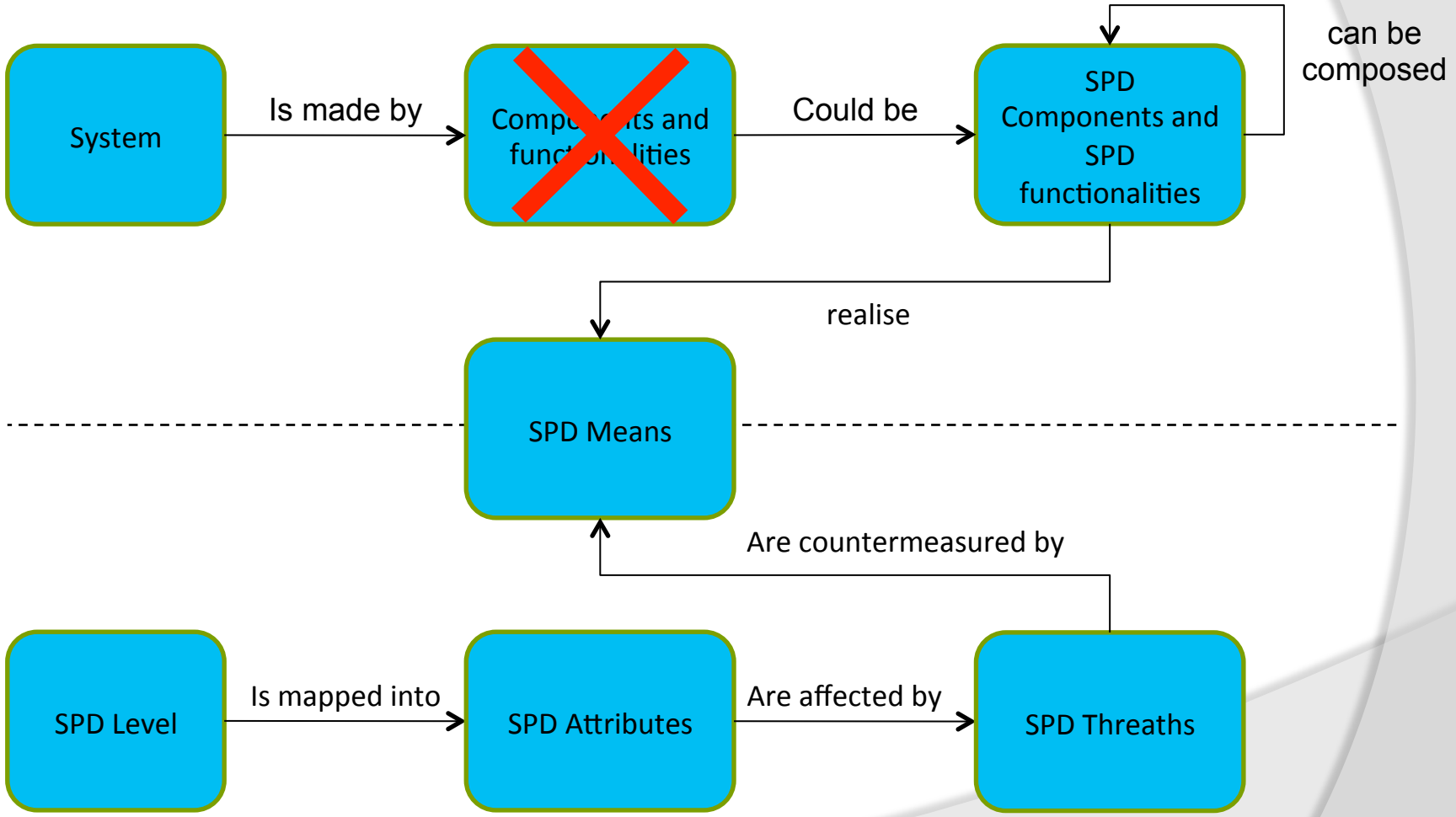
Semantic representation: nSHIELD achievements 3/4

