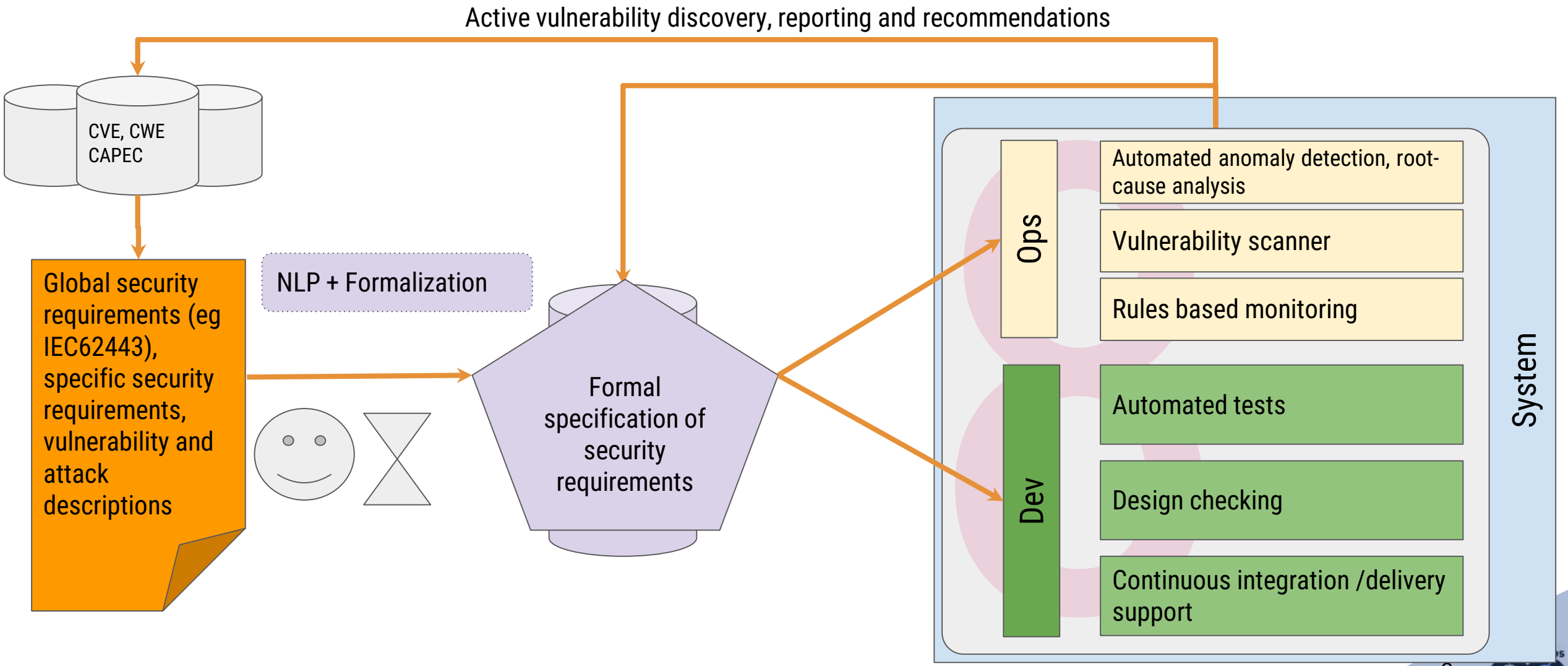
Security Requirements Formalization with RQCODE.

presented by
by andrey.sadovykh@softeam.fr

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 957212

VeriDevOps - automating security requirements verification in DevOps



Requirements as Code

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 957212

Related work

Bruel, J.M., Ebersold, S., Galinier, F., Mazzara, M., Naumchev, A., Meyer, B.: The Role of Formalism in System Requirements. ACM Comput. Surv. 2021

