YaJvaConf

BIECO project – Methodology for Security Evaluation

Marcin Byra
Software Developer, 7bulls.com

Academic Partners FUNDACJA

BIECO: Building Trust in Ecosystems and Ecosystems Components

UNINOVA

UNIVERSIDAD DE MURCIA

Fraunhofer IESE

TTTechAuto

TTTech

gradiant

TECHNICAL UNIVERSITY OF CLUJ NAPOCA ROMANIA

I-FEVS

7bulls.com

HOLISUN

RESILTECH

BIECO: Building Trust in Ecosystems and Ecosystems Components

- The problem
  - ICT components developed by third parties
  - Third party components: vulnerabilities? built with the best security practices?
  - vulnerabilities can propagate, high risk of cyber-attacks
  - vulnerabilities might remain undetected for years
  - …even if they are well-known

BIECO: Building Trust in Ecosystems and Ecosystems Components

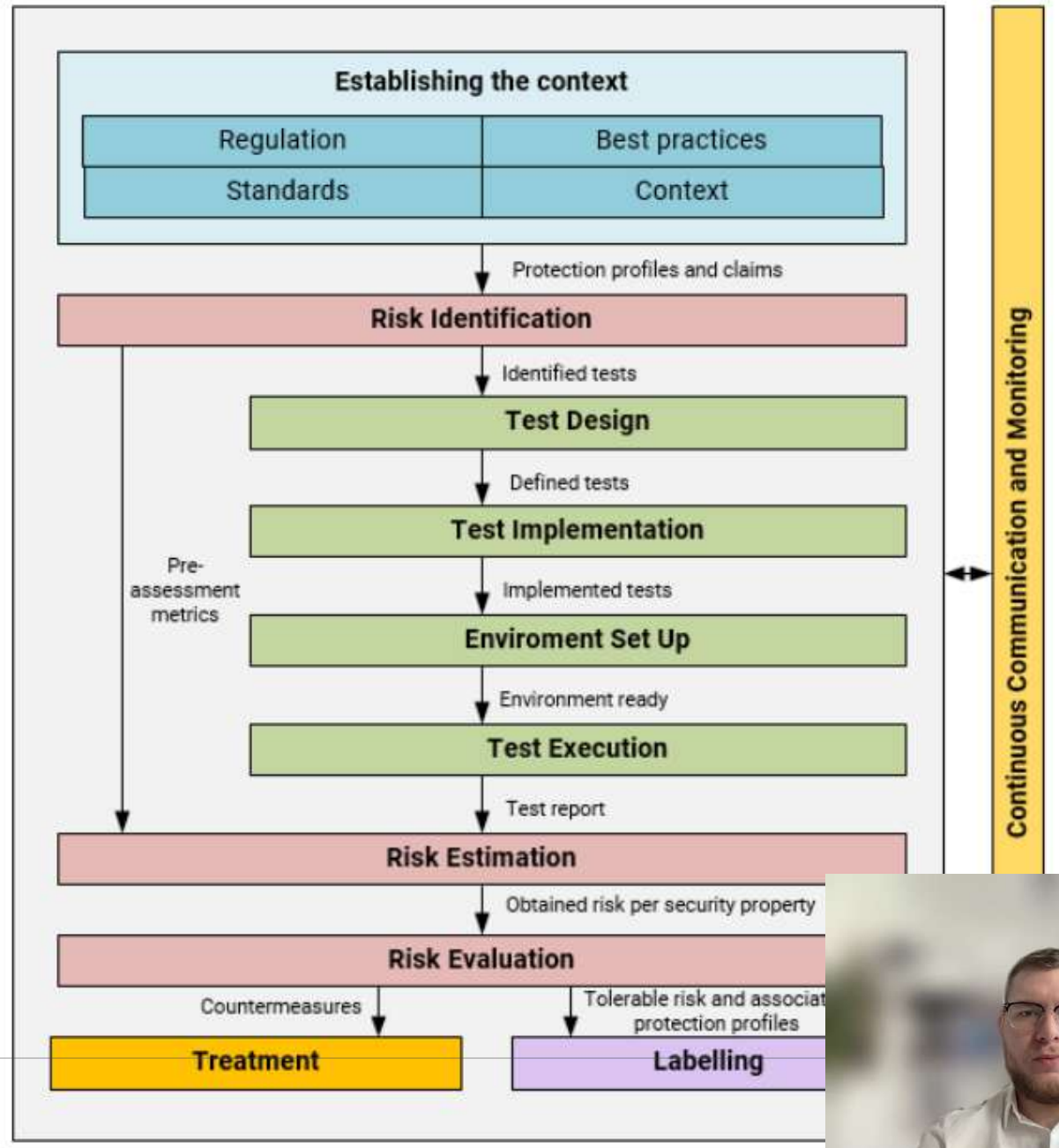
- The idea
  - Create a framework that provides mechanisms to manage the security risks
  - Establish a common level of trust and security
  - Here comes the BIECO

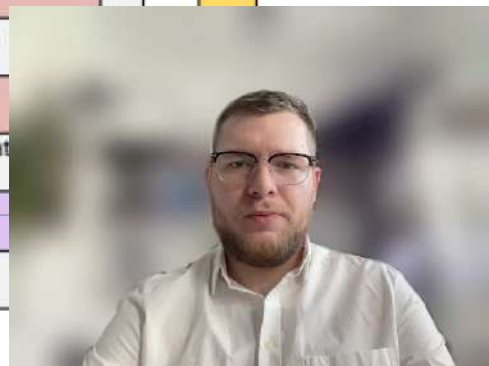
BIECO: Building Trust in Ecosystems and Ecosystems Components

- **Main goal: deliver a framework for improving trust and security within ICT ecosystems**
- There is a lot of background research behind the BIECO framework:
  - how we build on the existing standards, knowledge, vulnerability databases
  - how we translate general knowledge about software components to unified numerical results
  - Development of all the BIECO tools
  - …
- If you are interested, check out www.bieco.org to learn all about it [1]

Methodology scheme (theory)

1. **Establishing the context**: understanding the case
2. **Risk identification**: finding our vulnerabilities
3. **Risk estimation:** gathering information from components
4. **Risk evaluation:** understanding the impact of the risks
5. **Treatment and labelling**: visualization and mitigation