Example of technology independent abstraction



Semantic representation: nSHIELD achievements 4/4

One step reduction: less queries



Security Agent: Lesson Learned

Limits of pSHIELD approach to be overcome

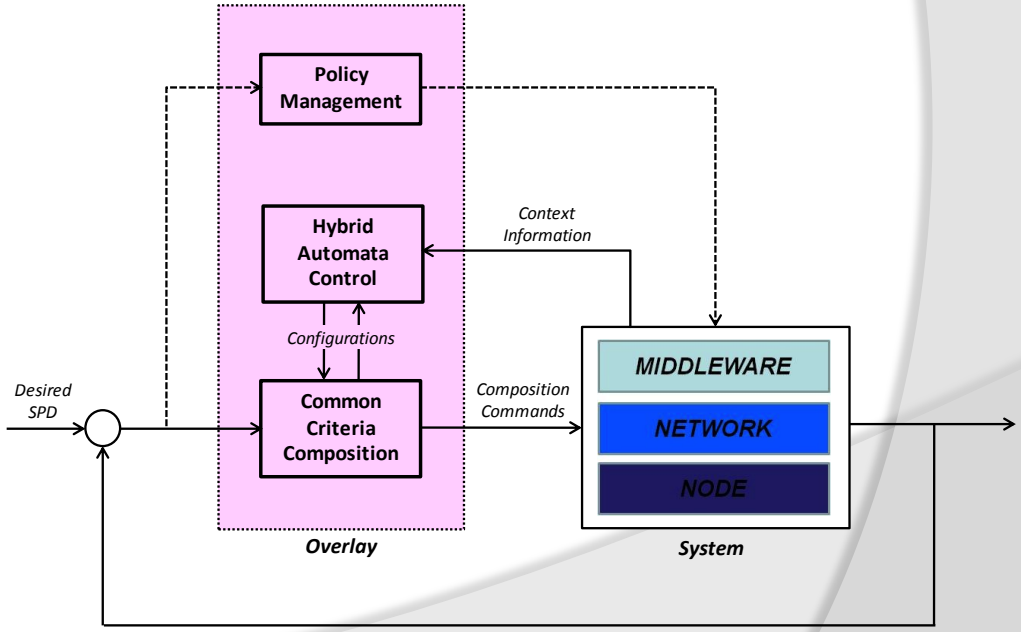
- The security agent derived for the pilot purposes worked adequately in the demonstration scenario, but is far away from being a consolidated entity to be deployed.

- The control algorithm module is too weak to manage the ambitious composability expected by the SHIELD framework, so a more complex (and articulated) architecture should be derived.

