On Language for Algorithmic Law

Joost J. Joosten

Universitat de Barcelona

Universität Passau
Passau, June 1

Disclaimer

- We have no ambition on answering the question

Disclaimer

- We have no ambition on answering the question

  Is automated law enforcement a good idea?

Disclaimer

- We have no ambition on answering the question

  Is automated law enforcement a good idea?

- Practical experience seems to have answered this question:

Disclaimer

- We have no ambition on answering the question

  Is automated law enforcement a good idea?

- Practical experience seems to have answered this question:

  They simply do it.

Disclaimer

- We have no ambition on answering the question

  Is automated law enforcement a good idea?

- Practical experience seems to have answered this question:

  They simply do it.

- Our position:

Disclaimer

- We have no ambition on answering the question

  Is automated law enforcement a good idea?

- Practical experience seems to have answered this question:

  They simply do it.

- Our position:

  If you do it, then do it properly.

Disclaimer

- We have no ambition on answering the question

  Is automated law enforcement a good idea?

- Practical experience seems to have answered this question:

  They simply do it.

- Our position:

  If you do it, then do it properly.

Disclaimer

- We have no ambition on answering the question

  Is automated law enforcement a good idea?

- Practical experience seems to have answered this question:

  They simply do it.

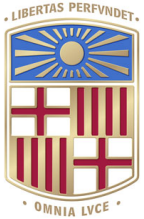
- Our position:

  If you do it, then do it properly.

- *What does that mean:*

  Do it properly?

The methodology of our research lab



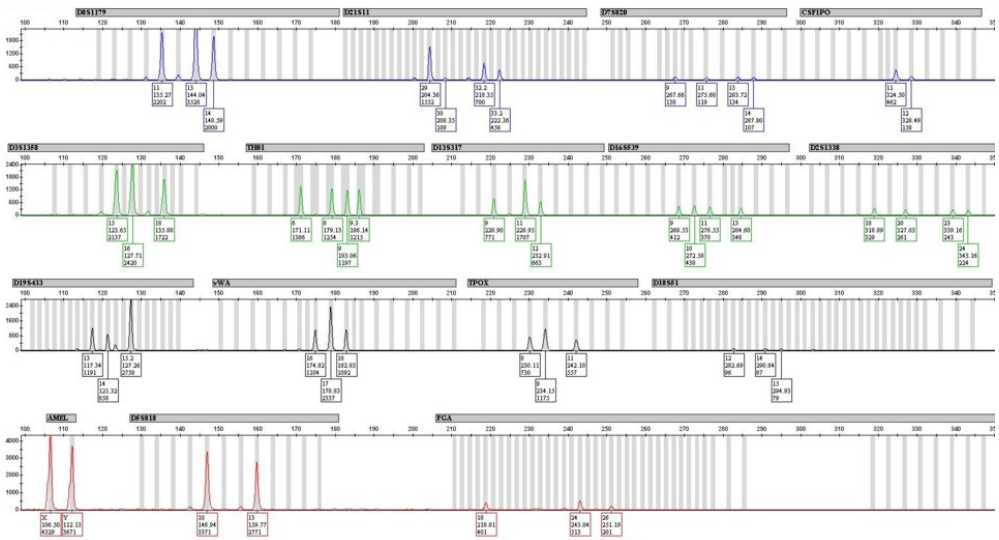
UNIVERSITAT DE BARCELONA

Business focussed: from concrete to abstract

Only rule-based automated decision making.

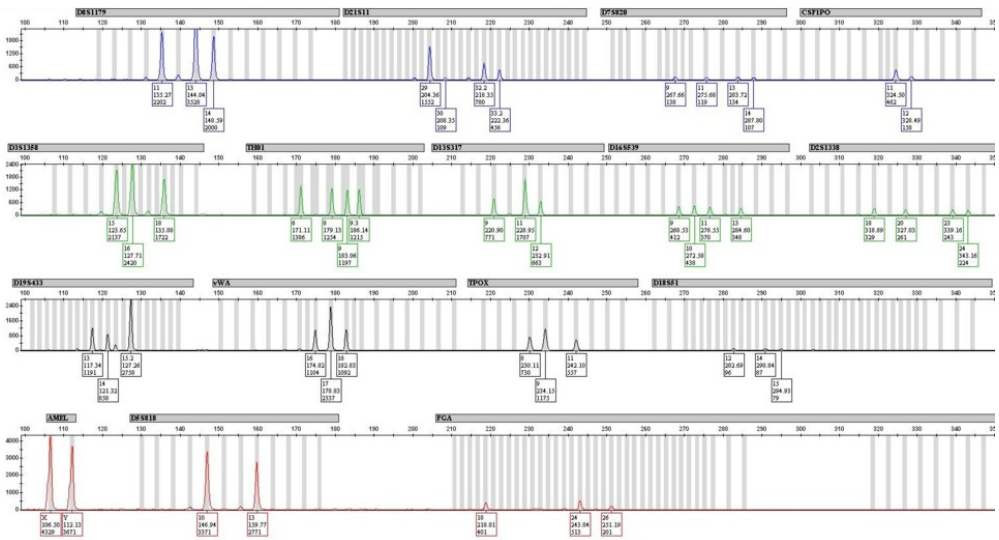
Doubtful evidence

- In three law-suits in the USA, the defence requested to **open the proprietary source code to the jury** of DNA sequencing software since there were some doubts.
  - STRmix



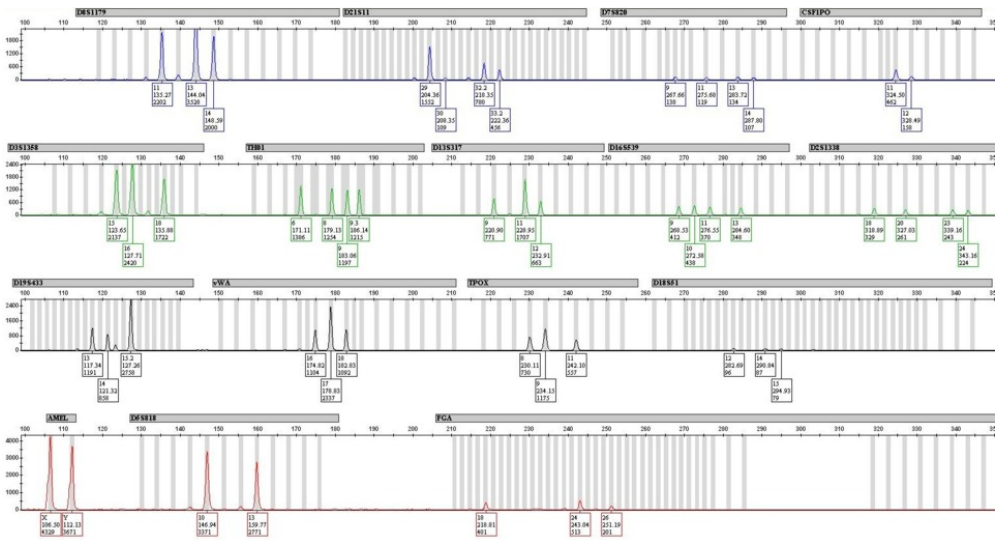
Doubtful evidence

- In three law-suits in the USA, the defence requested to **open the proprietary source code to the jury** of DNA sequencing software since there were some doubts.
  - STRmix
  - FST



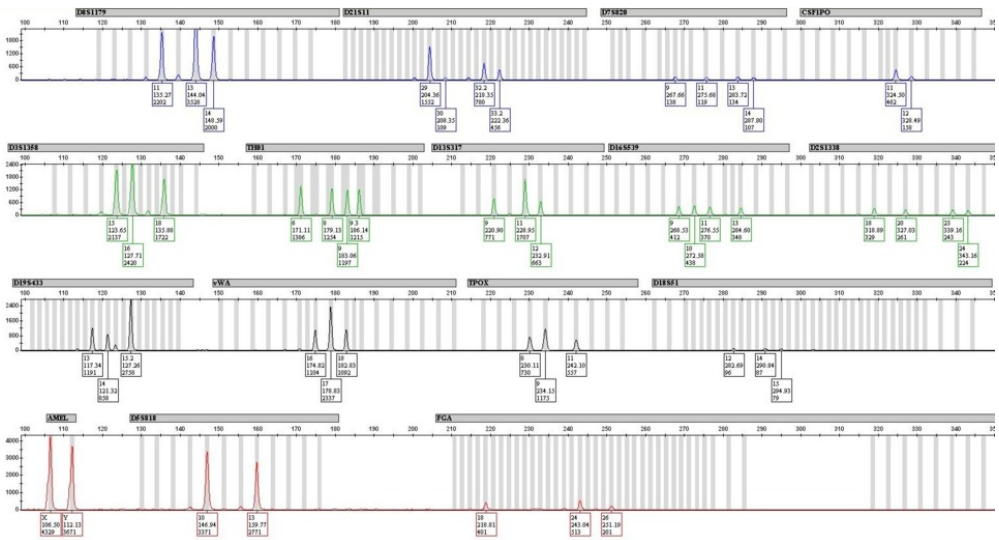
Doubtful evidence

- In three law-suits in the USA, the defence requested to **open the proprietary source code to the jury** of DNA sequencing software since there were some doubts.
  - STRmix
  - FST
  - TrueAllele (not granted)



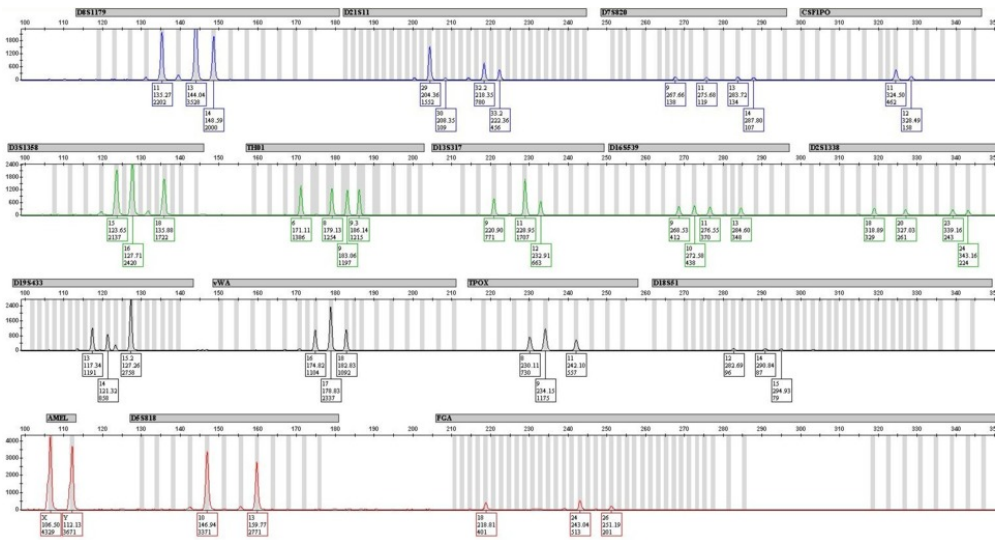
Doubtful evidence

- In three law-suits in the USA, the defence requested to **open the proprietary source code to the jury** of DNA sequencing software since there were some doubts.
  - STRmix
  - FST
  - TrueAllele (not granted)
- In two cases the request was granted

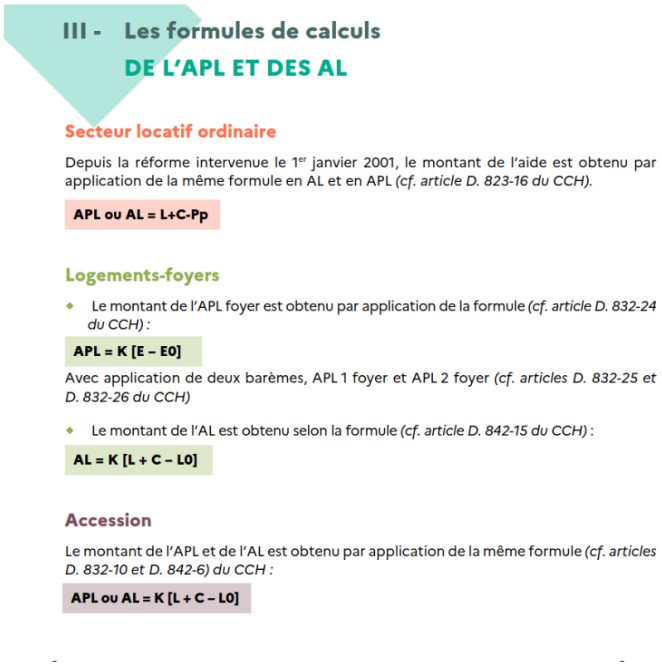


Doubtful evidence

- In three law-suits in the USA, the defence requested to **open the proprietary source code to the jury** of DNA sequencing software since there were some doubts.
  - STRmix
  - FST
  - TrueAllele (not granted)
- In two cases the request was granted
- Again, access to the source code will not solve all problems!