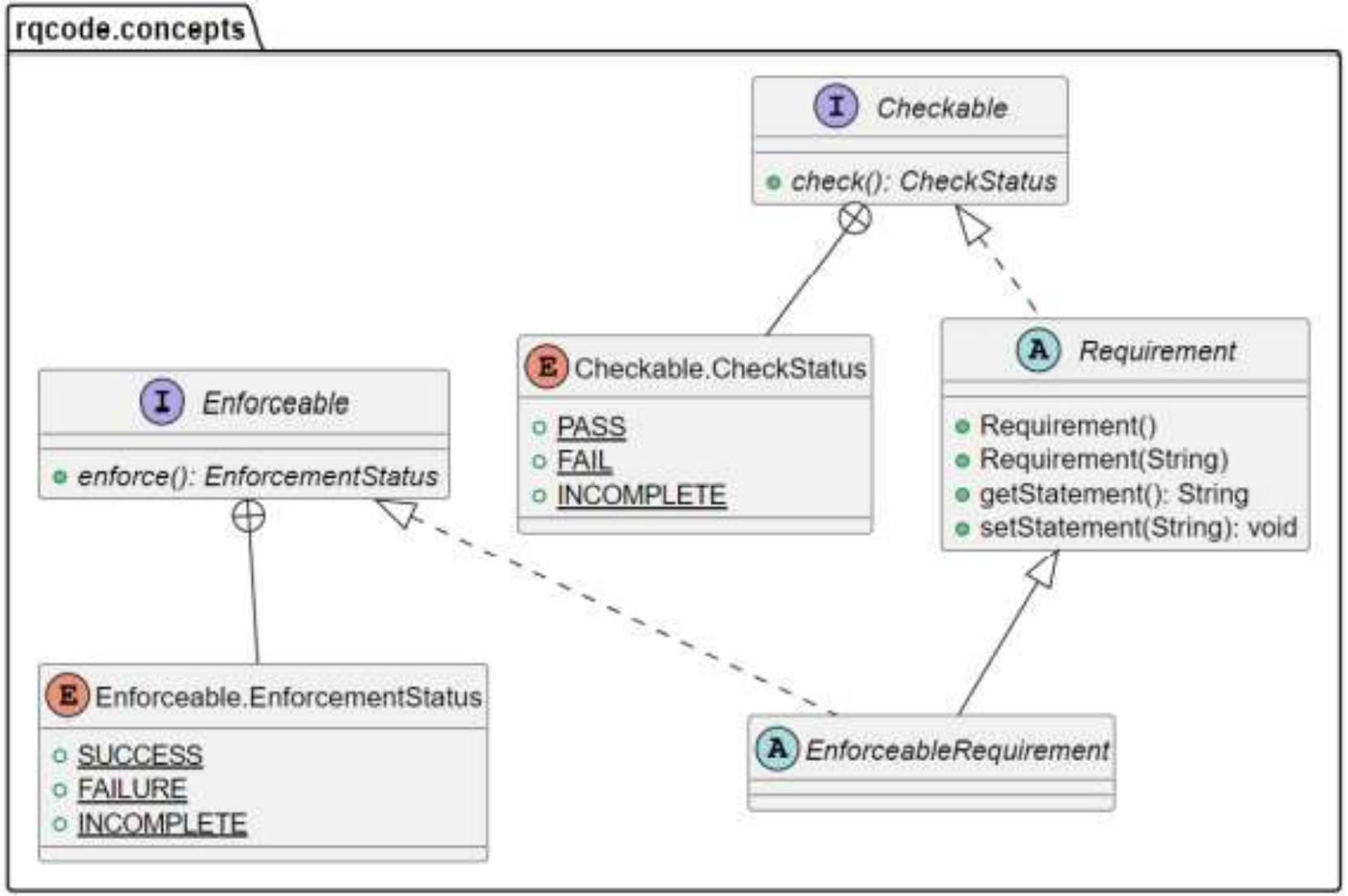
1. **Natural language**
    a. most used
    b. validation of the technical specification is conducted by oral discussions and interpretation with stakeholders[51].
2. **Semi-Formal**
    a. Modelling approach e.g. SysML[11]
    b. The analysis still remains mostly manual by constructing and considering various viewpoints.
3. **Automata/graphs**
    a. based on automata or graph theory. They
    b. deals with the concepts of automata, formal languages, grammar, algorithms, computability, decidability, and complexity [1].
4. **Mathematical methods**
    a. based on fundamental mathematical and algebra formalisms
    b. eg Event-B [5], Alloy [28], Form-L [10], VDM [7], Tabular Relations [50].
5. **Seamless methods** are
    a. programming-language-based methods [44]
    b. constraint logic and programming by contract

Requirements as Code (Java impl)

Originating from: Naumchev, A. and Meyer, B., 2017. Seamless requirements. Computer Languages, Systems & Structures, 49, pp.119-132.