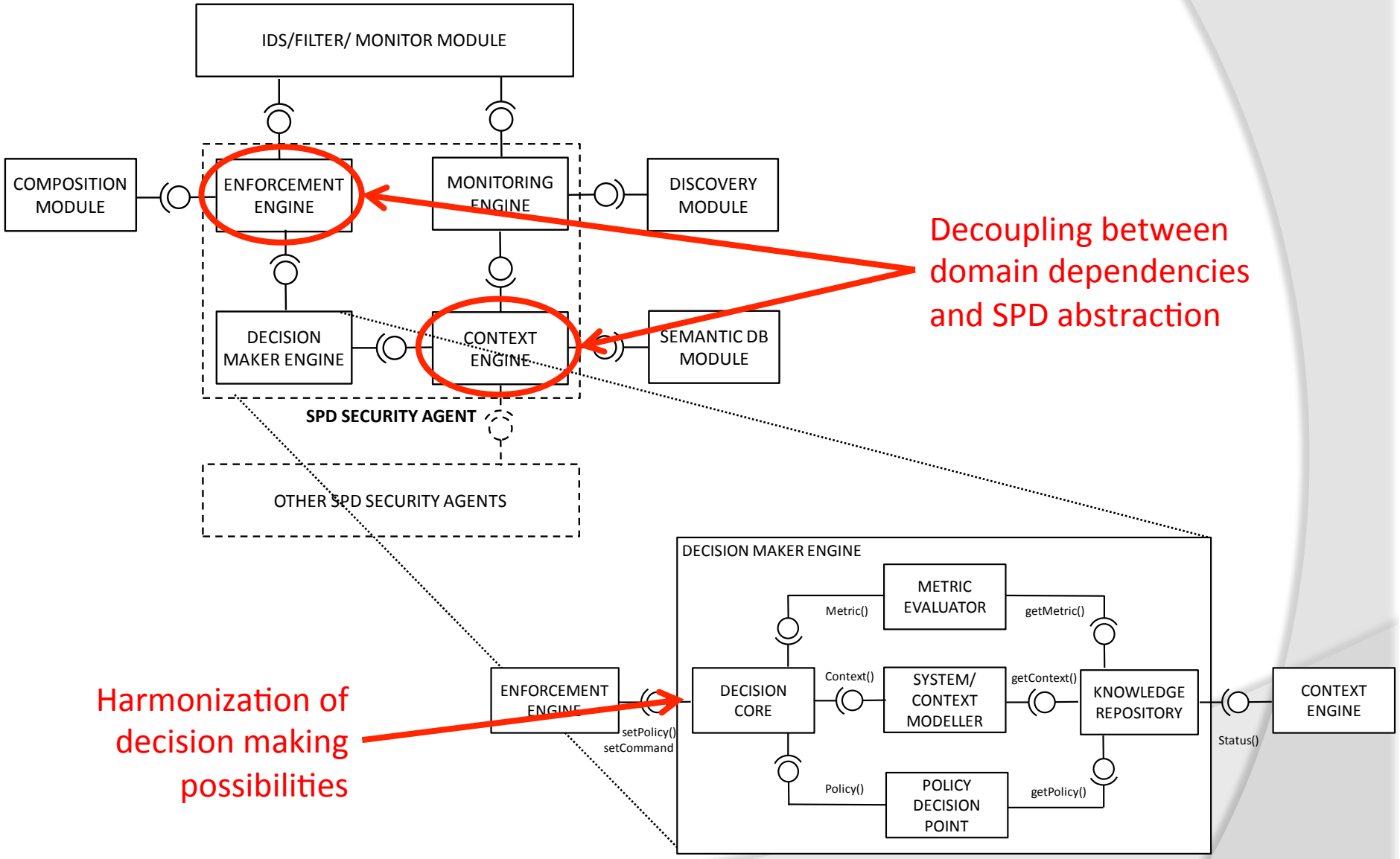
- Only static composability is implemented: it is necessary to enable dynamic composability

Good heritage

- Decoupling between decision making and knowledge assure flexibility, scalability and increases the lifetime of the proposed solution.



Security Agent: nSHIELD achievements 1/4



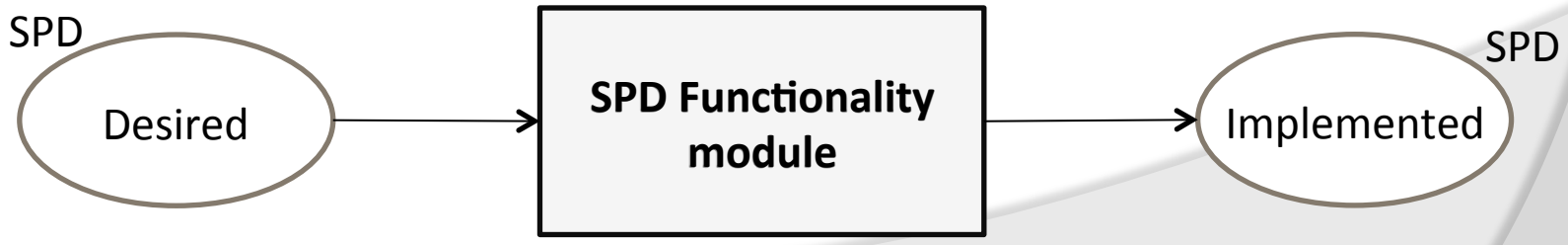
Security Agent: nSHIELD achievements 2/4

Colored Petri Nets

- Colored Petri Nets (CPNs) is an extension to Petri Nets developed to model complex systems.
- CPNs combine the PNs structure with the high-level programming capabilities
- The hierarchical structure is the strength of CPNs, the sub-modules composition allows to model complex system and to work at different abstraction level

System Model

- To model an heterogeneous and complex system is necessary a scalable structure to overcome state explosion
- The system basic element are *SPD functionality module*
  > Input state contains an SPD token that represents the desired value i.e. desired metrics (number) and/or specific implementation (string)
  > Output state represent the current state (if/how the SPD functionality is implemented).