Some examples of automated decision making



III - Les formules de calculs
DE L'APL ET DES AL

Secteur locatif ordinaire

Depuis la réforme intervenue le 1er janvier 2001, le montant de l'aide est obtenu par application de la même formule en AL et en APL *(cf. article D. 823-16 du CCH).*

APL ou AL = L+C-Pp

Logements-foyers

- Le montant de l'APL foyer est obtenu par application de la formule *(cf. article D. 832-24 du CCH) :*

APL = K [E – E0]

Avec application de deux barèmes, APL 1 foyer et APL 2 foyer *(cf. articles D. 832-25 et D. 832-26 du CCH)*

- Le montant de l'AL est obtenu selon la formule *(cf. article D. 842-15 du CCH) :*

AL = K [L + C – L0]

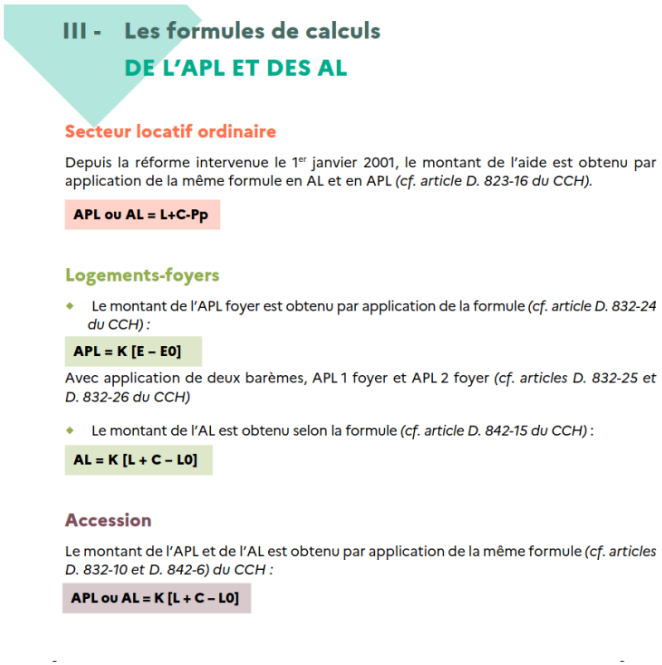
Accession

Le montant de l'APL et de l'AL est obtenu par application de la même formule *(cf. articles D. 832-10 et D. 842-6) du CCH :*

APL ou AL = K [L + C – L0]

- Bonus payment system of the French Army: Louvois/SourceSolde

Some examples of automated decision making



III - Les formules de calculs
DE L'APL ET DES AL

Secteur locatif ordinaire

Depuis la réforme intervenue le 1er janvier 2001, le montant de l'aide est obtenu par application de la même formule en AL et en APL *(cf. article D. 823-16 du CCH).*

APL ou AL = L+C-Pp

Logements-foyers

- Le montant de l'APL foyer est obtenu par application de la formule *(cf. article D. 832-24 du CCH) :*

APL = K [E – E0]

Avec application de deux barèmes, APL 1 foyer et APL 2 foyer *(cf. articles D. 832-25 et D. 832-26 du CCH)*

- Le montant de l'AL est obtenu selon la formule *(cf. article D. 842-15 du CCH) :*

AL = K [L + C – L0]

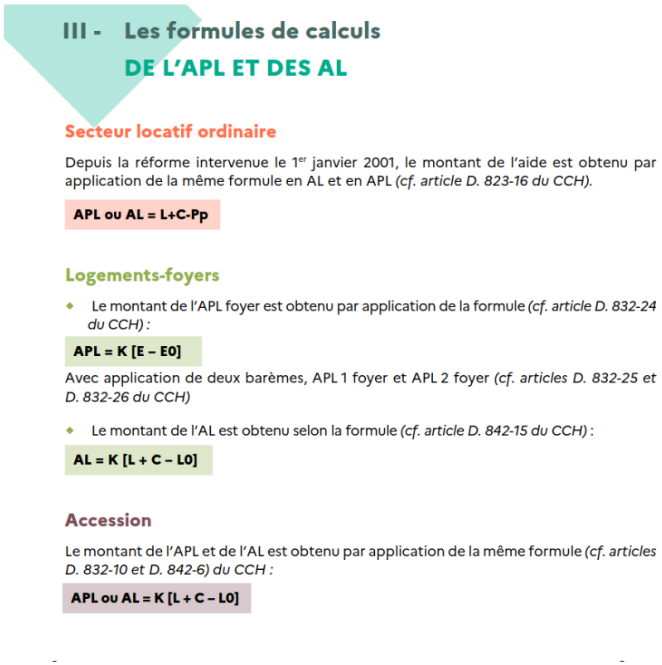
Accession

Le montant de l'APL et de l'AL est obtenu par application de la même formule *(cf. articles D. 832-10 et D. 842-6) du CCH :*

APL ou AL = K [L + C – L0]

- Bonus payment system of the French Army: Louvois/SourceSolde
- In 2012: 465 M € incorrect payments

Some examples of automated decision making



III - Les formules de calculs
DE L'APL ET DES AL

Secteur locatif ordinaire

Depuis la réforme intervenue le 1er janvier 2001, le montant de l'aide est obtenu par application de la même formule en AL et en APL *(cf. article D. 823-16 du CCH).*

APL ou AL = L+C-Pp

Logements-foyers

- Le montant de l'APL foyer est obtenu par application de la formule *(cf. article D. 832-24 du CCH) :*

APL = K [E – E0]

Avec application de deux barèmes, APL 1 foyer et APL 2 foyer *(cf. articles D. 832-25 et D. 832-26 du CCH)*

- Le montant de l'AL est obtenu selon la formule *(cf. article D. 842-15 du CCH) :*

AL = K [L + C – L0]

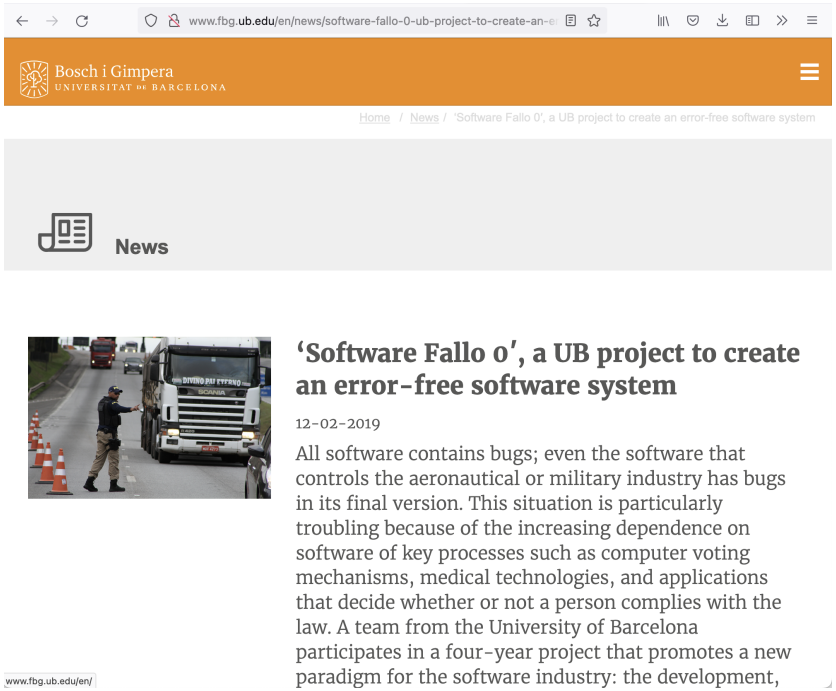
Accession

Le montant de l'APL et de l'AL est obtenu par application de la même formule *(cf. articles D. 832-10 et D. 842-6) du CCH :*

APL ou AL = K [L + C – L0]

- Bonus payment system of the French Army: Louvois/SourceSolde
- In 2012: 465 M € incorrect payments
- It left some soldiers and their families without any income at all for months!

Tachographs can even imprison drivers in Europe



www.fbg.ub.edu/en/news/software-fallo-0-ub-project-to-create-an-e

Bosch i Gimpera
UNIVERSITAT DE BARCELONA

Home / News / 'Software Fallo 0', a UB project to create an error-free software system

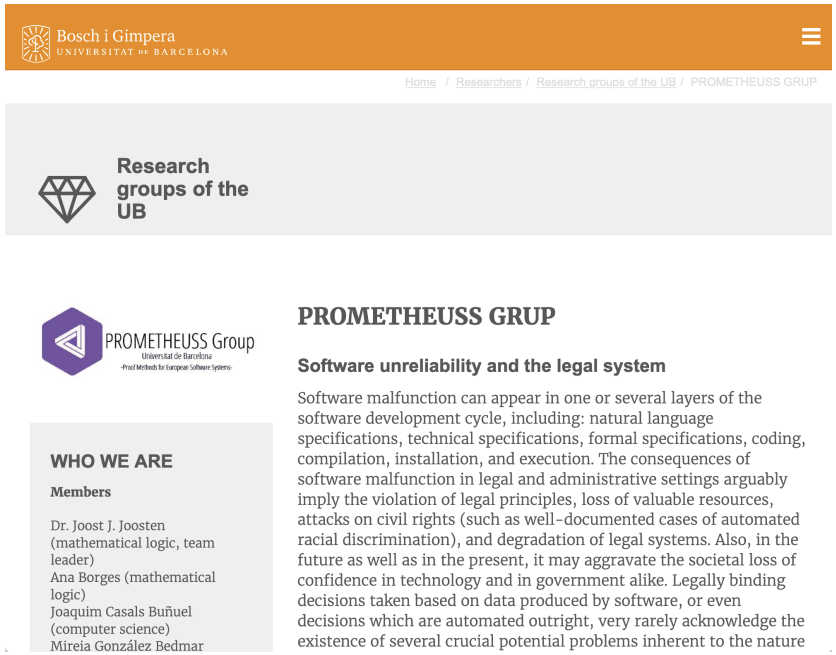
News

'Software Fallo 0', a UB project to create an error-free software system

12-02-2019

All software contains bugs; even the software that controls the aeronautical or military industry has bugs in its final version. This situation is particularly troubling because of the increasing dependence on software of key processes such as computer voting mechanisms, medical technologies, and applications that decide whether or not a person complies with the law. A team from the University of Barcelona participates in a four-year project that promotes a new paradigm for the software industry: the development,

www.fbg.ub.edu/en/



Bosch i Gimpera
UNIVERSITAT DE BARCELONA

Home / Researchers / Research groups of the UB / PROMETHEUSS GRUP

Research groups of the UB

PROMETHEUSS Group
Universitat de Barcelona
-Proof Methods for European Software Systems-

WHO WE ARE

Members

Dr. Joost J. Joosten (mathematical logic, team leader)
Ana Borges (mathematical logic)
Joaquim Casals Buñuel (computer science)
Mireia González Bedmar

PROMETHEUSS GRUP

Software unreliability and the legal system

Software malfunction can appear in one or several layers of the software development cycle, including: natural language specifications, technical specifications, formal specifications, coding, compilation, installation, and execution. The consequences of software malfunction in legal and administrative settings arguably imply the violation of legal principles, loss of valuable resources, attacks on civil rights (such as well-documented cases of automated racial discrimination), and degradation of legal systems. Also, in the future as well as in the present, it may aggravate the societal loss of confidence in technology and in government alike. Legally binding decisions taken based on data produced by software, or even decisions which are automated outright, very rarely acknowledge the existence of several crucial potential problems inherent to the nature

Covenant between the University of Barcelona (FBG), Formal Vindications S.L. & Guretruck S.L.

Closer to (my) home: Civio vs Bosco

This article belongs to the debate » The Rule of Law versus the Rule of the Algorithm

02 April 2022

The Paradox of Efficiency: Frictions Between Law and Algorithms

On the 13th of January 2022, a Spanish Administrative court ruled in favour of algorithmic opacity. Fundación Civio, an independent foundation that monitors and accounts public authorities, reported that an algorithm used by the government was committing errors.[1)] BOSCO, the name of the application which contained the algorithm, was implemented by the Spanish public administration to more efficiently identify citizens eligible for grants to pay electricity bills. Meanwhile, Civio designed a web app to inform citizens whether they would be entitled for this grant.[2)] Thousands of citizens used this application and some of them reported that, while Civio's web app suggested

Ana Valdivia
Dr Ana Valdivia is a Postdoctoral Researcher at King's College London (ERC Security Flows). She examines how algorithms impact on people's life from a technical, political, and legal perspective.

Javier de la Cueva
Javier de la Cueva is a lawyer, lecturer and researcher in topics related to open knowledge, ethics and the digital world.

Explore posts related to this:
Algorithmic Efficiency, Algorithmic Justice, Rule of Law, Rule of the Algorithm

The Bosco computer program : errors in the computation of the social welfare bonuses

Least requirement: access to source code

In France it is mandatory to publish source code of software that is used in public administration.

However, access to source code will not resolve all problems

Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...

Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...
- SyRI: System Risk Indication

Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...
- SyRI: System Risk Indication
- legal algorithmic instrument used by Dutch government

Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...
- SyRI: System Risk Indication
- legal algorithmic instrument used by Dutch government
- For welfare fraud prevention (benefits, allowances, taxes, ...)

Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...
- SyRI: System Risk Indication
- legal algorithmic instrument used by Dutch government
- For welfare fraud prevention (benefits, allowances, taxes, ...)
- seriously affected/destroyed thousands of Dutch households

Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...
- SyRI: System Risk Indication
- legal algorithmic instrument used by Dutch government
- For welfare fraud prevention (benefits, allowances, taxes, ...)
- seriously affected/destroyed thousands of Dutch households
- urged Rutte III to resign

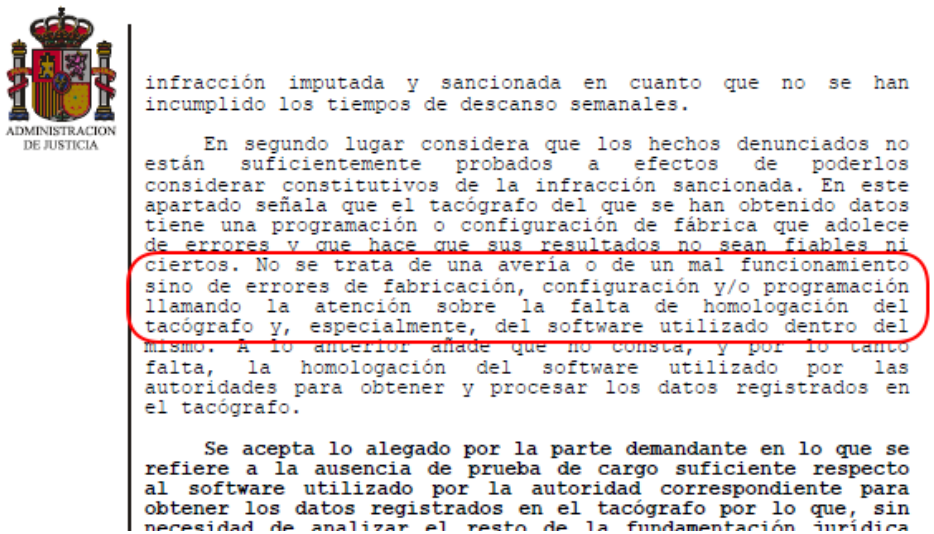
Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...
- SyRI: System Risk Indication
- legal algorithmic instrument used by Dutch government
- For welfare fraud prevention (benefits, allowances, taxes, ...)
- seriously affected/destroyed thousands of Dutch households
- urged Rutte III to resign
- which was later re-elected in an isomorphic constellation...

Closer to (my former) home: SyRI

- Euphemistically called *Toeslagenaffaire*...
- SyRI: System Risk Indication
- legal algorithmic instrument used by Dutch government
- For welfare fraud prevention (benefits, allowances, taxes, ...)
- seriously affected/destroyed thousands of Dutch households
- urged Rutte III to resign
- which was later re-elected in an isomorphic constellation...
- call for regulatory overview: software certification, European AI Act, etc...

Society calls out for certification of software



ADMINISTRACION DE JUSTICIA

infracción imputada y sancionada en cuanto que no se han incumplido los tiempos de descanso semanales.

En segundo lugar considera que los hechos denunciados no están suficientemente probados a efectos de poderlos considerar constitutivos de la infracción sancionada. En este apartado señala que el tacógrafo del que se han obtenido datos tiene una programación o configuración de fábrica que adolece de errores y que hace que sus resultados no sean fiables ni ciertos. No se trata de una avería o de un mal funcionamiento sino de errores de fabricación, configuración y/o programación llamando la atención sobre la falta de homologación del tacógrafo y, especialmente, del software utilizado dentro del mismo. A lo anterior añade que no consta, y por lo tanto falta, la homologación del software utilizado por las autoridades para obtener y procesar los datos registrados en el tacógrafo.