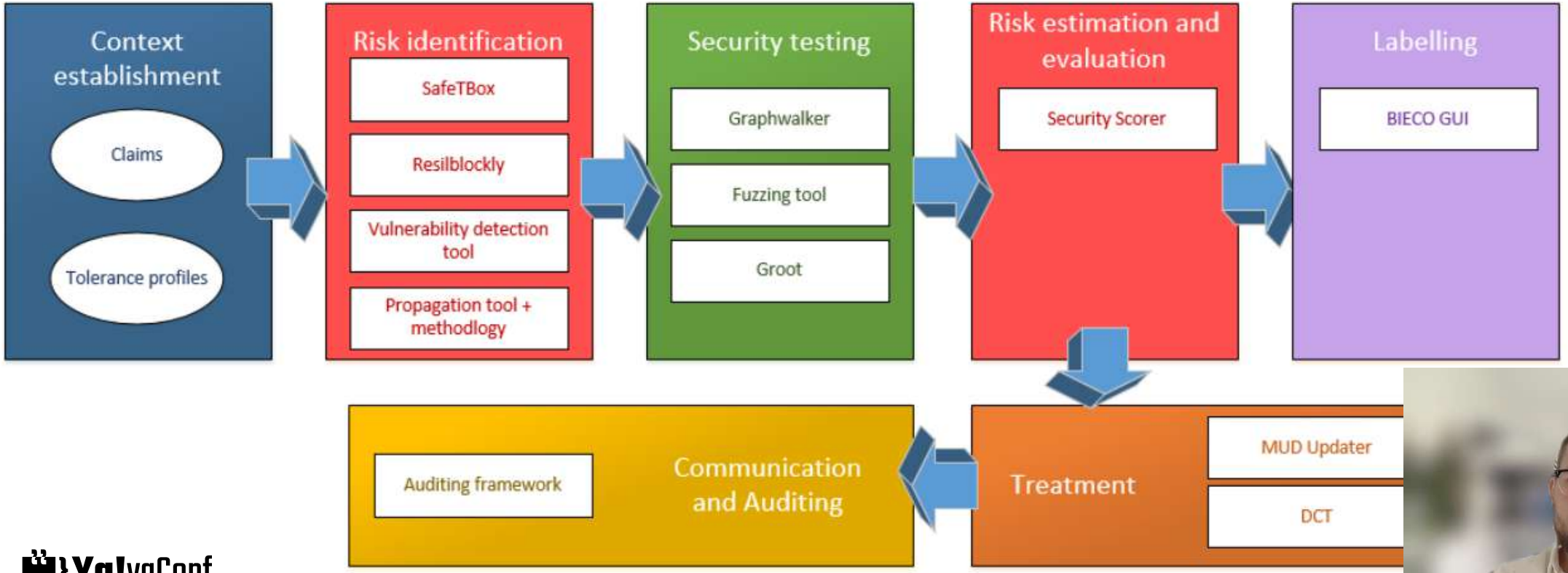
BIECO: Building Trust in Ecosystems and Ecosystems Components

From now on, we will go through *practical* side of the project

- **how to use the BIECO tools and obtain security evaluation of your software component**

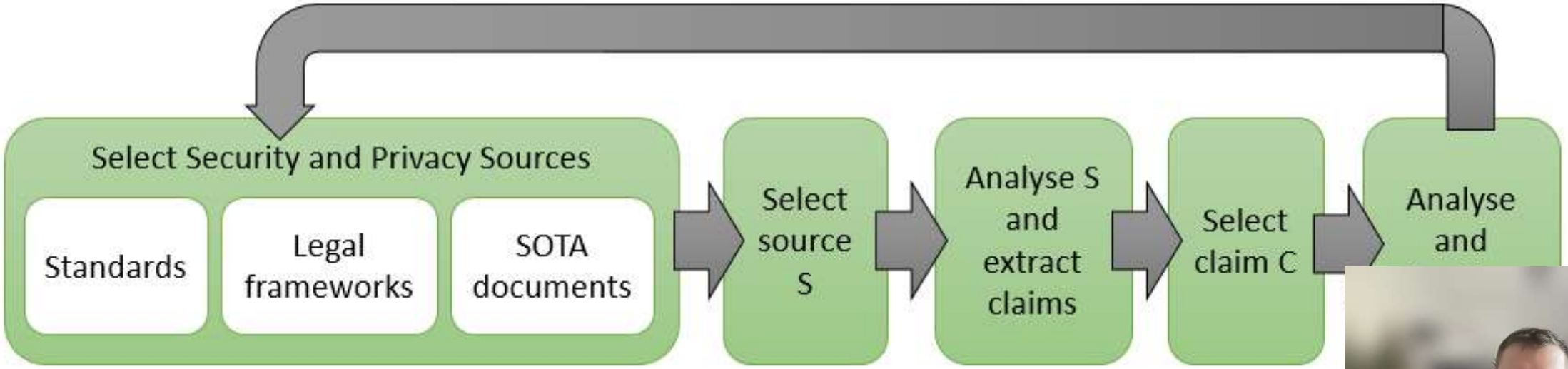
Methodology scheme (practice)

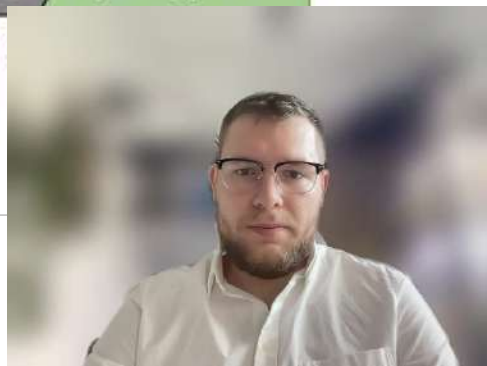
- That was the theory: establishing the background for **creating** the methodology
- In practice: we want to **use** the methodology, we use these tools
- think **that was a model and this is the implementation**



1. Establishing the context: Claims

- Starting point:
  - *What* we should evaluate?
  - We create a set of **claims** against which the Target of Evaluation (TOE: a software component, etc.) will be assessed