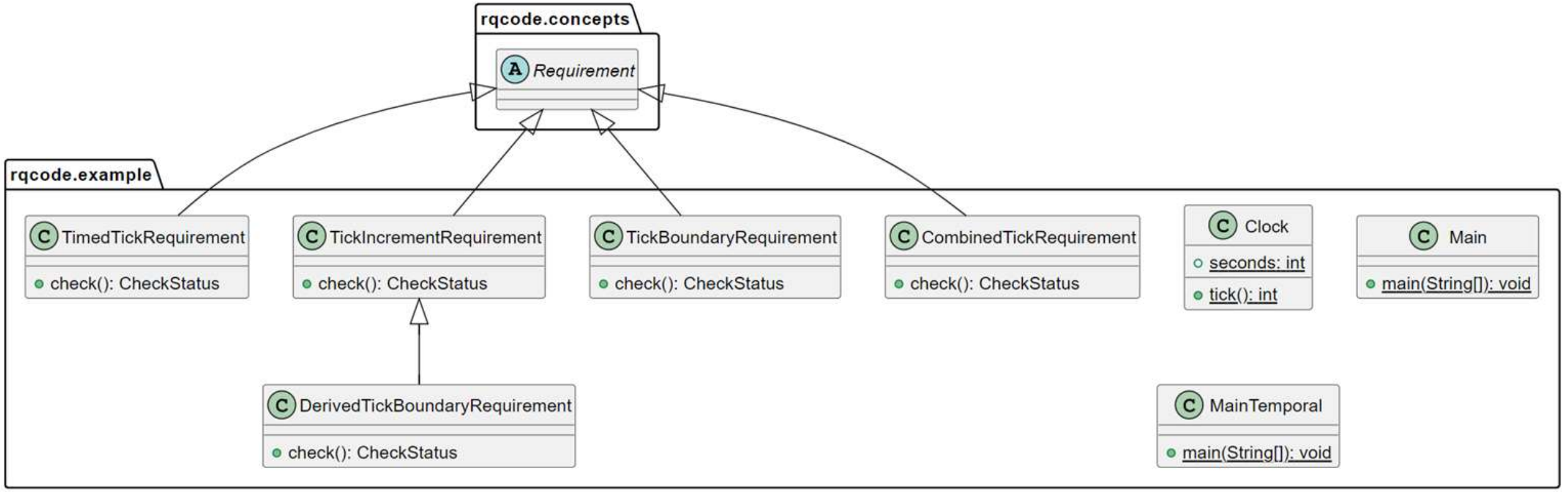
1. Defining *Text* and *Test* in a Class
2. Hypothesis for benefits
   - Enhanced *verifiability* => clearness
   - Lightweight formalization
   - Easy combination, integration
   - Reuse, maintenance, traceability, OOP analysis.

RQCODE Concepts



Example

(REQ1) A clock tick increments current second if it is smaller than 59.”