Se acepta lo alegado por la parte demandante en lo que se refiere a la ausencia de prueba de cargo suficiente respecto al software utilizado por la autoridad correspondiente para obtener los datos registrados en el tacógrafo por lo que, sin

Sentence Number: 30/2019, CONTENCIOSO/ADMTVO court N. 4 of Valladolid

What is certification?

- Is it just a matter of trust? (combined with some sanity checks and experience)

What is certification?

- Is it just a matter of trust? (combined with some sanity checks and experience)
- Certificate $\Longrightarrow$ something is certain

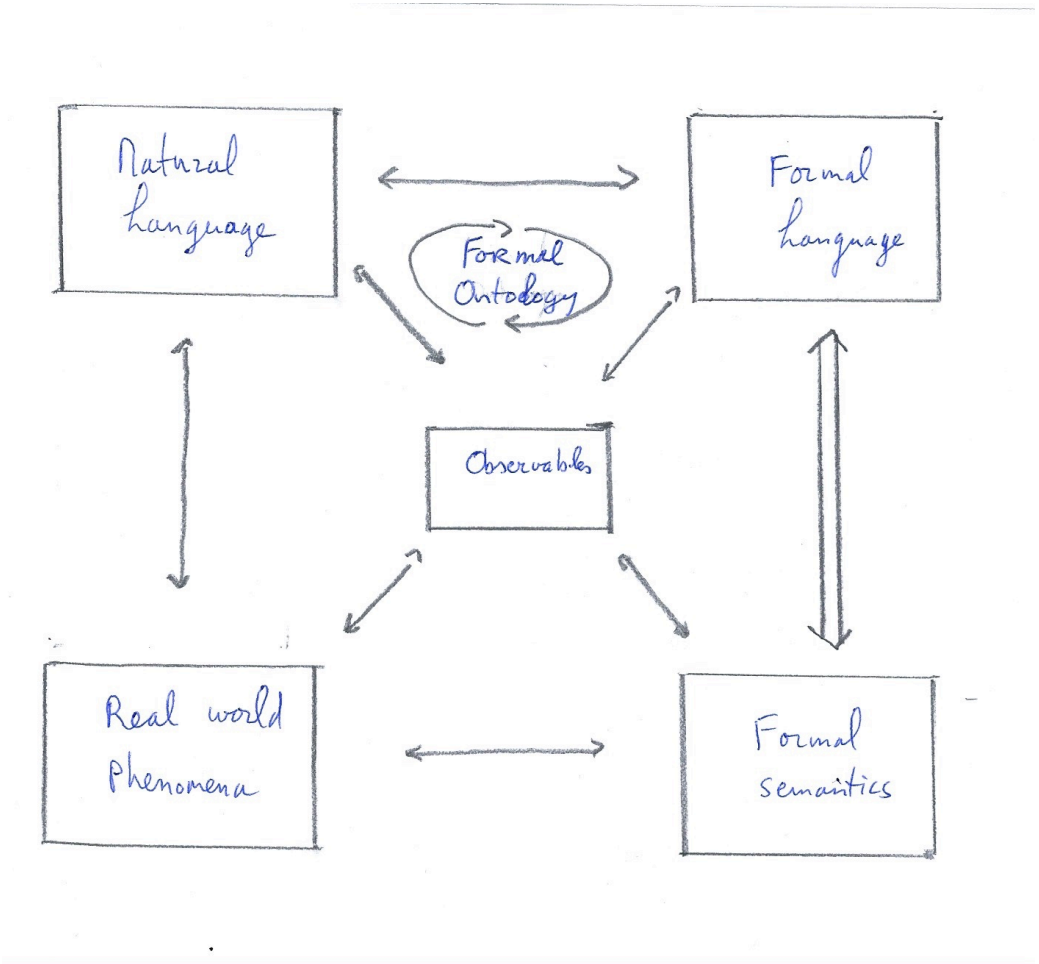
What is certification?

- Is it just a matter of trust? (combined with some sanity checks and experience)
- Certificate $\Longrightarrow$ something is certain
- Verify $\Longrightarrow$ something is veridical

What is certification?

- Is it just a matter of trust? (combined with some sanity checks and experience)
- Certificate $\Longrightarrow$ something is certain
- Verify $\Longrightarrow$ something is veridical
- Nobody really knows, and it is a very difficult question!

Formal methods



Only at the double arrows can we attain mathematical certainty
(modulo some very reasonable assumptions)

The impossibility of unrestricted certification

Alan Turing

- A mathematical theorem:

The impossibility of unrestricted certification

Alan Turing

- A mathematical theorem:
- Unrestricted certification is impossible.

The impossibility of unrestricted certification

Alan Turing

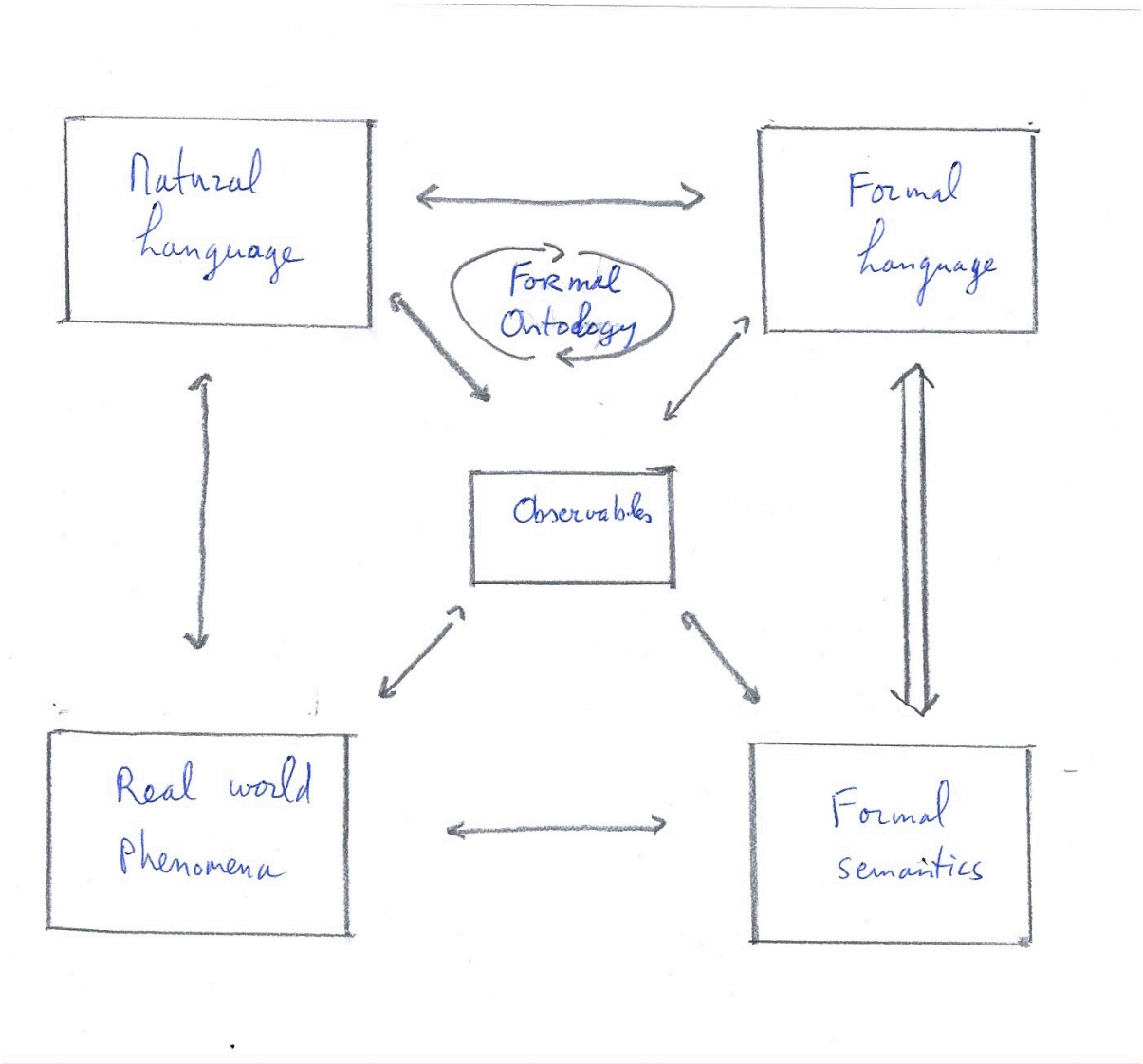
- A mathematical theorem:
- Unrestricted certification is impossible.
- Depending on the expressivity of the language:

The impossibility of unrestricted certification

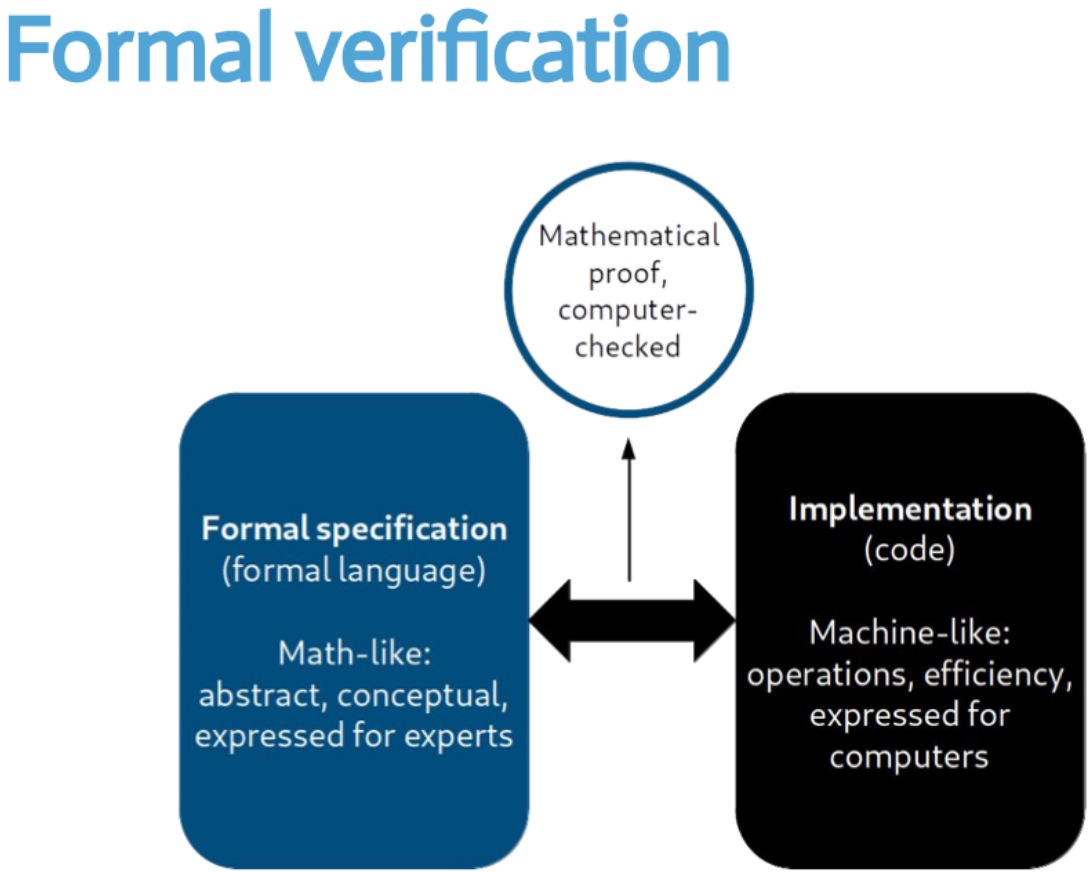
Alan Turing

- A mathematical theorem:
- Unrestricted certification is impossible.
- Depending on the expressivity of the language:
- Restricted certification is possible.

Formal methods



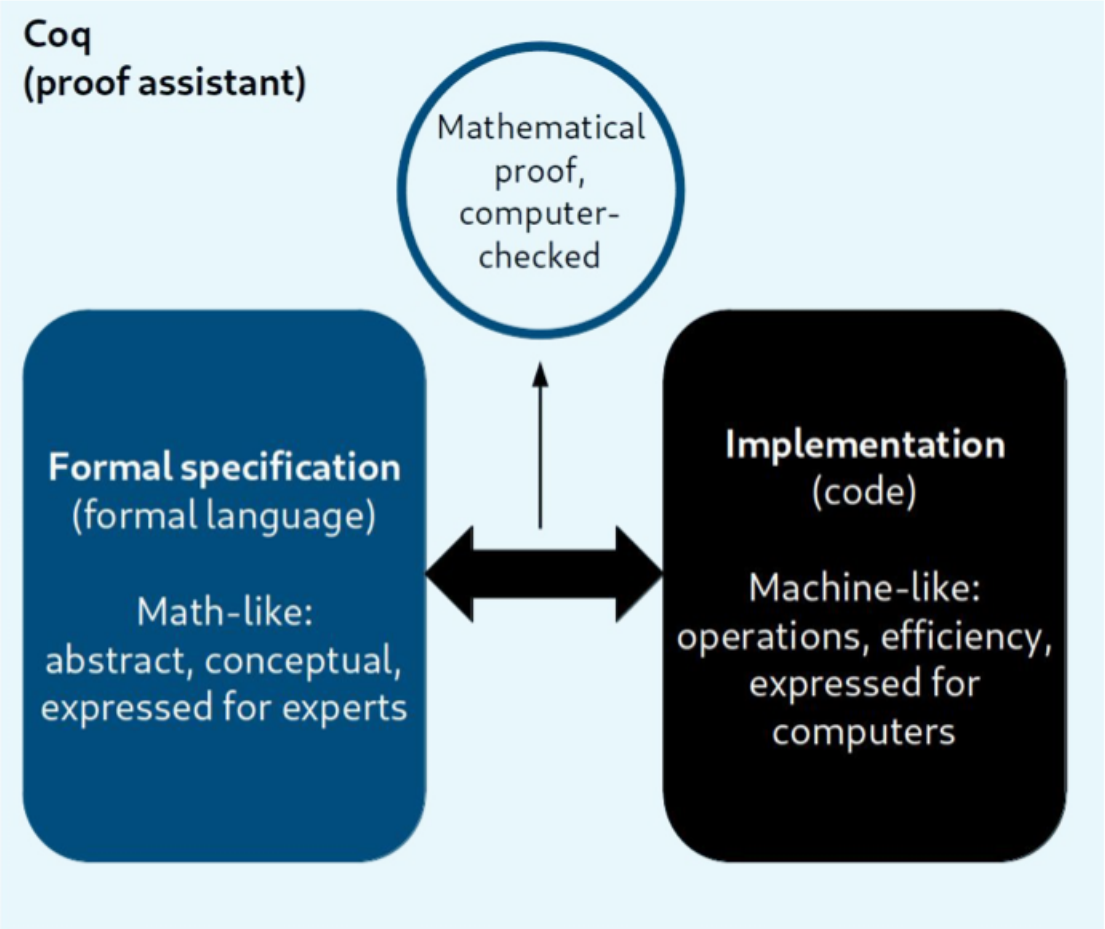
Formal verification of software correctness