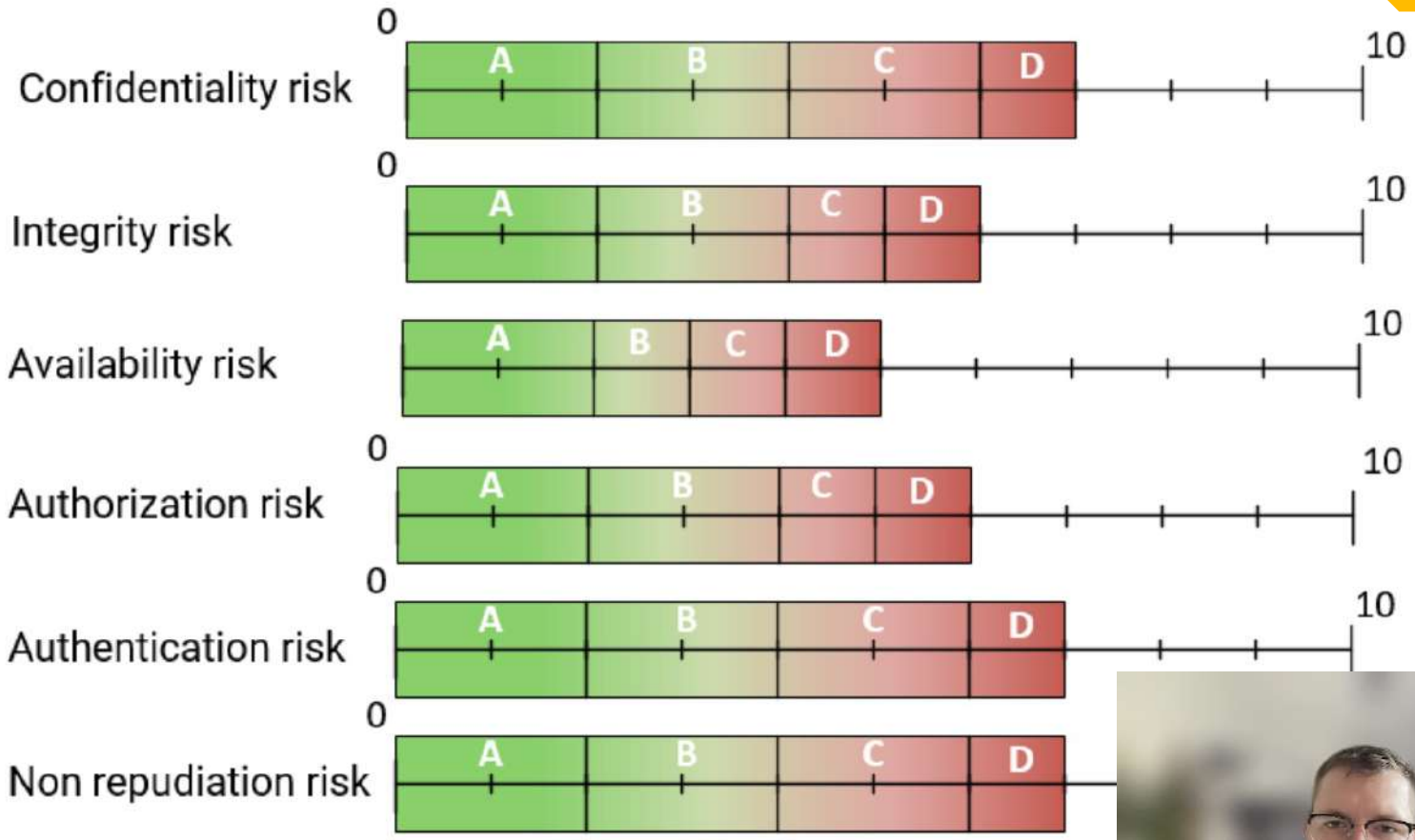
1. Establishing the context: STRIDE categories

- Each claim belongs to one of the STRIDE categories [3]

Threat	Desired Security Property
Spoofing	Authenticity
Tampering	Integrity
Repudiation	Non-repudiability
Information Disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorization

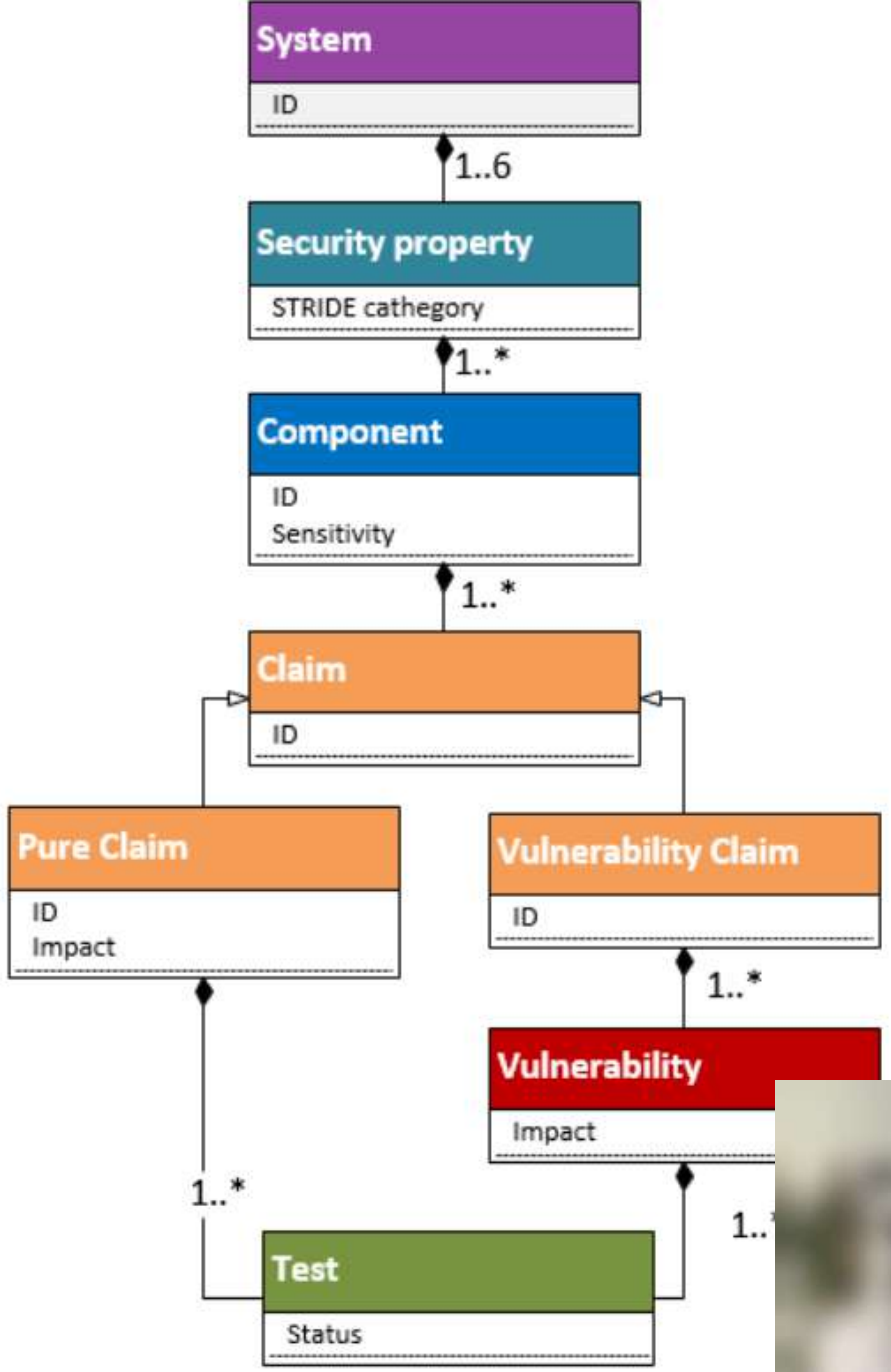
1. Establishing the context: Tolerance Profiles