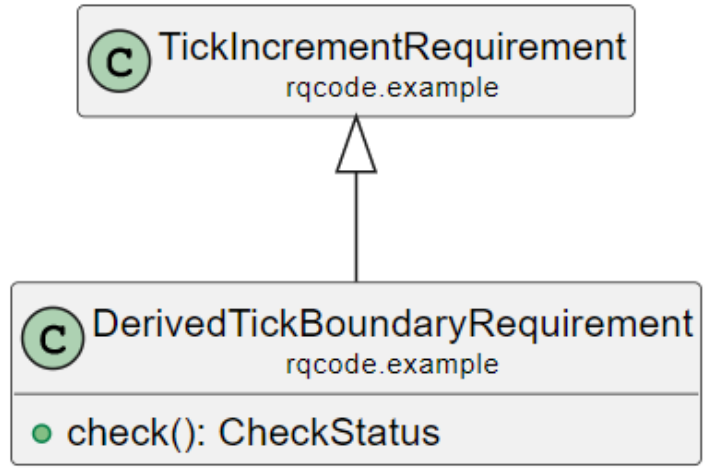


(REQ1) A clock tick increments current second if it is smaller than 59.”

```
1  public class TickIncrementRequirement extends Requirement {
2      public TickIncrementRequirement() {
3          super("A clock tick increments current second if it is smaller
               ↪than 59");
4      }
5      @Override
6      public CheckStatus check() {
7          if (Clock.seconds < 59)
8              return ((Clock.seconds + 1) == Clock.tick()) ?
9              CheckStatus.PASS : CheckStatus.FAIL;
10         return CheckStatus.INCOMPLETE;
11     }
12 }
```

Deriving (REQ2) Clock seconds value must be between 0 and 59

```
1  public class DerivedTickBoundaryRequirement extends
       ↪TickIncrementRequirement {
2      public DerivedTickBoundaryRequirement() {
3          super.setStatement(super.getStatement() +"\nIn addition, Clock
               ↪seconds value must be between 0 and 59.");
4      }
5      @Override
6      public CheckStatus check() {
7          if (Clock.seconds > 59) return CheckStatus.FAIL;
8          if (Clock.seconds < 0) return CheckStatus.FAIL;
9          return super.check();
10     }
11 }
```



Combined (REQ2) Clock seconds value must be between 0 and 59

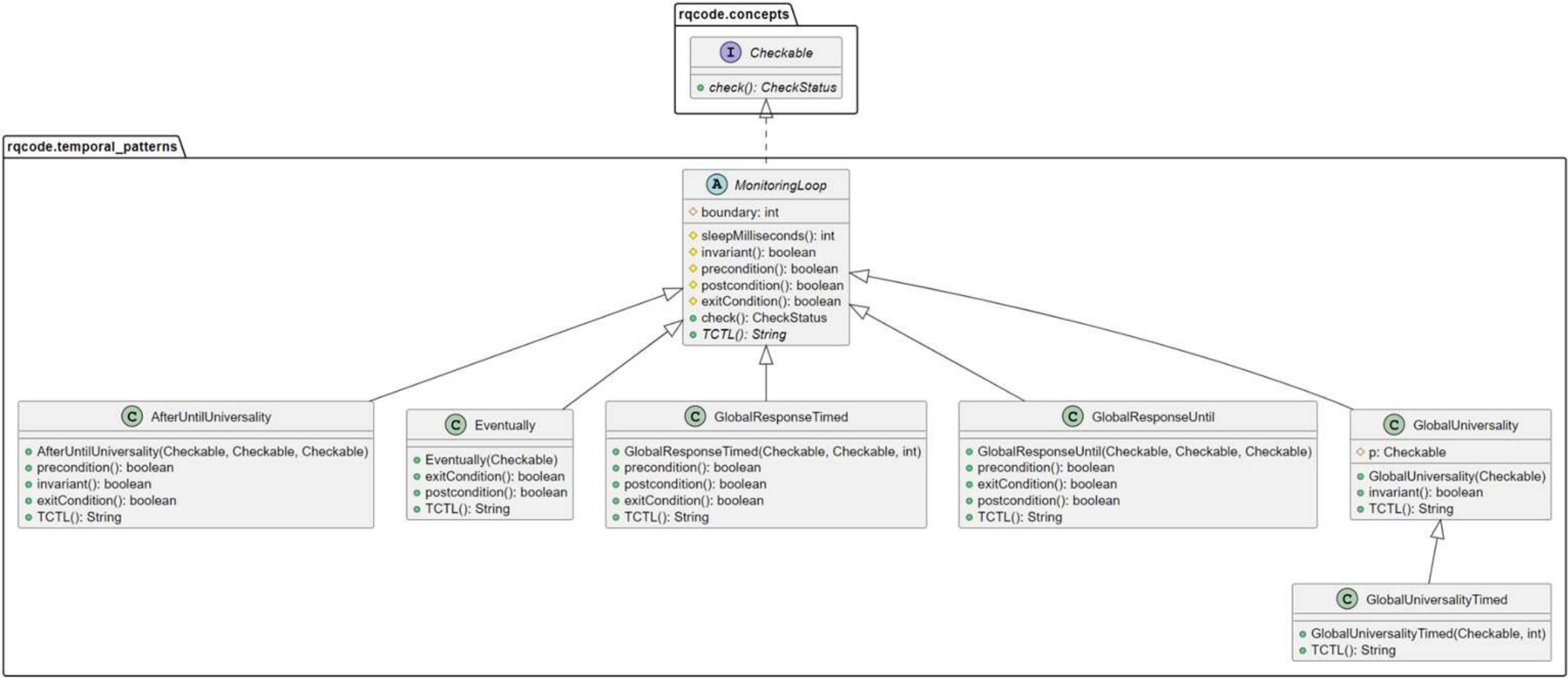
```
1  public class CombinedTickRequirement extends Requirement {
2      TickIncrementRequirement tr;
3      TickBoundaryRequirement br;
4      public CombinedTickRequirement()  {
5          super("The Clock must satisfy the tick increment (REQ1) and
               ↪seconds boundary (REQ2) requirements.");
6          tr = new TickIncrementRequirement();
7          br = new TickBoundaryRequirement();
8      }
9      @Override
10     public CheckStatus check() {
11         if (br.check()== CheckStatus.FAIL) return CheckStatus.FAIL;
12         return tr.check();
13     }
14 }
```