Slides FV: González Bedmar

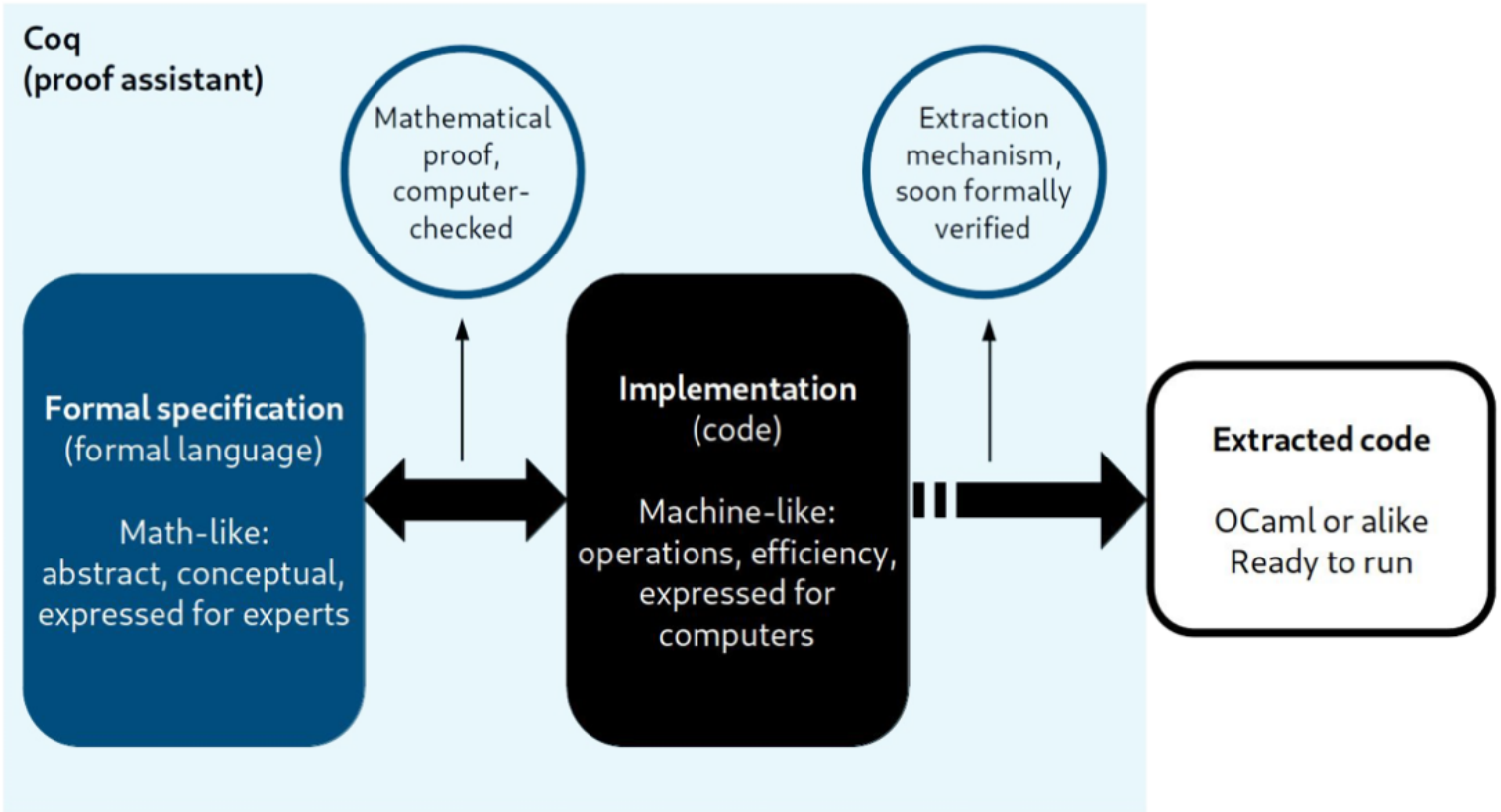
Mathematical precision

Formal verification

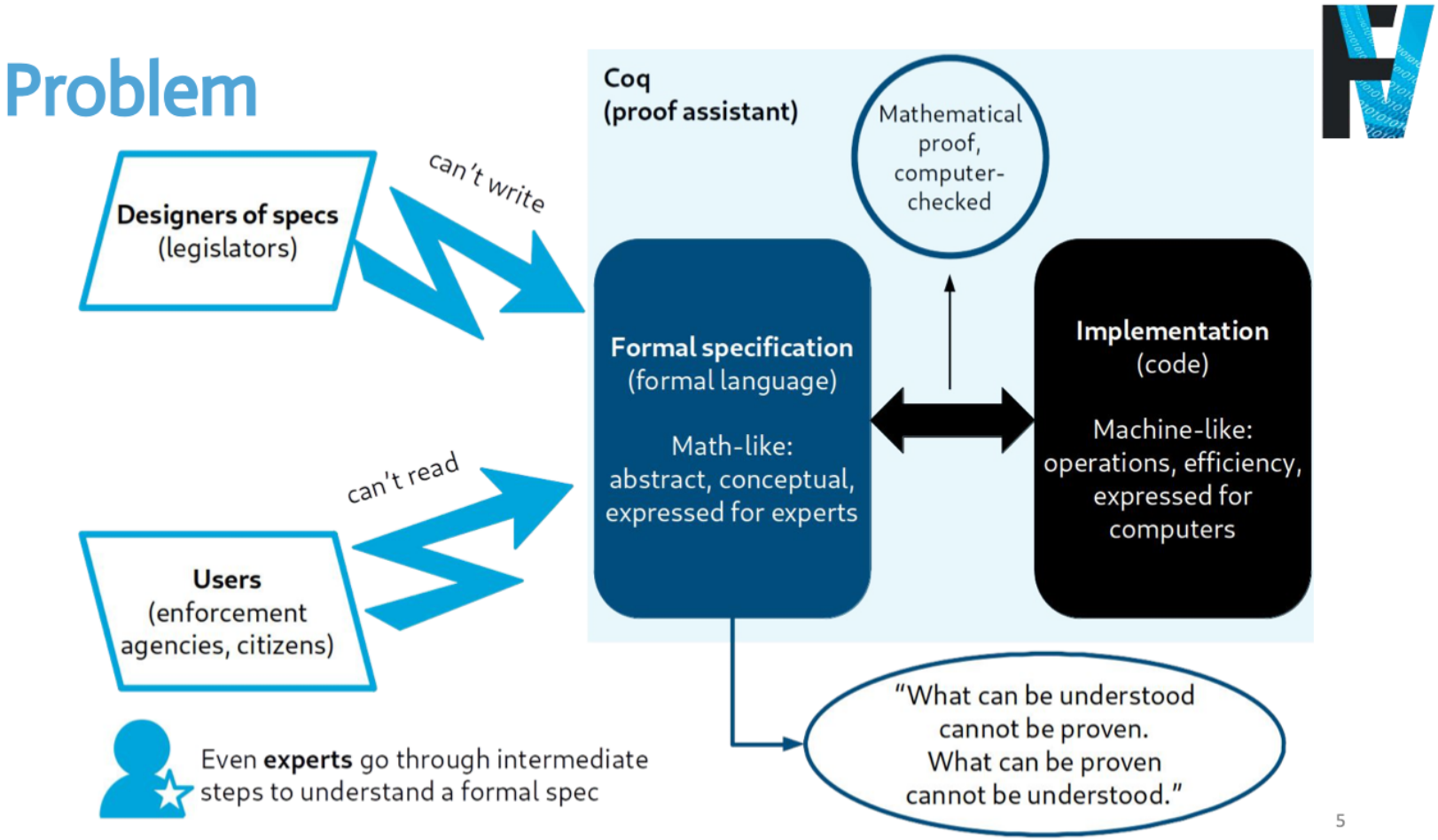


Program extraction

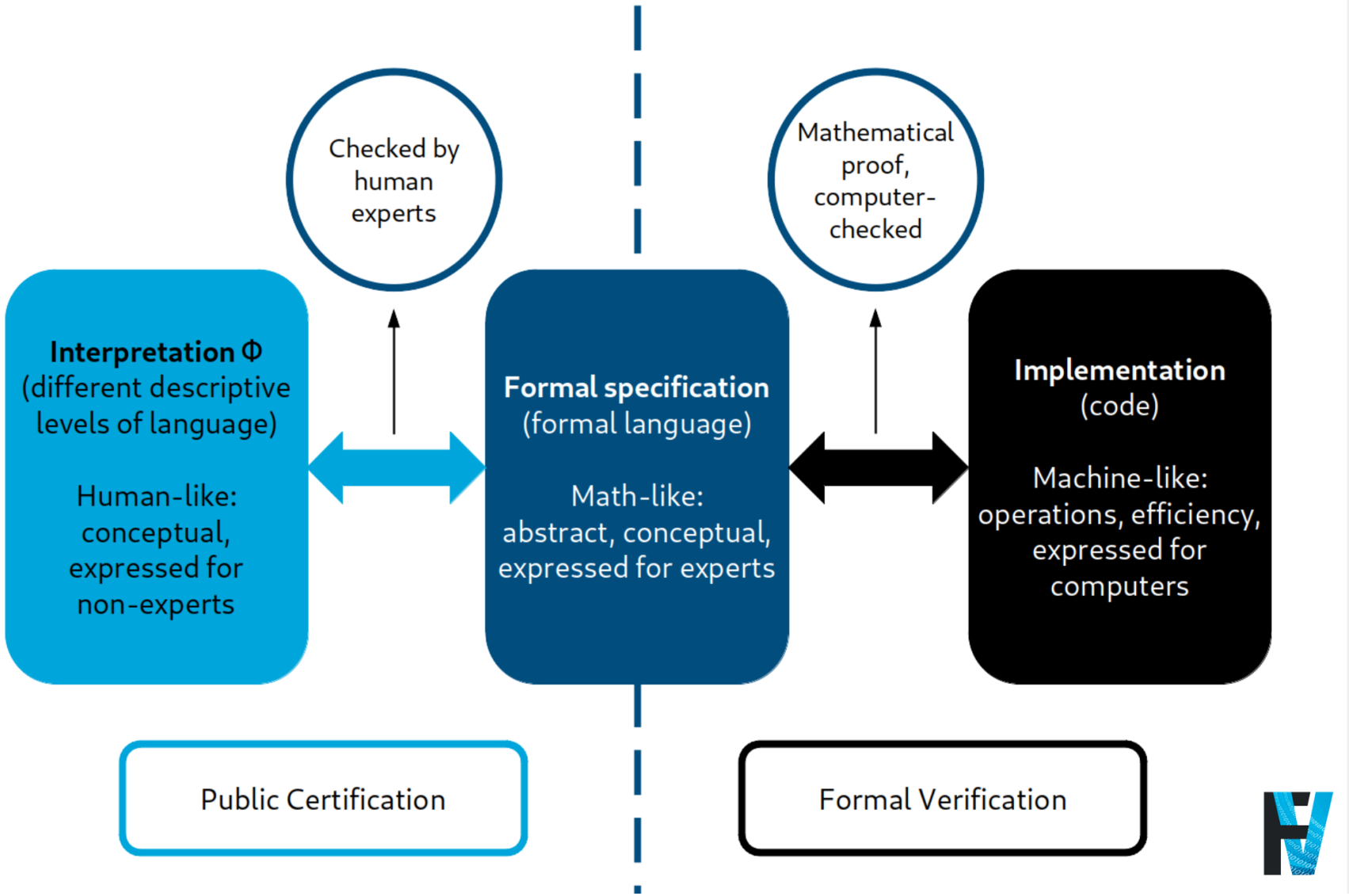
Formal verification



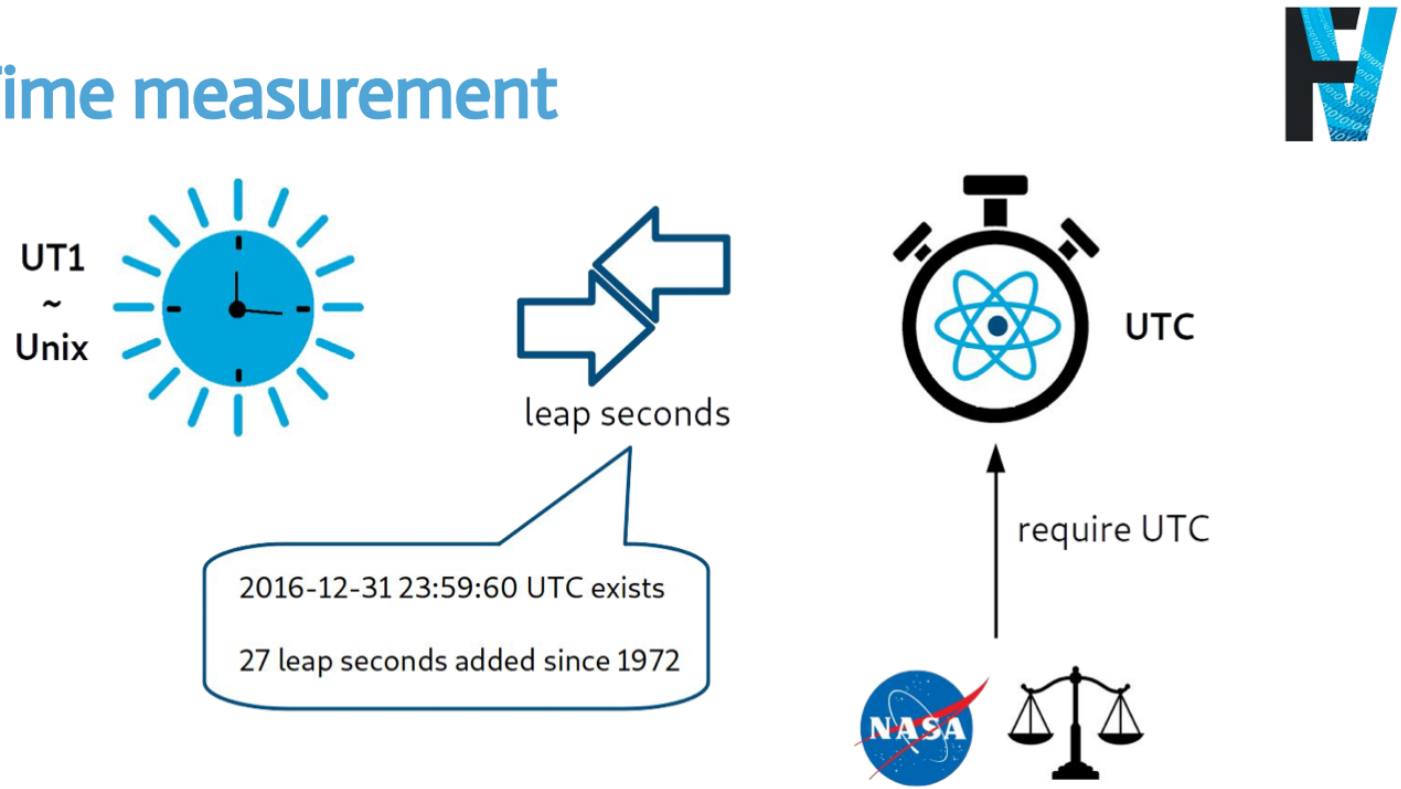
Program vs specification language



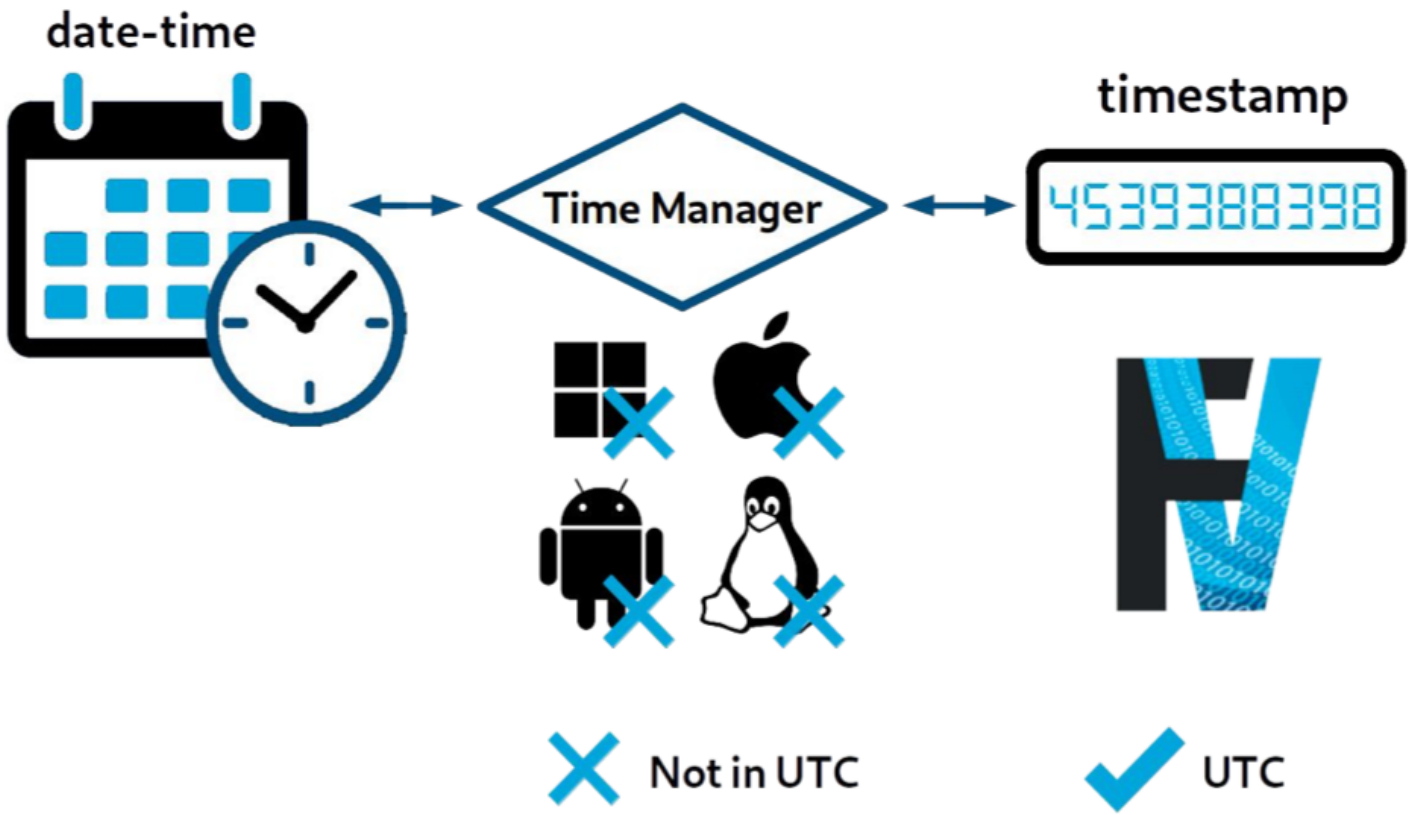
One solution: Public Certification



Time library

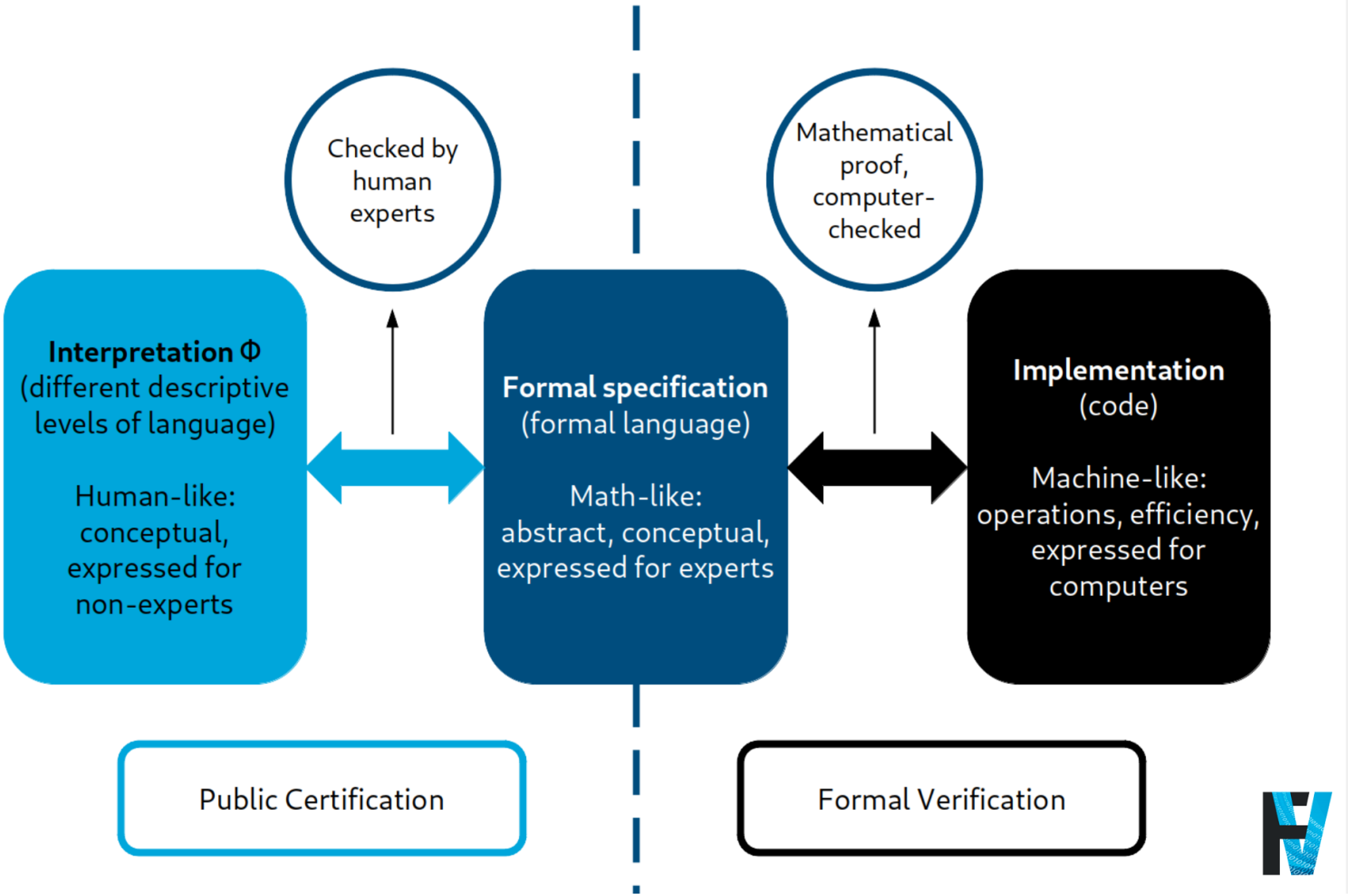


Time formats and managers

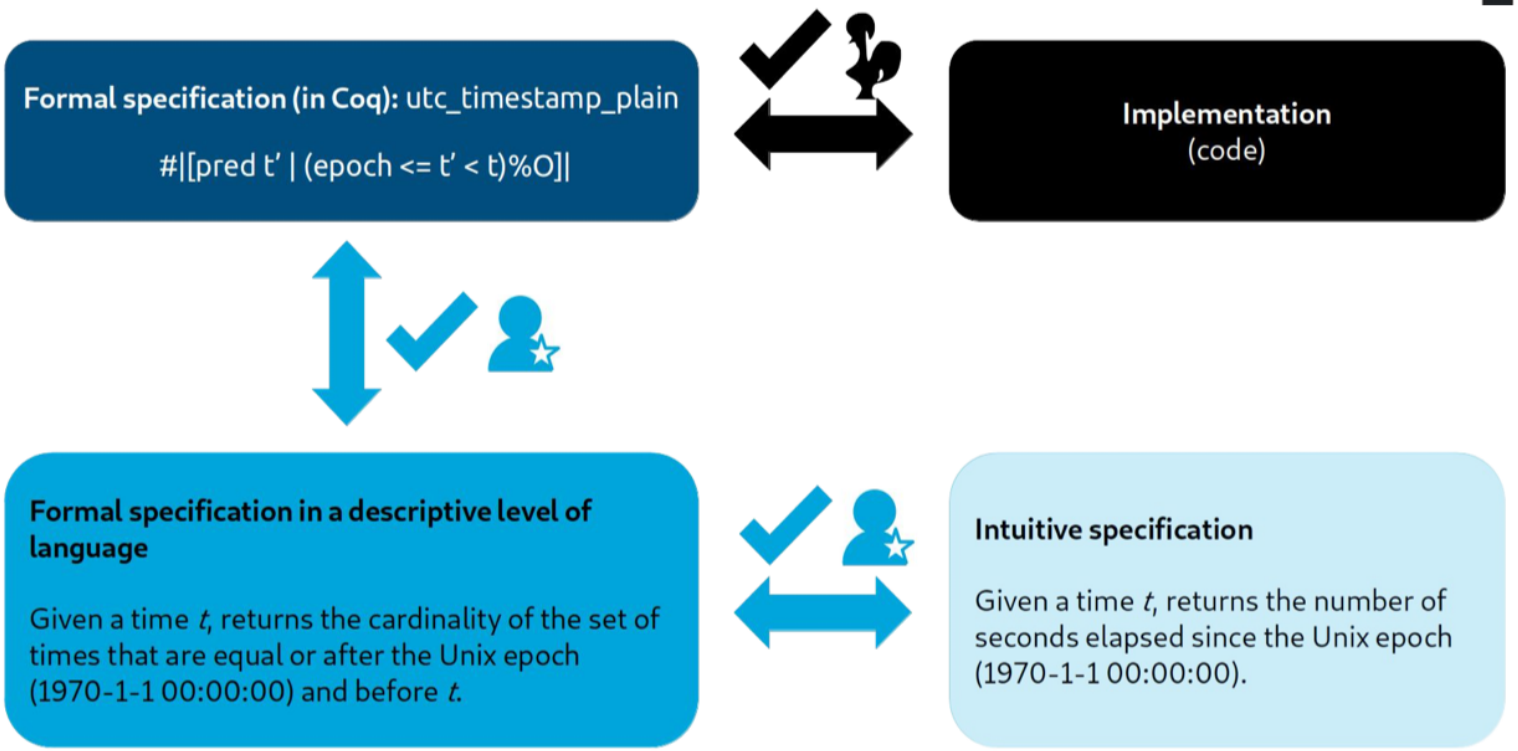


Most important feature: formally verified!

What is certification?



Example in FV Time



Around one-thousand times more expensive!

Lab activities

- Regulation analysis via logical/mathematical analysis

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?
  - Is the implied behaviour of the regulation desirable?

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?
  - Is the implied behaviour of the regulation desirable?
    - Shift-invariance of labelling, locality of legality checks, etc.

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?
  - Is the implied behaviour of the regulation desirable?
    - Shift-invariance of labelling, locality of legality checks, etc.
    - But also: if x and y are similar data containers (e.g., formal repr. of persons) with the only difference that x has one passport and y has two passports, will the program behave the same for x and y?

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?
  - Is the implied behaviour of the regulation desirable?
    - Shift-invariance of labelling, locality of legality checks, etc.
    - But also: if x and y are similar data containers (e.g., formal repr. of persons) with the only difference that x has one passport and y has two passports, will the program behave the same for x and y?
  - Are the algorithms implied computationally feasible?

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?
  - Is the implied behaviour of the regulation desirable?