nSHIELD

Security Agent: nSHIELD achievements 3/4

Means to mitigate threaths or SPD level Translated into tokens

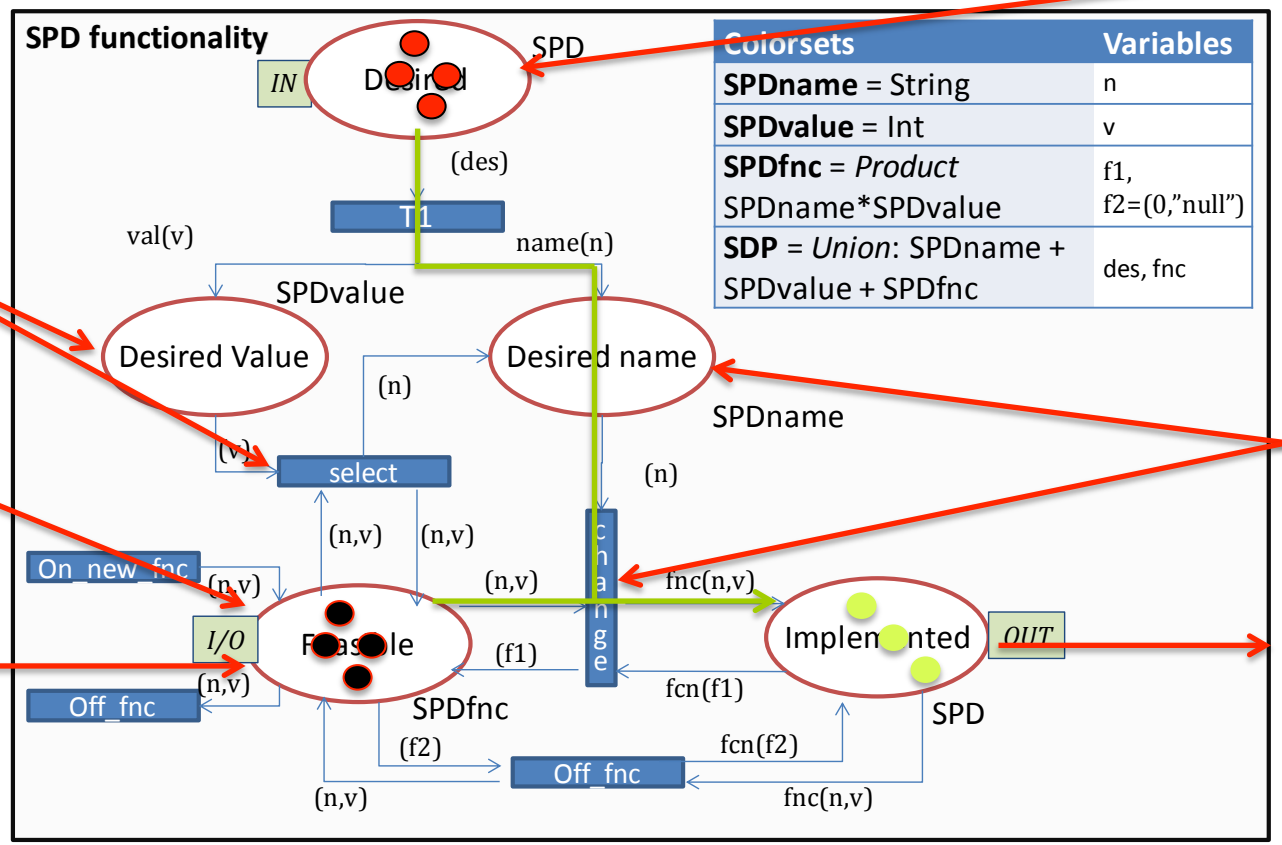
Metric Evaluation

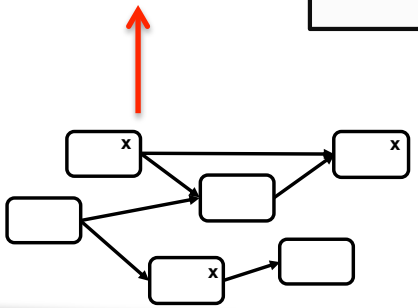
Context Information

Available SPD functionalities Translated into tokens

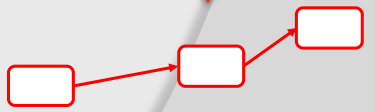
Policy Enforcement

Back to enforcement engine





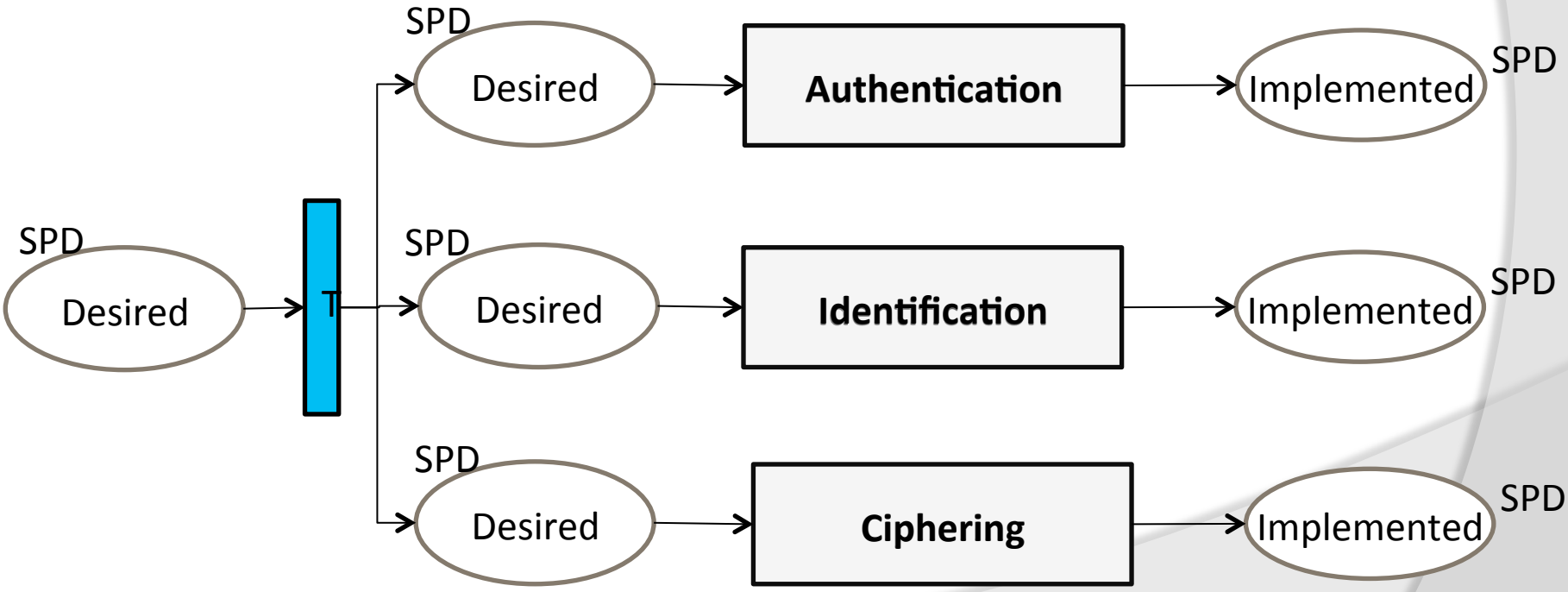
FROM «PROBLEM» TO «SOLUTION»



nSHIELD

Security Agent: nSHIELD achievements 4/4

- The approach can be scaled: one CPN per functionality or one CPN for the whole system
- Complexity is at most linear
- Hierarchical structure
- Easy implementaion of a Petri Net into software code
- Model checking



Overlay composability: conclusions

We have learned that the overlay composability is

SPD Driven –> focus on SPD components

Modular and **scalable** –> abstraction of the SPD functionality

Close to **existing standards** –> CC logical chain

Measurable -> possibility to quantify SPD level

Flexible – > decision making at different levels

Deterministic –> formal modeling for composability (model cheking)

In one word

Feasible –> we can transform the proof of concept into a product, but…

…enough way ahead still to go!

nSHIELD

Thanks for your attention

Any questions?

Andrea Fiaschetti