Temporal Patterns

Dwyer, M. B., Avrunin, G. S., & Corbett, J. C. (1999, May). Patterns in property specifications for finite-state verification. In Proceedings of the 21st international conference on Software engineering (pp. 411-420).

1. *P* always eventually holds
2. Globally, Universally: Globally, it is always the case that *P* holds.
3. After *Q* Until *R* Universally *P*: After *Q*, it is always the case that *P* holds until *R* holds.
4. Globally, it is always the case that if *P* holds then, unless *R* holds, *Q* will eventually hold
5. Timed Globally, Universally: Globally, it is always the case that if *P* held for *T* time units, then *S* holds.
6. Globally, Real-time Response: Globally, it is always the case that if *P* holds, the *S* eventually holds within *T* time units.

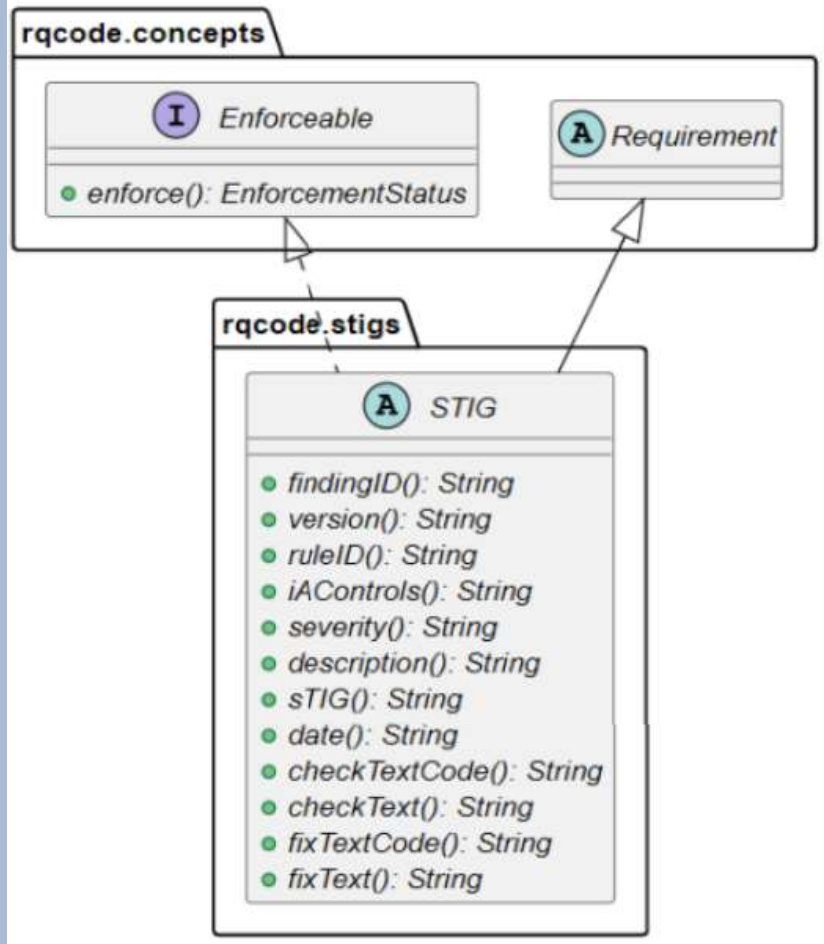
Temporal Patterns in RQCODE



Example. REQ1 to hold for 10 sec

```
1        TickIncrementRequirement tr = new TickIncrementRequirement();