    - Shift-invariance of labelling, locality of legality checks, etc.
    - But also: if x and y are similar data containers (e.g., formal repr. of persons) with the only difference that x has one passport and y has two passports, will the program behave the same for x and y?
  - Are the algorithms implied computationally feasible?
- Implement software in a ZERO-ERROR fashion using proof assistants.

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?
  - Is the implied behaviour of the regulation desirable?
    - Shift-invariance of labelling, locality of legality checks, etc.
    - But also: if x and y are similar data containers (e.g., formal repr. of persons) with the only difference that x has one passport and y has two passports, will the program behave the same for x and y?
  - Are the algorithms implied computationally feasible?
- Implement software in a ZERO-ERROR fashion using proof assistants.
- Develop general purpose models with the above considerations taken into account so that zero-error software scales.

Lab activities

- Regulation analysis via logical/mathematical analysis
  - Is the regulation consistent?
  - Is the implied behaviour of the regulation desirable?
    - Shift-invariance of labelling, locality of legality checks, etc.
    - But also: if x and y are similar data containers (e.g., formal repr. of persons) with the only difference that x has one passport and y has two passports, will the program behave the same for x and y?
  - Are the algorithms implied computationally feasible?
- Implement software in a ZERO-ERROR fashion using proof assistants.
- Develop general purpose models with the above considerations taken into account so that zero-error software scales.
- Provide verified software with a dialogue fragment that enables a possible rudimentary dialogue between the user and the software about the software's behaviour

Further problems

- Develop explanatory certificates

Further problems

- Develop explanatory certificates
- Zero-knowledge certificates for proprietary software!