- Tolerance profiles:
  - To simplify the results and define where the highest risk is in our use case



2. Risk identification

- **Decomposition of the system**
  - a way of describing the system and the relation between components claims, and tests…
  - in a simple YAML file



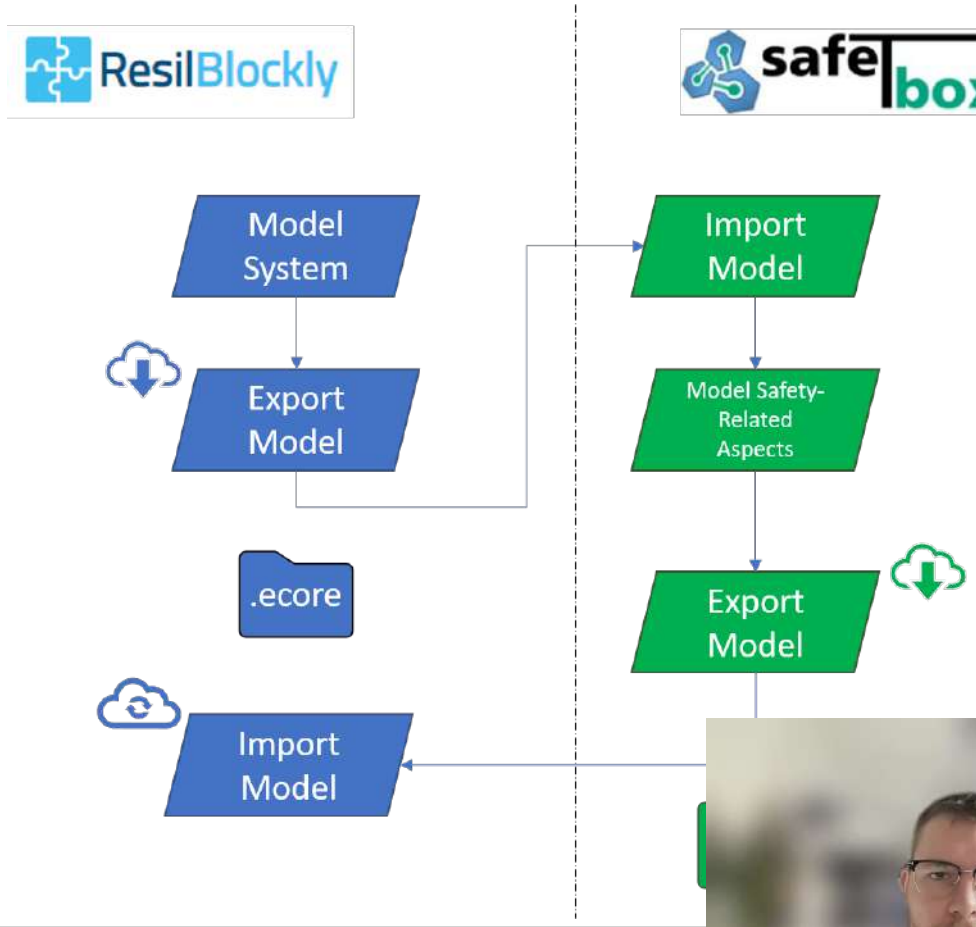
2. Risk identification

Let’s visualise the output of the risk identification phase

System: Vehicle				
LEVEL 1	**LEVEL 2**	**LEVEL 3**	**LEVEL 4**	**LEVEL5**
Security property (STRIDE)	**Component (Sensitivity)**	**Claims (D7.1)**	**Vulnerabilities and Impact (for applicable claims[1])**	**Tests**
Confidentiality	Multimedia ECU (7)	C0		TEST_00, TEST_07
		C1		TEST_01
	Controller (9)	C11		TEST_02, TEST_03
		C32	CVE-2021-36988 / 7.5	TEST_04, TEST_05, TEST_06
Integrity	Controller (9)	...		
	Database (8)	...		
	OBD (6)	...		
	...			

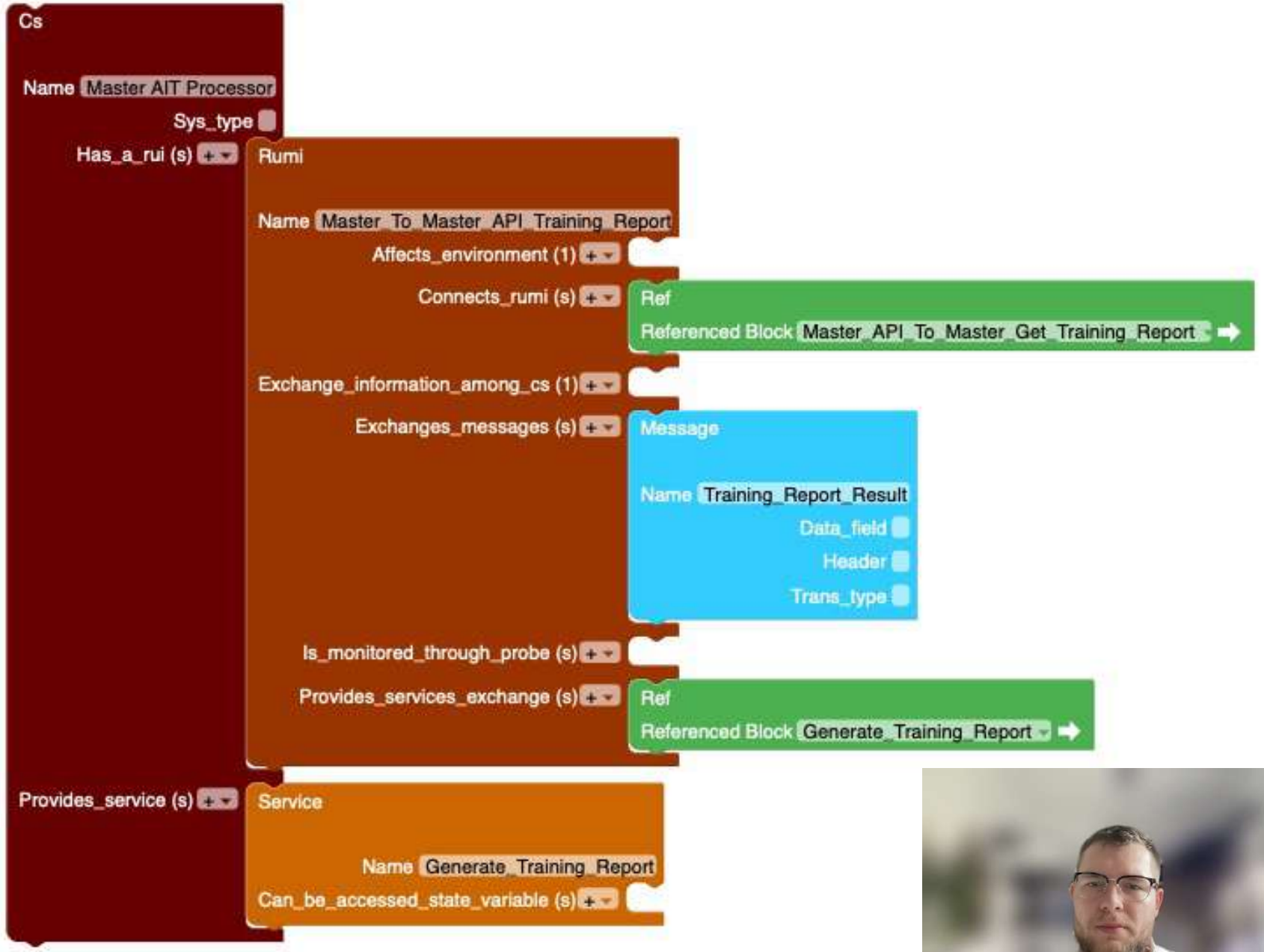
2. Risk Identification – SafeTBox & ResilBlockly