2        ttr = new GlobalUniversalityTimed(tr, 10);
3        setStatement(ttr.toString());
```

Security Technical Implementation Guide



The Ubuntu operating system must not have the Network Information Service (NIS) package installed.

Overview

Finding ID	Version	Rule ID	IA Controls	Severity
V-219157	UBTU-18-010018	SV-219157r610963_rule		High

Description

Removing the Network Information Service (NIS) package decreases the risk of the accidental (or intentional) activation of NIS or NIS+ services.

STIG	Date
Canonical Ubuntu 18.04 LTS Security Technical Implementation Guide	2022-08-25

Details

Check Text (C-20882r304799_chk)

Verify that the Network Information Service (NIS) package is not installed on the Ubuntu operating system.

Check to see if the NIS package is installed with the following command:

dpkg -l | grep nis

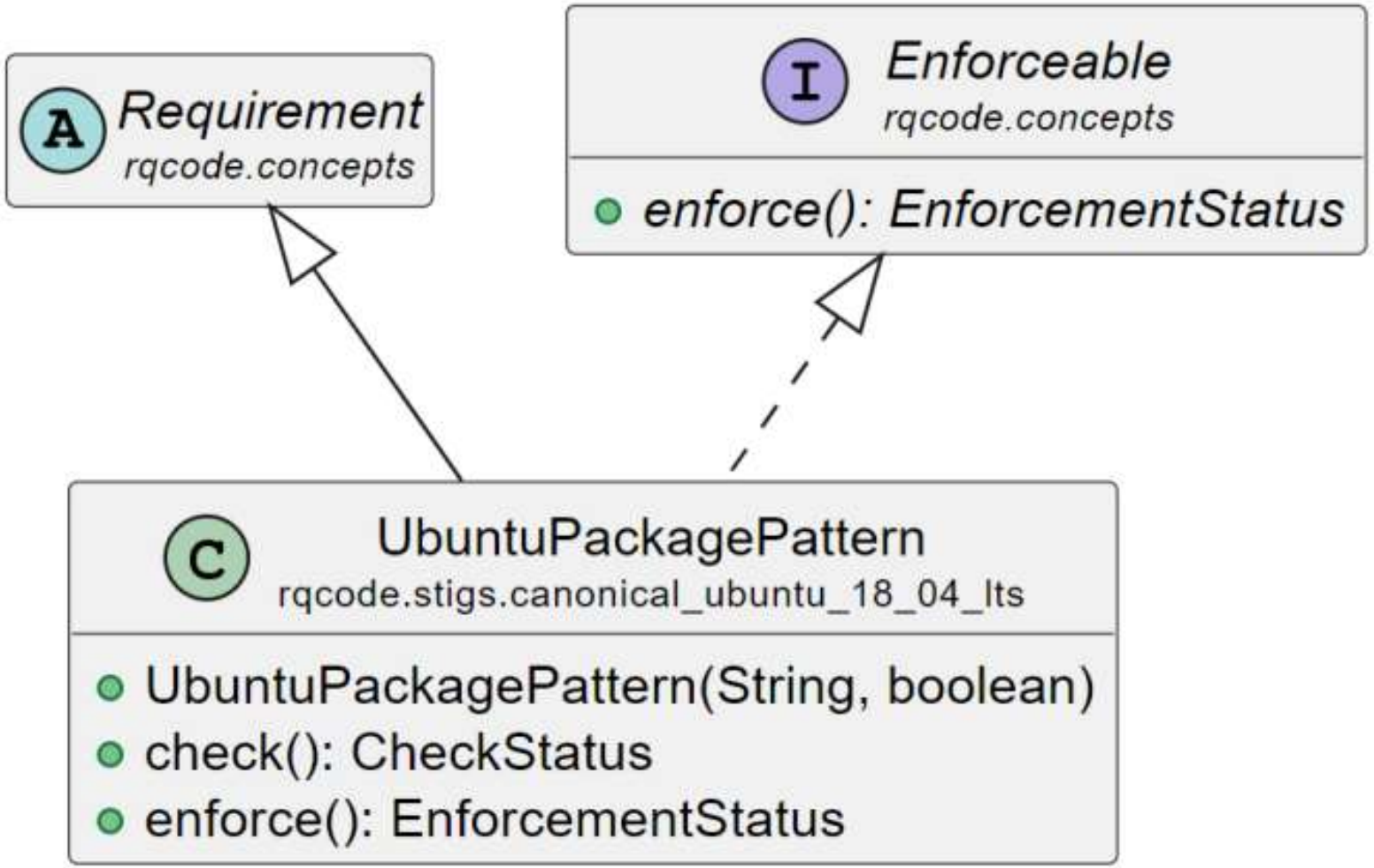
If the NIS package is installed, this is a finding.