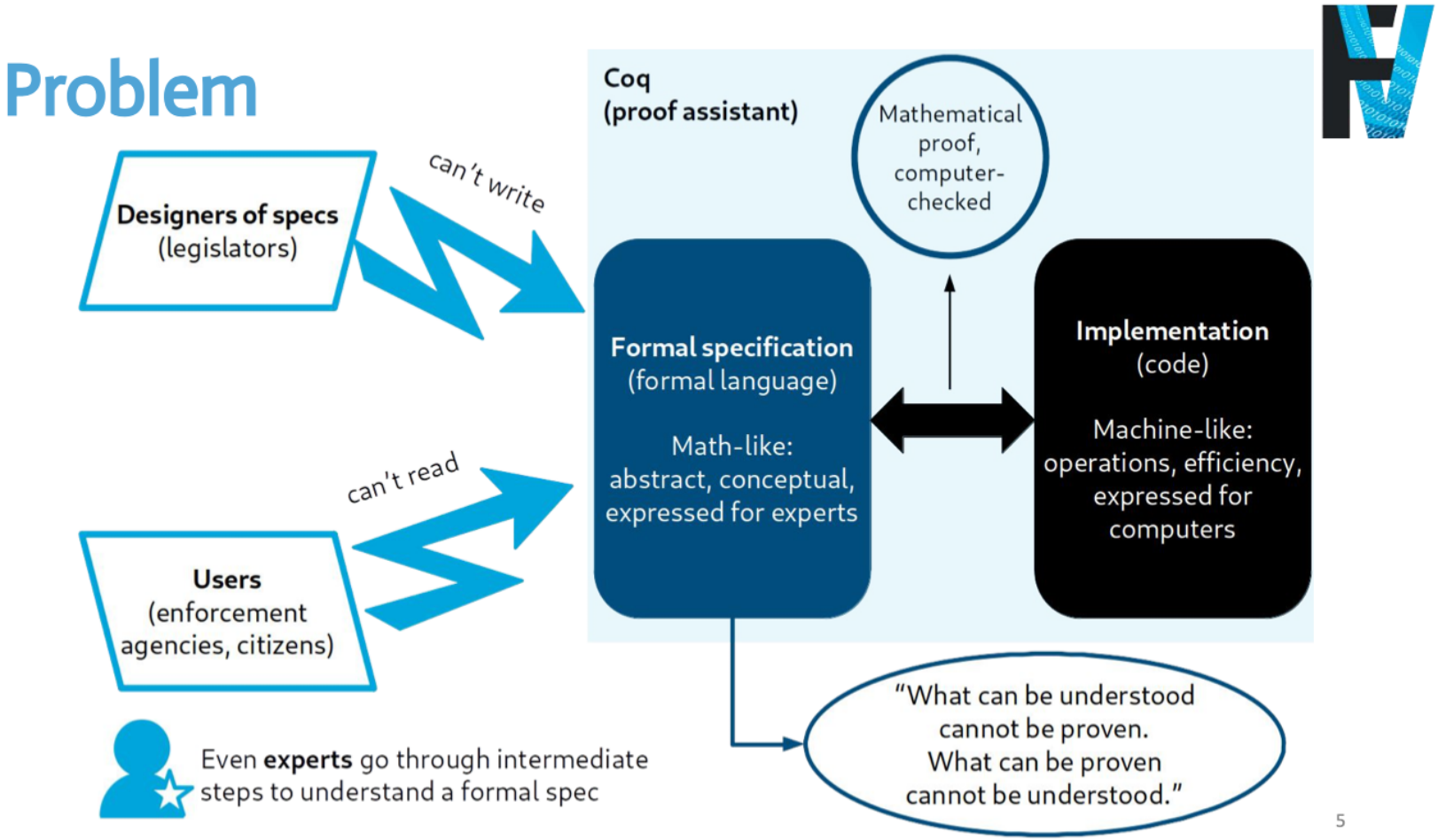
Further problems

- Develop explanatory certificates
- Zero-knowledge certificates for proprietary software!
- Standards and good practices for public certification.

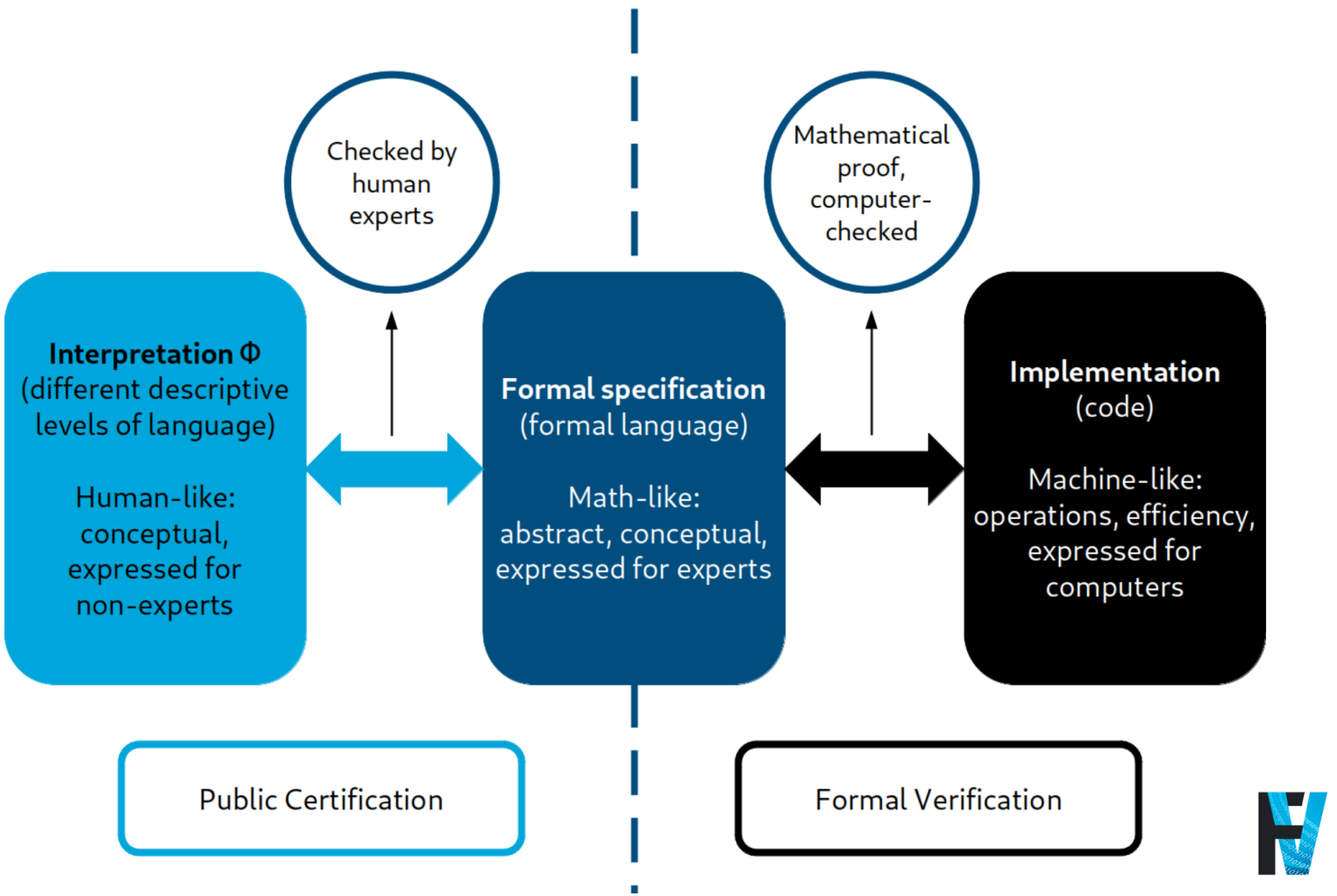
Further problems

- Develop explanatory certificates
- Zero-knowledge certificates for proprietary software!
- Standards and good practices for public certification.
- A formal subset of natural language (domain specific) to bridge the gap between formal specifications and technical specifications (public certification)

A central problem



Public certification versus formal verification

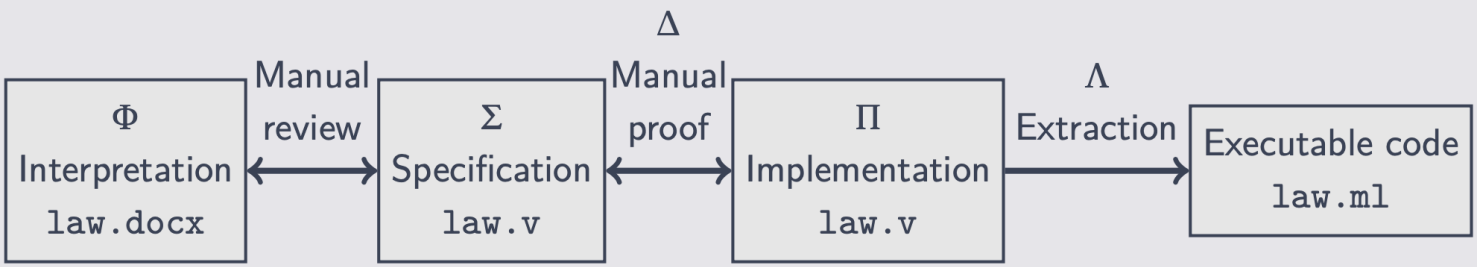


Another solution: Catala

Catala: A Shortcut For Legal Expert System Certification

The Usual Way to Produce Verified Software

Using Mireia Gonzáles Bedmar's conceptual framework from yesterday's presentation:



Catala's approach:

8

Slides Catala: Merigoux

❶ Catala: A Language Reviewable by lawyers

US Tax Code, Section 132, (c)(1) Qualified employee discount

The term "qualified employee discount" means any employee discount with respect to qualified property or services to the extent such discount does not exceed—
(A) in the case of property, the gross profit percentage of the price at which the property is being offered by the employer to customers

```
scope QualifiedEmployeeDiscount :
  definition qualified_employee_discount
    under condition is_property consequence equals
      if employee_discount >$ customer_price *$ gross_profit_percentage then
        customer_price *$ gross_profit_percentage
      else employee_discount
```

- FRET and FRETISH

- FRET and FRETISH
- Formal Requirement Elicitation Tool

- FRET and FRETISH
- Formal Requirement Elicitation Tool
- FRETISH is like Gallina

- FRET and FRETISH
- Formal Requirement Elicitation Tool
- FRETISH is like Gallina
- But now, it is a restricted part of natural language

- FRET and FRETISH
- Formal Requirement Elicitation Tool
- FRETISH is like Gallina
- But now, it is a restricted part of natural language
- With a precise and unambiguous (hence formal) meaning

Update Requirement

Requirement ID: AP-002A | Parent Requirement ID | Project: LM_requirements

Rationale and Comments

Rationale

Roll hold mode shall only be active, when the autopilot is engaged. Simultaneous engagement of several lateral modes shall not be possible.

Comments

Requirement Description

A requirement follows the sentence structure displayed below, where fields are optional unless indicated with "*". For information on a field format, click on its corresponding bubble.

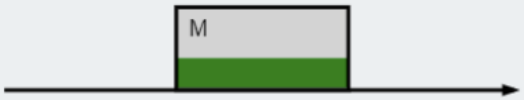
SCOPE | CONDITIONS | COMPONENT* | SHALL* | TIMING | RESPONSES*

in roll_hold mode RollAutopilot shall always satisfy autopilot_engaged & no_other_lateral_mode

SEMANTICS

Semantics

ENFORCED: in every interval where ***roll_hold*** holds. TRIGGER: first point in the interval. REQUIRES: for every trigger, RES must hold at all time points between (and including) the trigger and the end of the interval.



M = ***roll_hold***, Response = ***(autopilot_engaged & no_other_lateral_mode)***.

Diagram Semantics

Formalizations

Future Time LTL

```
(LAST V (roll_hold -> (autopilot_engaged &
no_other_lateral_mode)))
```

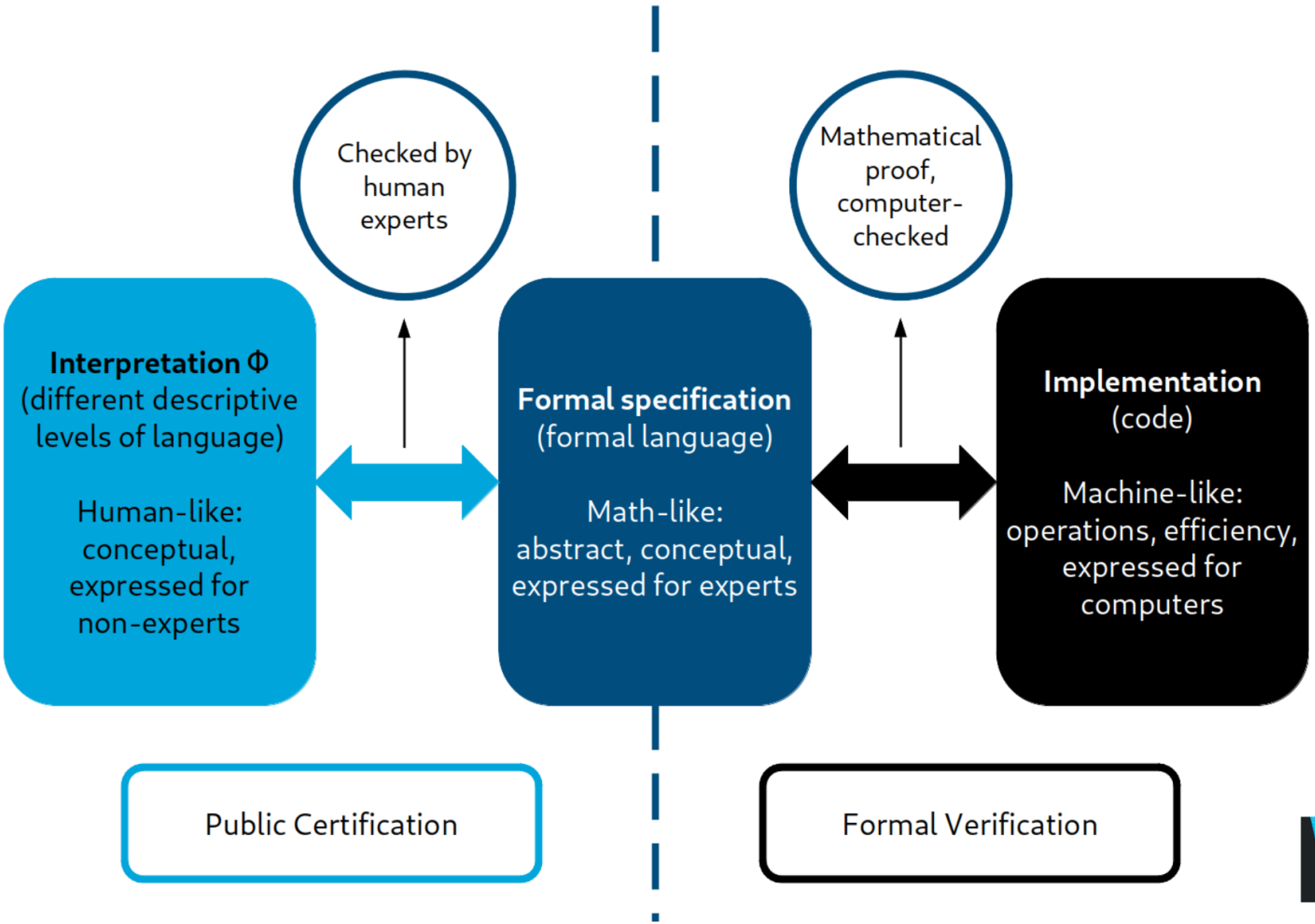
Target: ***RollAutopilot*** component.