- **SafeTbox** and **ResilBlockly** are tools that have been included into the BIECO risk estimation phase
- They are two risk modelling & analysis tools
- They were extended to cooperate with each other in creating a…
- **Threat MUD file** – BIECO uses this file to represent the current state of analysis



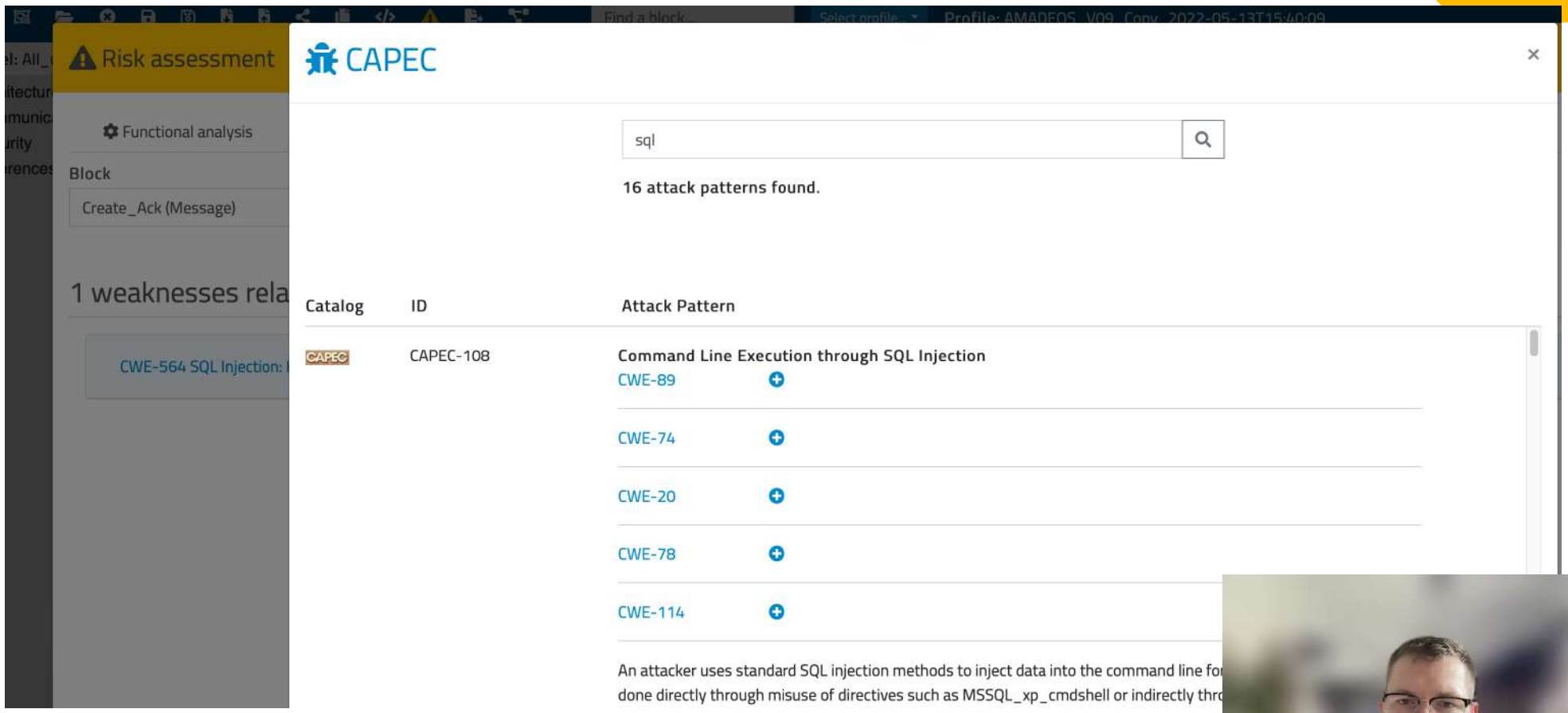
2. Risk Identification – SafeTBox & ResilBlockly