Fix Text (F-20881r304800_fix)

Configure the Ubuntu operating system to disable non-essential capabilities by removing the Network Information Service (NIS) package from the system with the following command:

sudo apt-get remove nis

Ubuntu Package STIG RQCODE pattern



V_219157: *The Ubuntu operating system must not have the Network Information Service (NIS) package installed.*

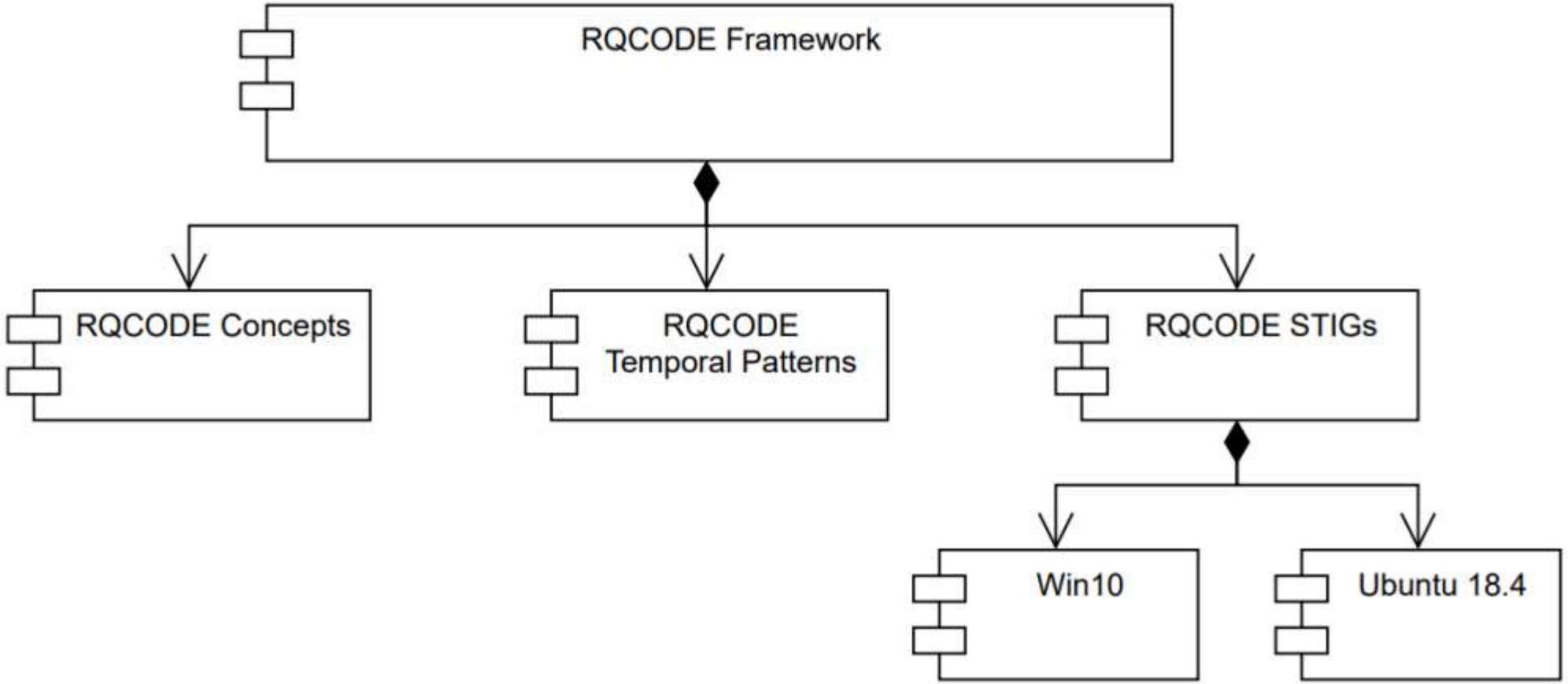
```
1  public class V_219157 extends UbuntuPackagePattern {
2      public V_219157() {
3          super("nis", false);
4      }
5  }
```