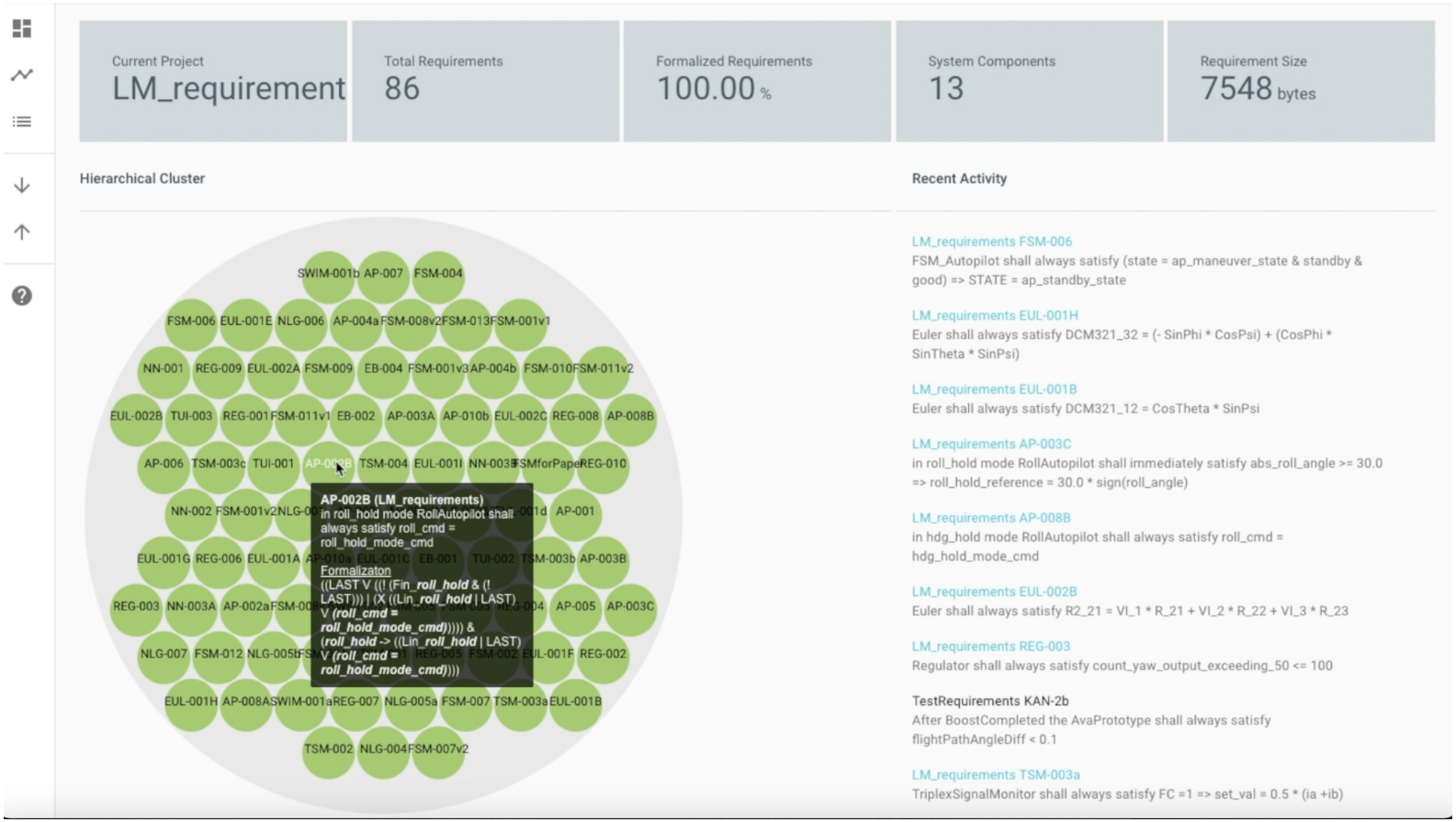
Past Time LTL

```
(H (roll_hold -> (autopilot_engaged &
no_other_lateral_mode)))
```

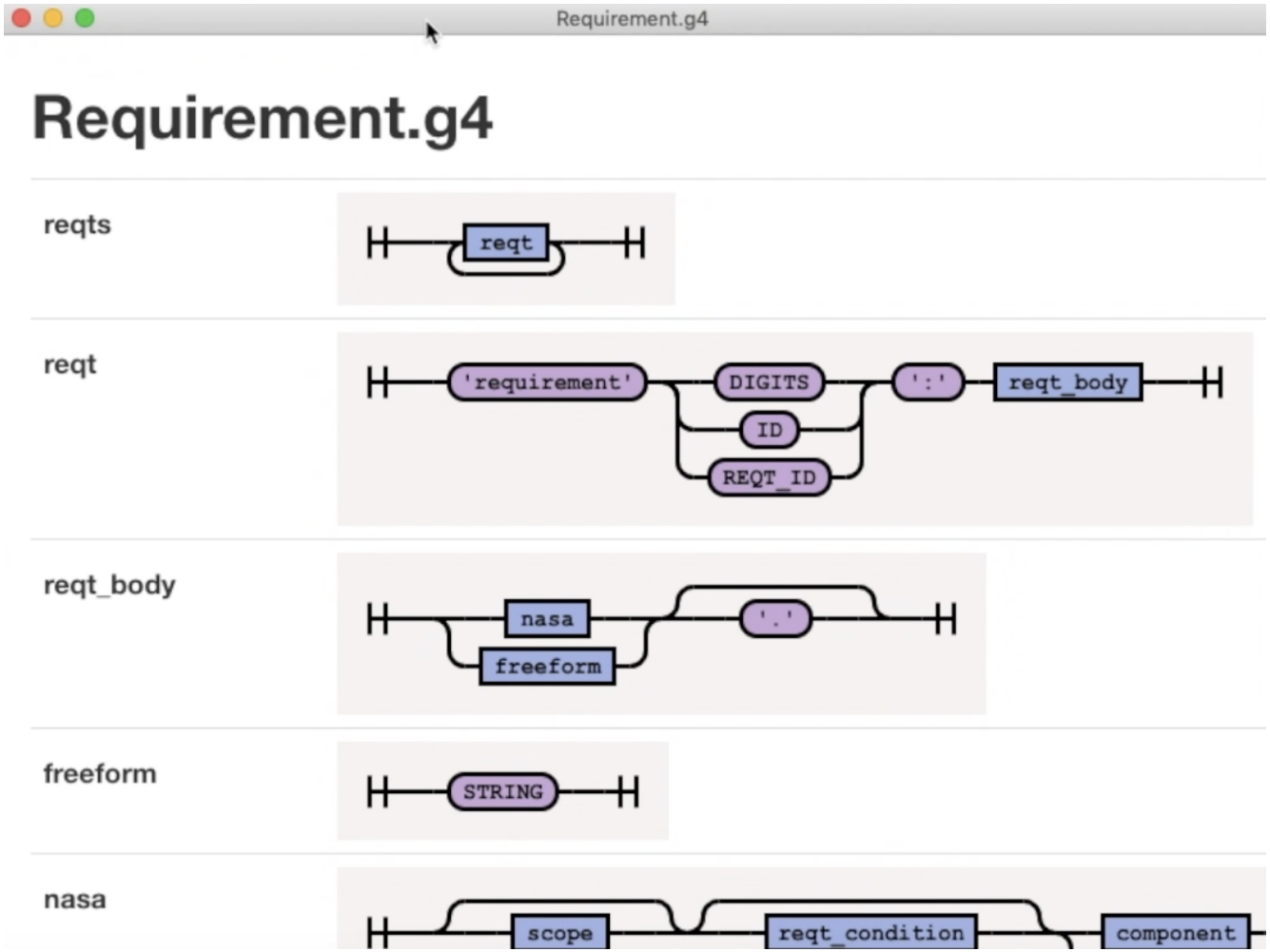
Target: ***RollAutopilot*** component.

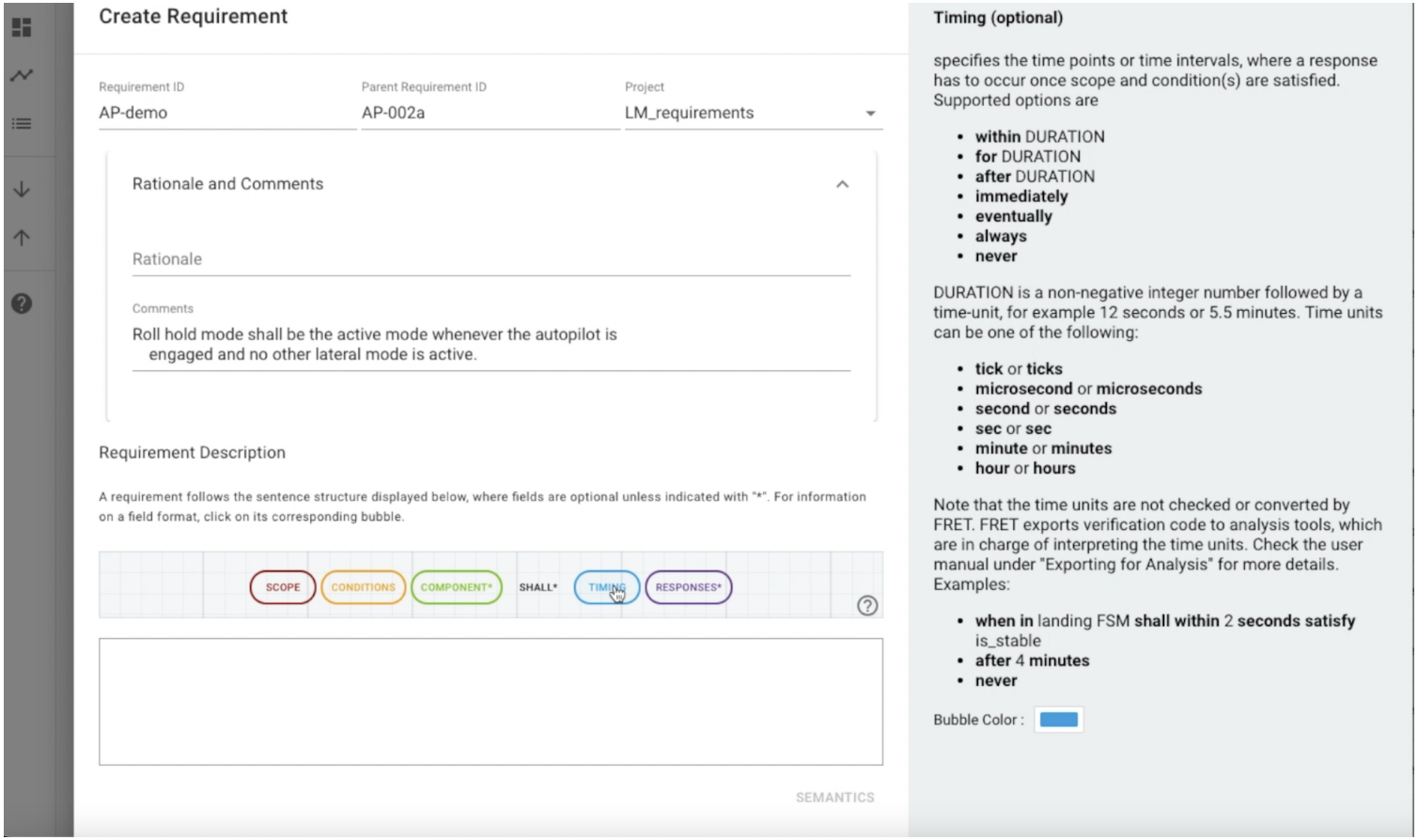
SIMULATE


Checked by human experts
Mathematical proof, computer-checked
Interpretation Φ
(different descriptive levels of language)
Human-like: conceptual, expressed for non-experts
Formal specification
(formal language)
Math-like: abstract, conceptual, expressed for experts
Implementation
(code)
Machine-like: operations, efficiency, expressed for computers
Public Certification
Formal Verification



Some requirements are children of others, etc.


Requirement.g4
Requirement.g4
reqts
reqt
reqt
'requirement'
DIGITS
ID
REQT_ID
':'
reqt_body
reqt_body
nasa
freeform
'.'
freeform
STRING
nasa
scope
reqt_condition
component



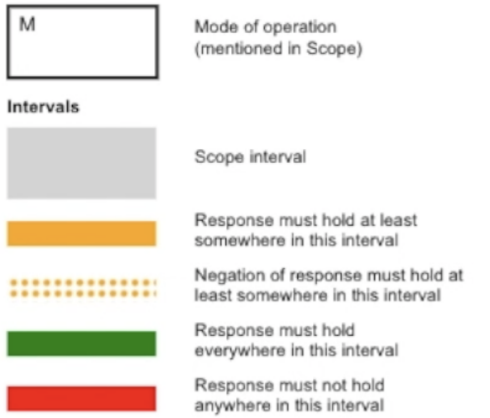
Only "Component", "Shall" and "Responses" are mandatory attributes, and each linguistic attribute comes with its explanation/specification. The order of the components is fixed.

Semantics

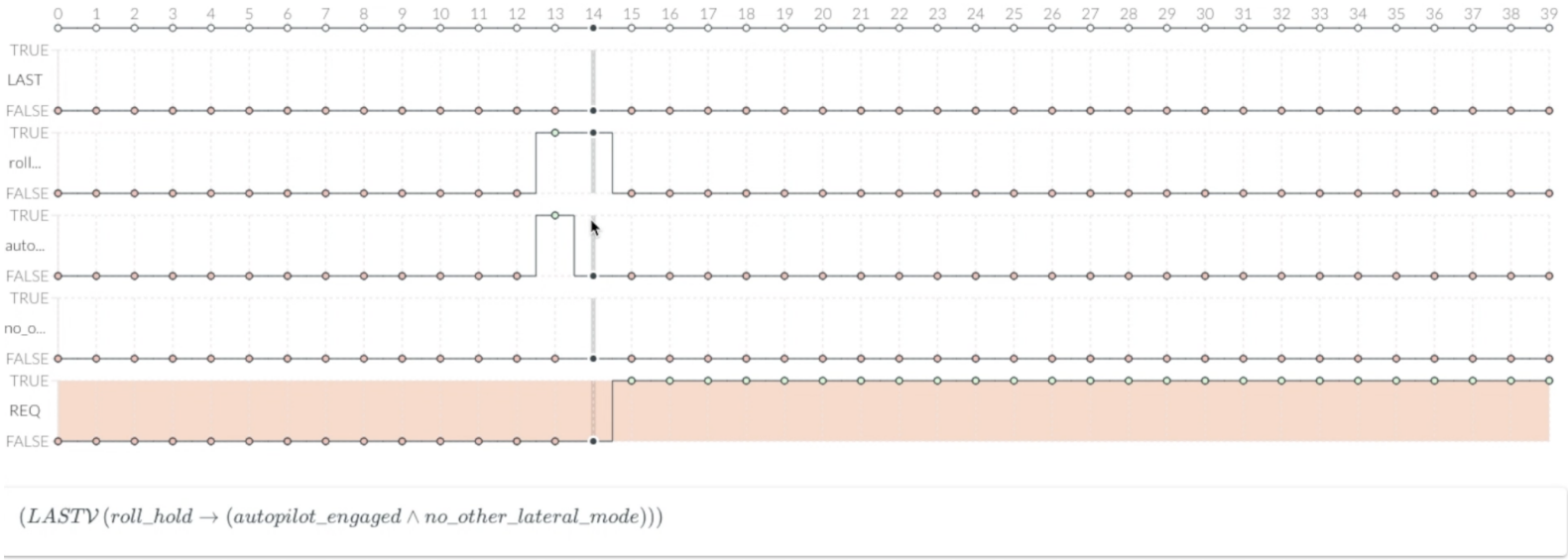
ENFORCED: in every interval where ***roll_hold*** holds. TRIGGER: first point in the interval. REQUIRES: for every trigger, RES must hold at all time points between (and including) the trigger and the end of the interval.