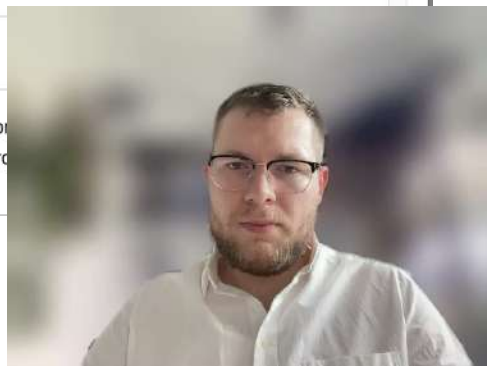
ResilBlockly model example



2. Risk Identification – SafeTBox & ResilBlockly

ResilBlockly adding a weakness to a model's element

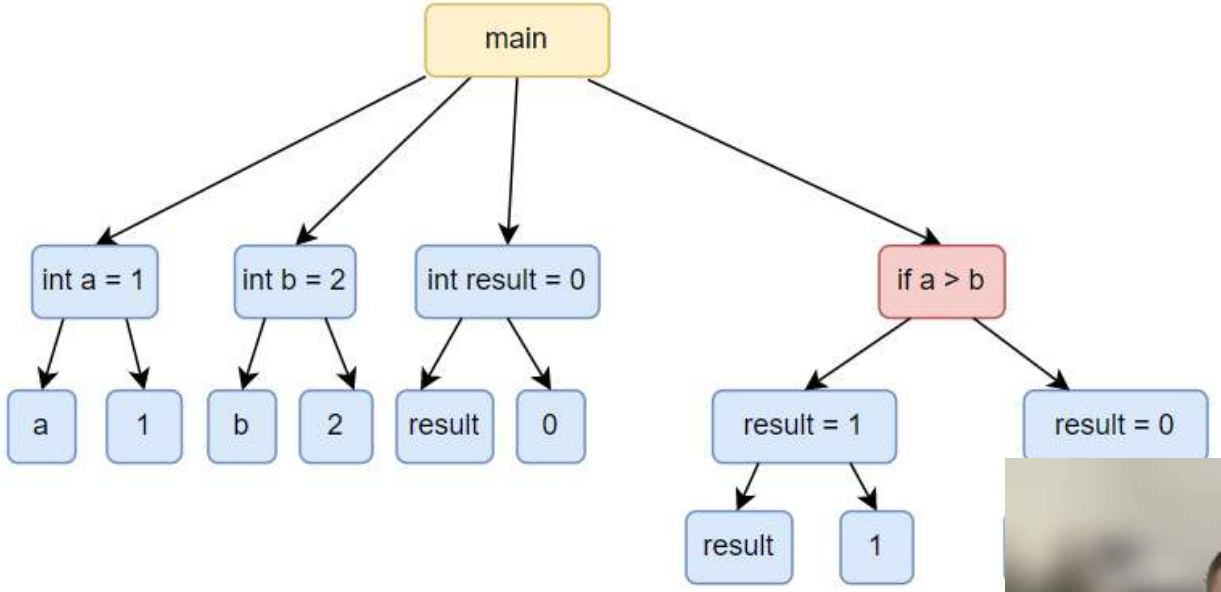


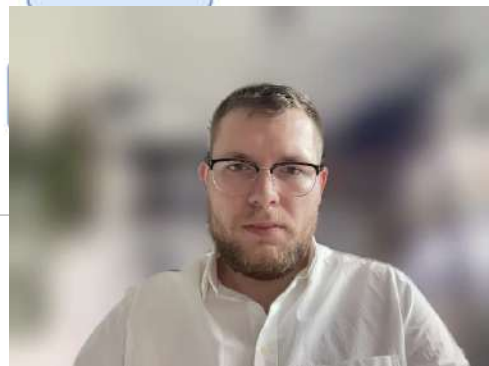


2. Risk Identification – Risk Propagation Tool

- The main goal of the **propagation tool** is to find and indicate the components or elements a single vulnerability can affect, and therefore, its path within the system.
- For C, C++, Java, Python

```
main:
        int a = 1
        int b = 2
        int result = 0
        if a>b:
                result = 1
        else:
                result = 0
```





2. Risk Identification – Security Testing

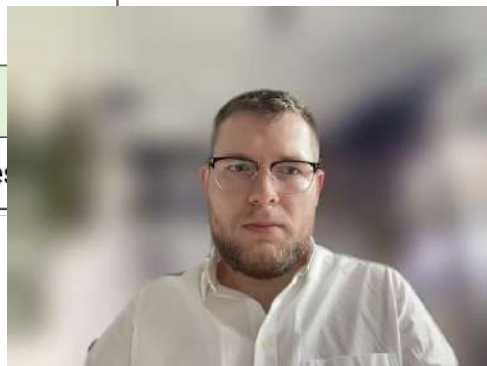
- The following tools are used to run real tests and produce files describing their results

- On the lowest level, each test passes, fails, or results in a specific value according to some metric:

Test result	Likelihood
PASS	0
FAIL	1
Specific metric	Metric weighted between 0 and 1

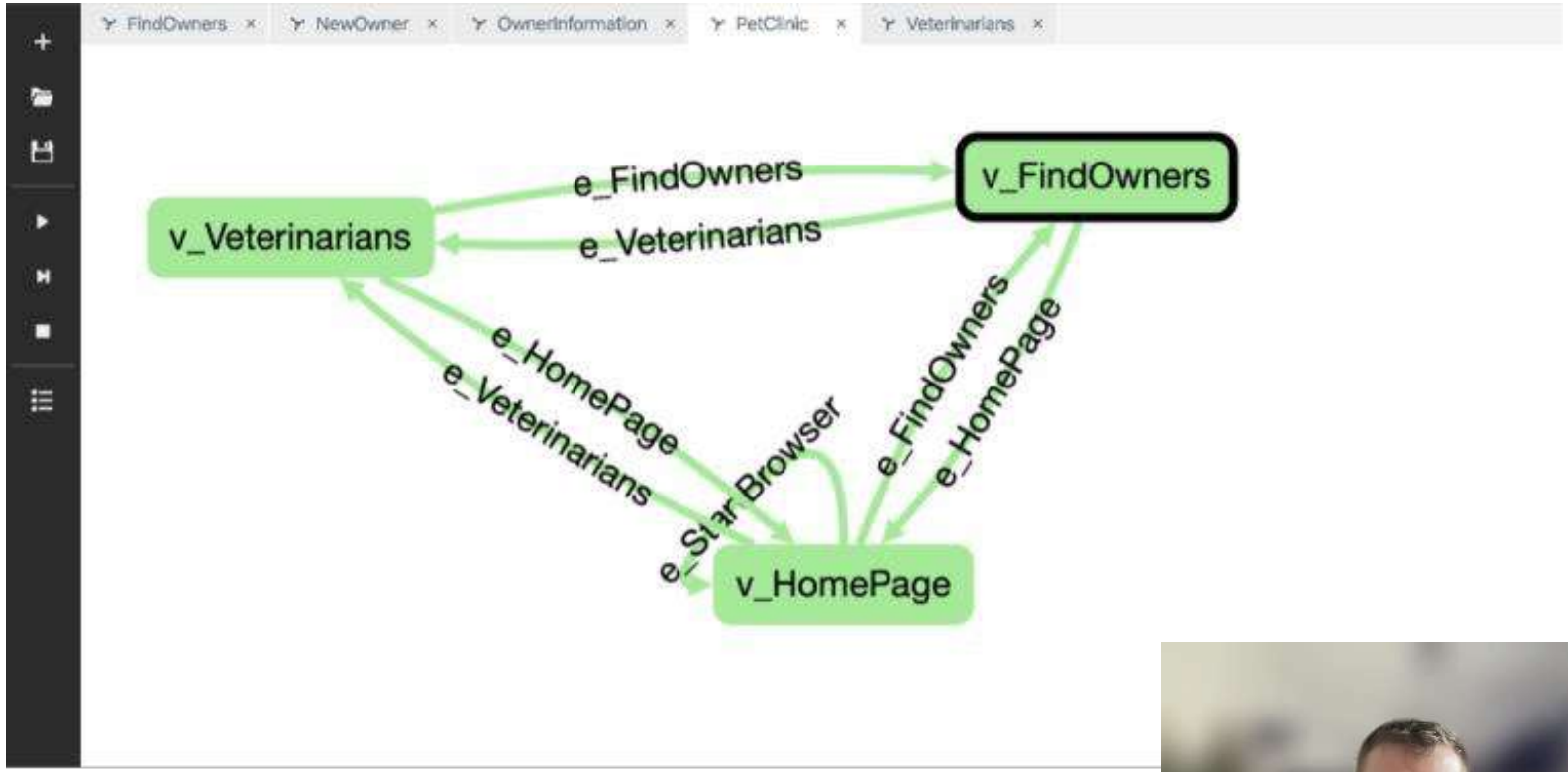
- Metric example:

Safety metric	Criteria
0	No injury.
10	Light and moderate injuries.
100	Severe and life-threatening injuries (survival probable).
1000	Life-threatening injuries (survival uncertainty), fatal injuries



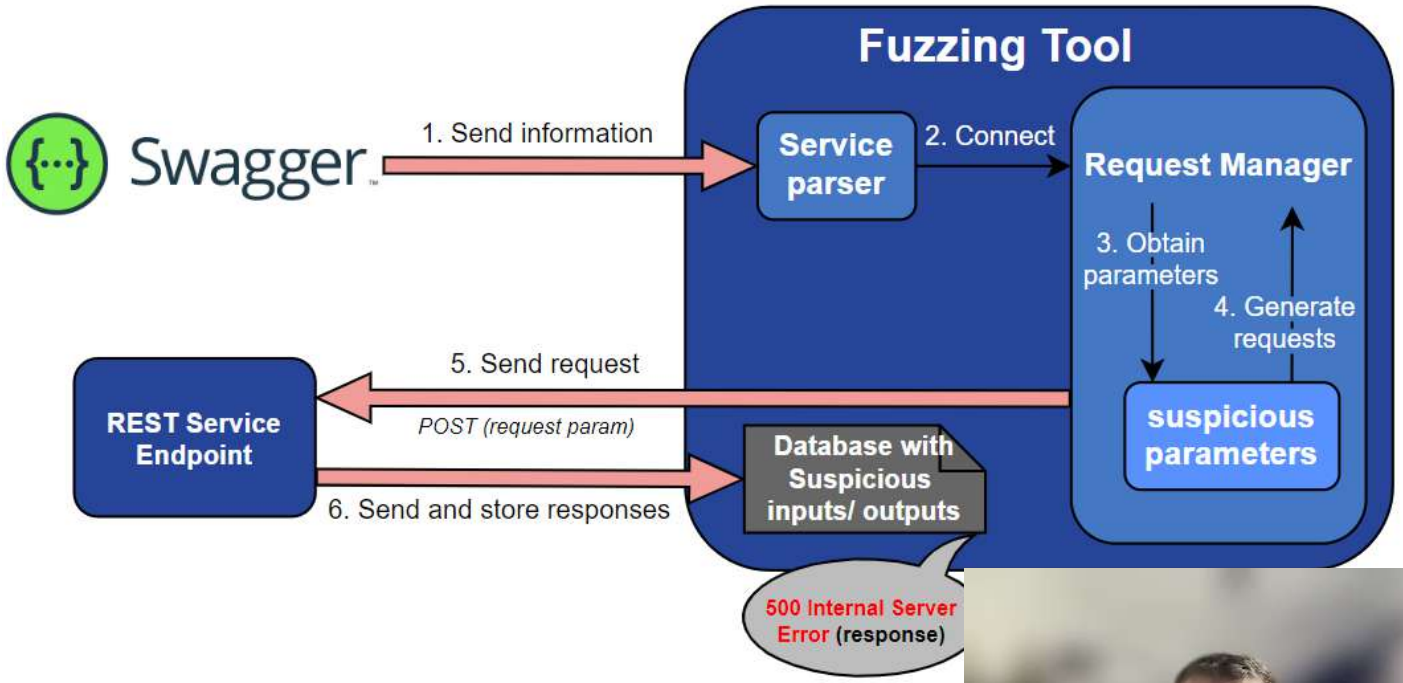
2. Risk Identification – Security Testing - Graphwalker

- **GraphWalker** [4] – an open-source tool, extended for the methodology by one of the BIECO partners, UMU
- **Model-based testing** tool: modelling the software as a graph of
  - Actions/transitions – edges
  - Verifications/assertions – vertices



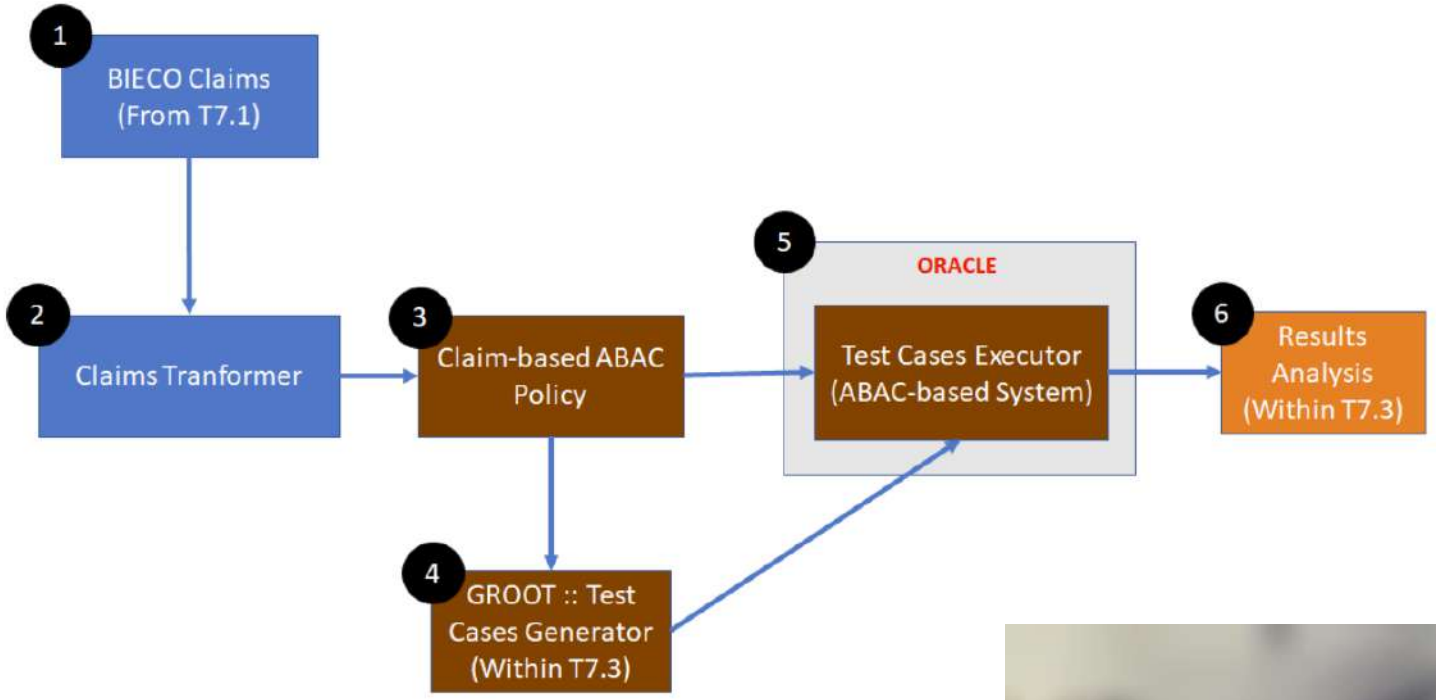
2. Risk Identification – Security Testing – Fuzzing tool