V_219157: *The Ubuntu operating system must not have the rsh-server package installed.*

```
1  public class V_219158 extends UbuntuPackagePattern {
2      public V_219158() {
3          super("rsh-server", false);
4      }
5  }
```

RQCODE Framework

https://github.com/VeriDevOps/RQCODE



Discussion

Capability\approach	RQCODE	Natural language (e.g. Textual)	Automata/graphs (e.g. FSP/LTSA)	Semi-formal (e.g. SysML)	Mathematical (e.g. EventB)
Formalisation of functional requirements	+	-	+++	+	+++
Formalisation of non-functional requirements	+	+/-	-	+/-	-
Validation/ Verification	++	-	++	+/-	+++
Traceability	++	-	+	+/-	
Reuse	+++	-	+	+	++
Maintenance	+++	-	+	+/-	-
Tool support	+++	+++	++	++	+
Learning barrier*	+	-	++	+	++
Analysis*	++	-	+++	-	-
Developer friendliness*	++	+++	+	-	-

Current state

1. RQCODE library is available
   a. Concepts, Temporal patterns, Examples
   b. Windows 10 STIGS
   c. Ubuntu STIGs examples
2. We gather feedback
   a. Benefits hypothesis validation
   b. Relevance
   c. Improvements/New features

Next

Time	Duration	Topic	Presenter	Organization
9:30	20 mins	VeriDevOps Technical Introduction	Andrey Sadovykh	SOFTEAM
Part I: Security Requirements Engineering				
9:50	20 mins	A Taxonomy of Vulnerabilities, Attacks, and Security Solutions in Industrial PLCs.	Eduard Paul Enoiu	Mälardalen University
10:10	20 mins	Natural Language Processing with Machine Learning for Security Requirements Analysis - Practical Approaches.	Andrey Sadovykh	SOFTEAM
10:30	20 mins	Security Requirements Formalization with RQCODE.	Andrey Sadovykh	SOFTEAM
10:50	10 mins	break	/	/

Agenda - Part 2 - Prevention

Part II: Prevention at Development Time				
11:00	20 mins	Vulnerability Detection and Response: Current Status and New Approaches	Jose Luis Flores	IKER
11:20	20 mins	Metamorphic Testing for Verification and Fault Localization in Industrial Control Systems	Gaadha Sudheerbabu	Åbo Akademi University
11:40	20 mins	Interactive Application Security Testing with Hybrid Fuzzing and Statistical Estimators	Ramon Barakat	FFK
12:00	10 mins	break	/	/

Thank You

andrey.sadovykh@softeam.fr

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 957212