- Fuzzing Tool:
  - Fuzzing – process of finding security vulnerabilities in a service's interface by repeatedly sending requests with modified inputs
  - **dynamic analysis** against a real, working service



2. Risk Identification – Security Testing – GROOT

- GROOT [5]
  - **G**dp**R**-based c**O**mbinat**O**rial **T**esting
  - is a general **combinatorial testing approach**, for validating systems managing GDPR's concepts (e.g., Data Subject, Personal Data or Controller)
  - **in general, the idea to perform automate analysis based on claims**

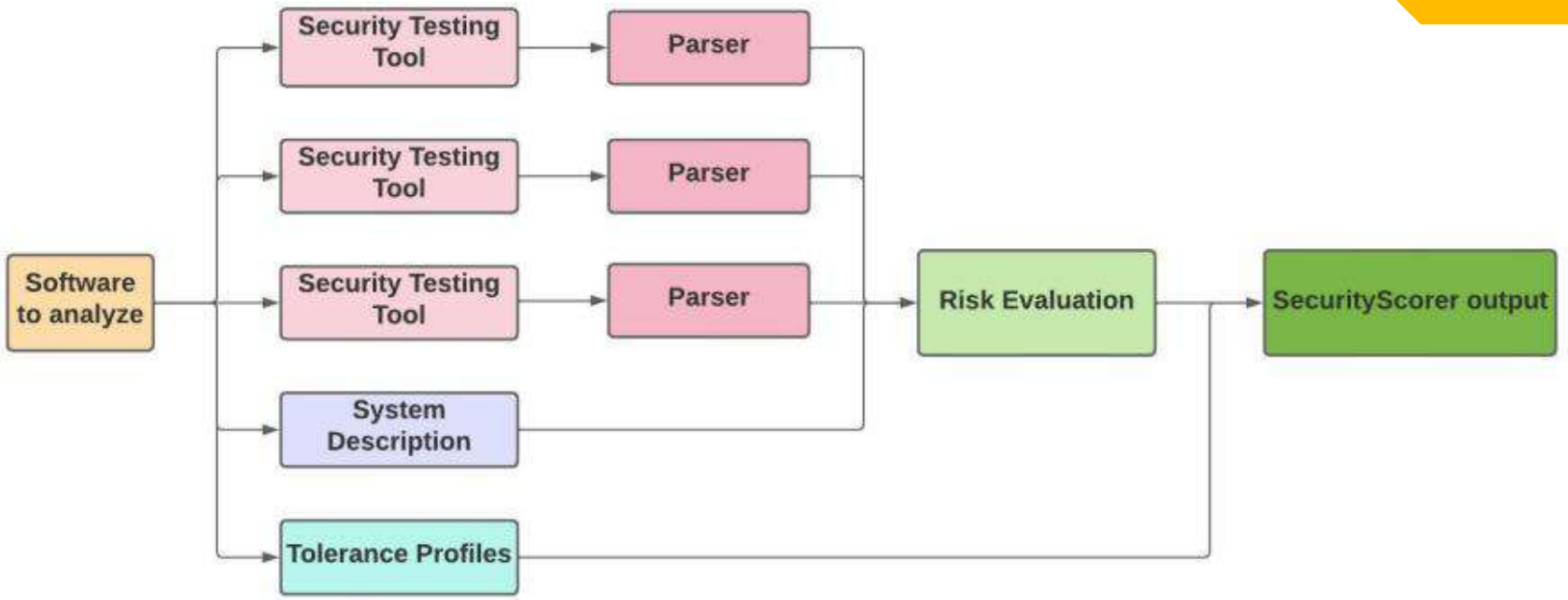


3. Risk Estimation and 4. Risk Evaluation

- At this step, we have:
  - System description
  - Tolerance profiles
  - Security testing tools' results in various formats
- We have to combine all this information:
  - Parse
  - Unify
  - Create the internal representation of the system and relate the test (and results)
    - to specific claims
    - in specific components
    - in specific STRIDE categories
  - Evaluate the results and calculate numerical values of the risk
- This combines the ideas of the **Risk Estimation** and **Risk evaluation** phases

3. Risk Estimation and 4. Risk Evaluation – Security Scorer

- All these actions are performed by **SecurityScorer**
- developed by 7bulls.com for the BIECO methodology




```
{
  "scores": {
    "confidentiality": 1.0,
    "integrity": 2.0,
    "availability": 0.0,
    "authorization": 4.2,
    "authentication": 0.6,
    "non_repudiation": 1.0
  }
}
```


5. Labelling and treatment – BIECO GUI

- BIECO GUI
  - Central interface and integration platform
  - Connecting all the tools in of the BIECO
  - adding new testing tools
  - Starting and tracking the testing progress
  - History
  - Example of the results – spider chart

- Treatment:
  - Having all the test results and reports from all tools, labelled or numerical results of STRIDE categories, you know how and where to improve your software security

Summary

- BIECO is in progress
  - This summer, we successfully instantiated the methodology, connecting all the tools presented today, integrated in the BIECO GUI
  - We evaluated the first Use Case using complete methodology
  - The background is prepared and formalized
  - By the end of 2023, BIECO implementation should reach the final, easy-to-use in real-world scenario state

References

[1] www.bieco.org

[2] ECSO, “European Cyber Security Certification A Meta-Scheme Approach v1.0.” 2017.

[3] Microsoft, “STRIDE chart | Microsoft Security Blog,” 2007. https://www.microsoft.com/security/blog/2007/09/11/stride-chart/ (accessed Jul. 01, 2021).

[4] https://graphwalker.github.io/

[5] Daoudagh, S., Marchetti, E. (2022). GROOT: A GDPR-Based Combinatorial Testing Approach. In: Clark, D., Menendez, H., Cavalli, A.R. (eds) Testing Software and Systems. ICTSS 2021. Lecture Notes in Computer Science, vol 13045. Springer, Cham. https://doi.org/10.1007/978-3-031-04673-5_17

YaJvaConf

Thank you for watching

Remember to rate the presentation and leave your questions in the section below.