M = ***roll_hold***, Response = ***(autopilo_engaged & no_other_lateral_mode)***.

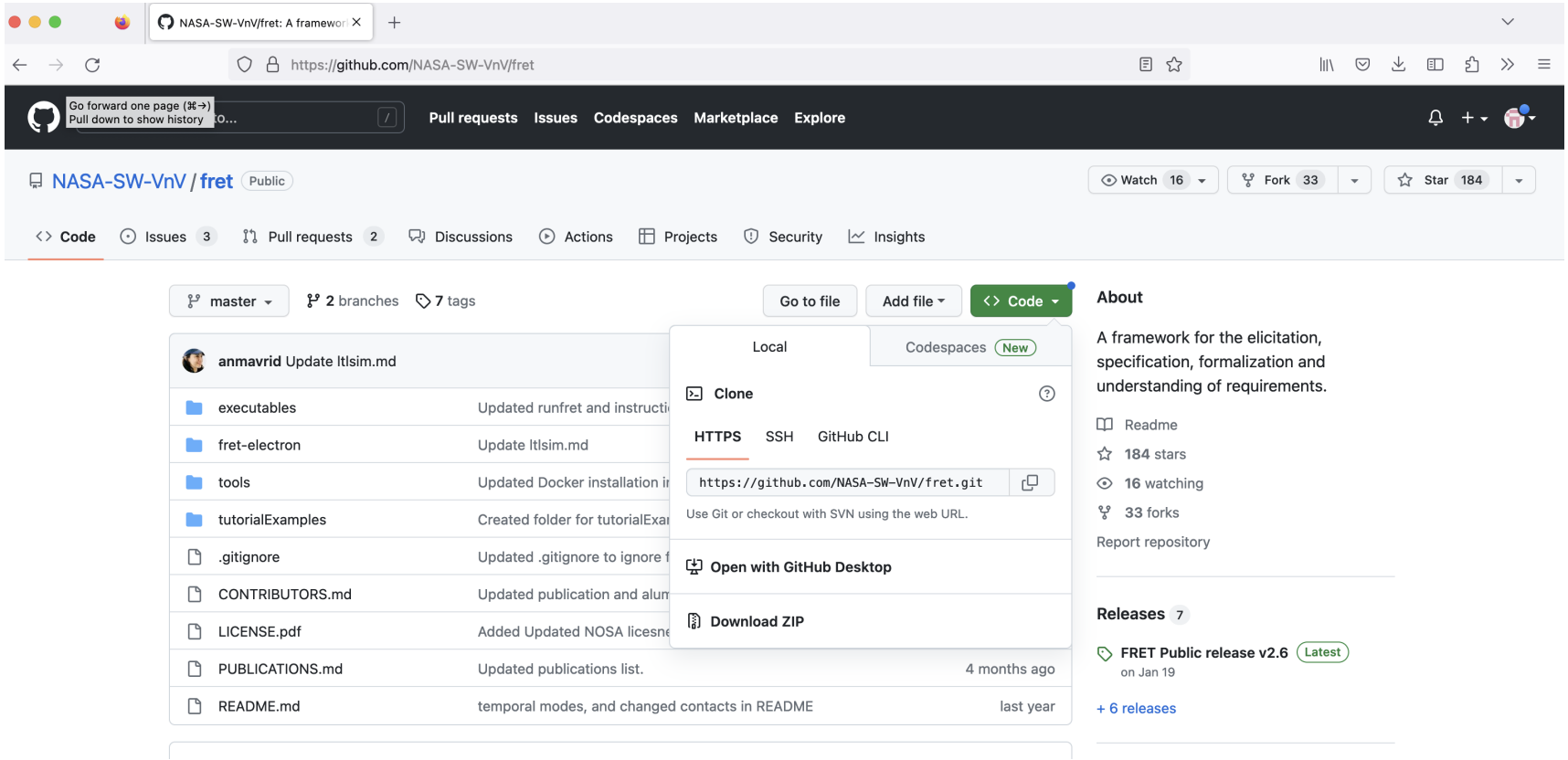
Diagram Semantics



Diagrams to explain the FRET Phrase



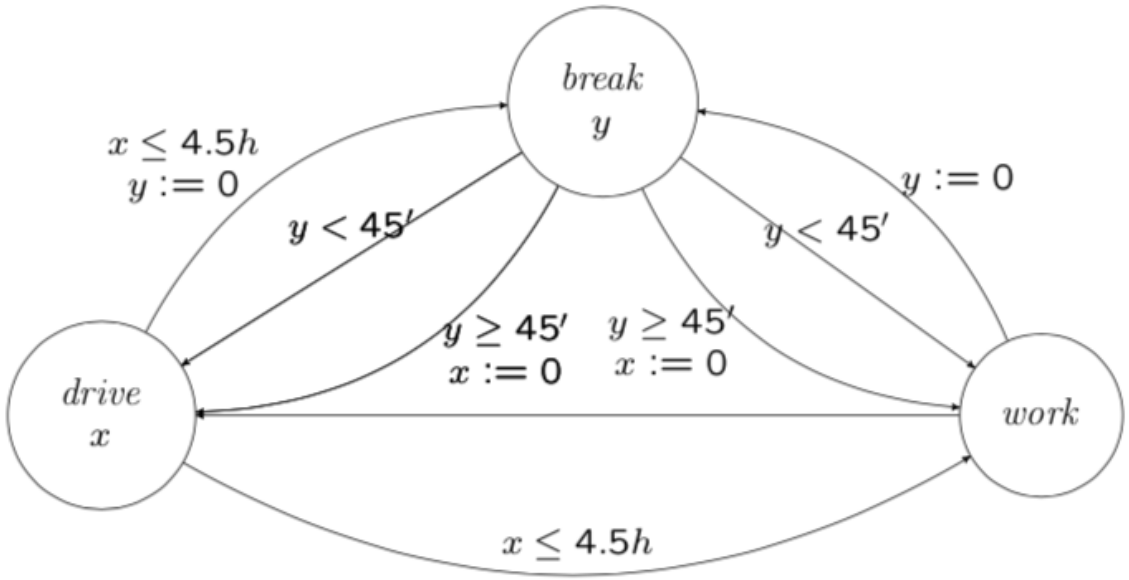
Users can test the behaviour of their FRET phrases



One can freely clone into it

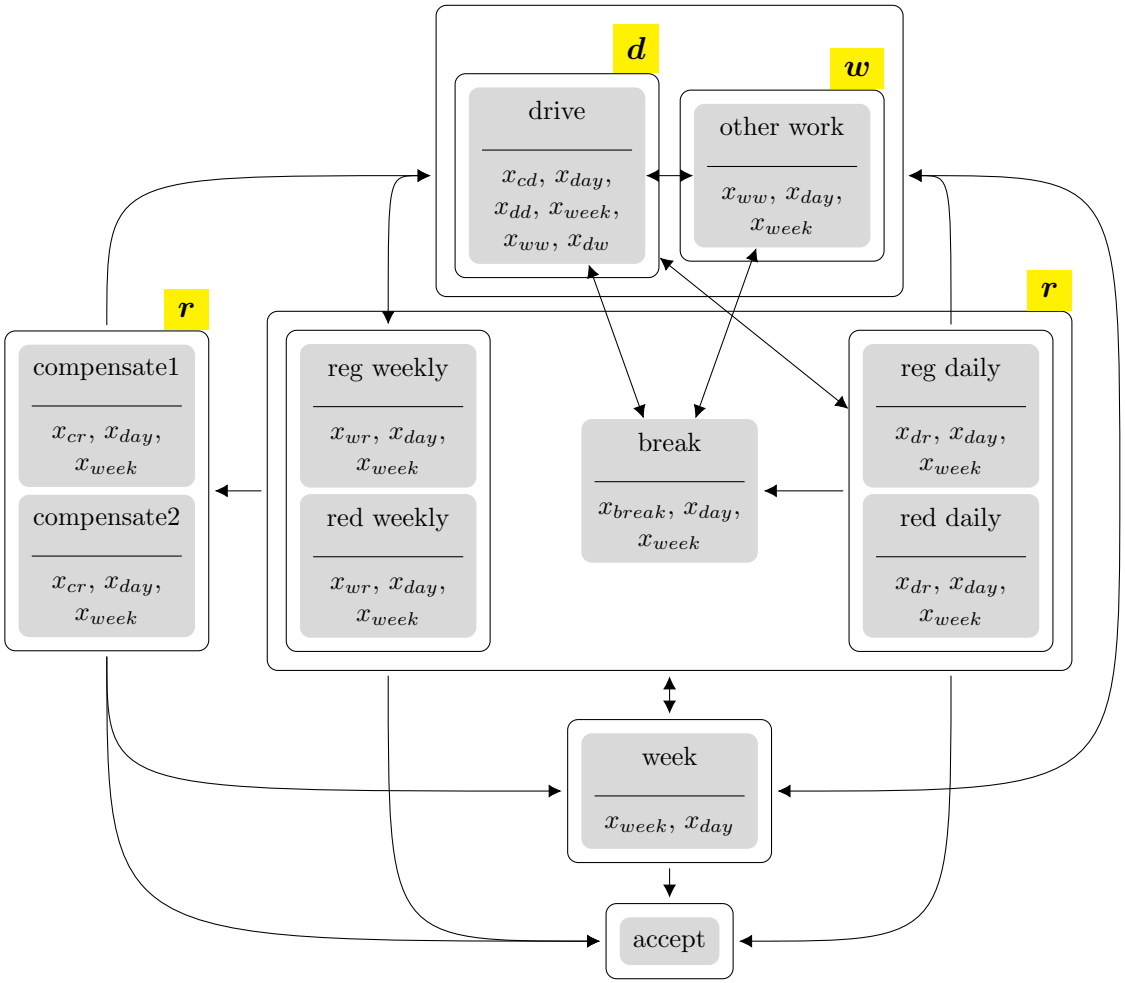
Example: continuous driving

Article 7 (1st part): After a driving period of four and a half hours a driver shall take an uninterrupted break of not less than 45 minutes,...



model checking images by Moritz Müller

Automaton that accepts exactly the legal words according to Reg. 561



12 states

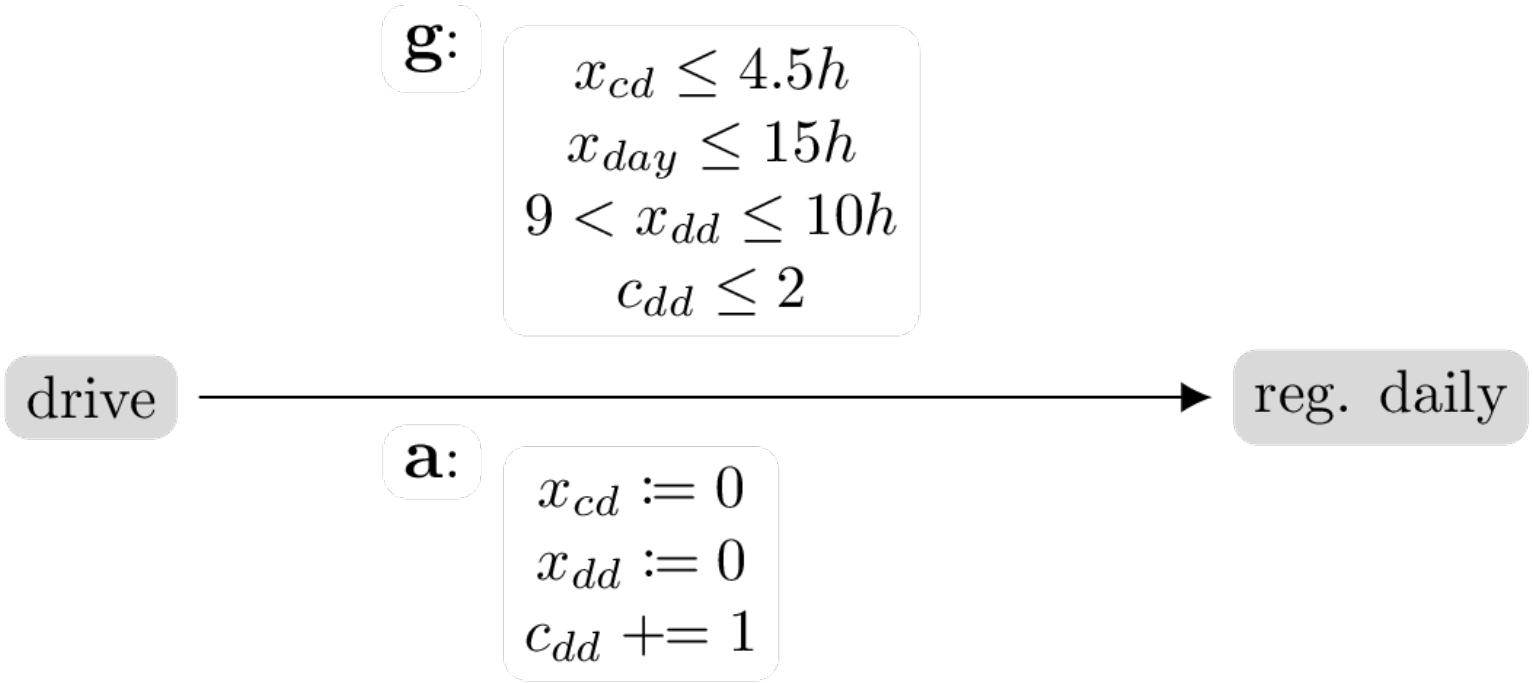
$>$ 100 transitions

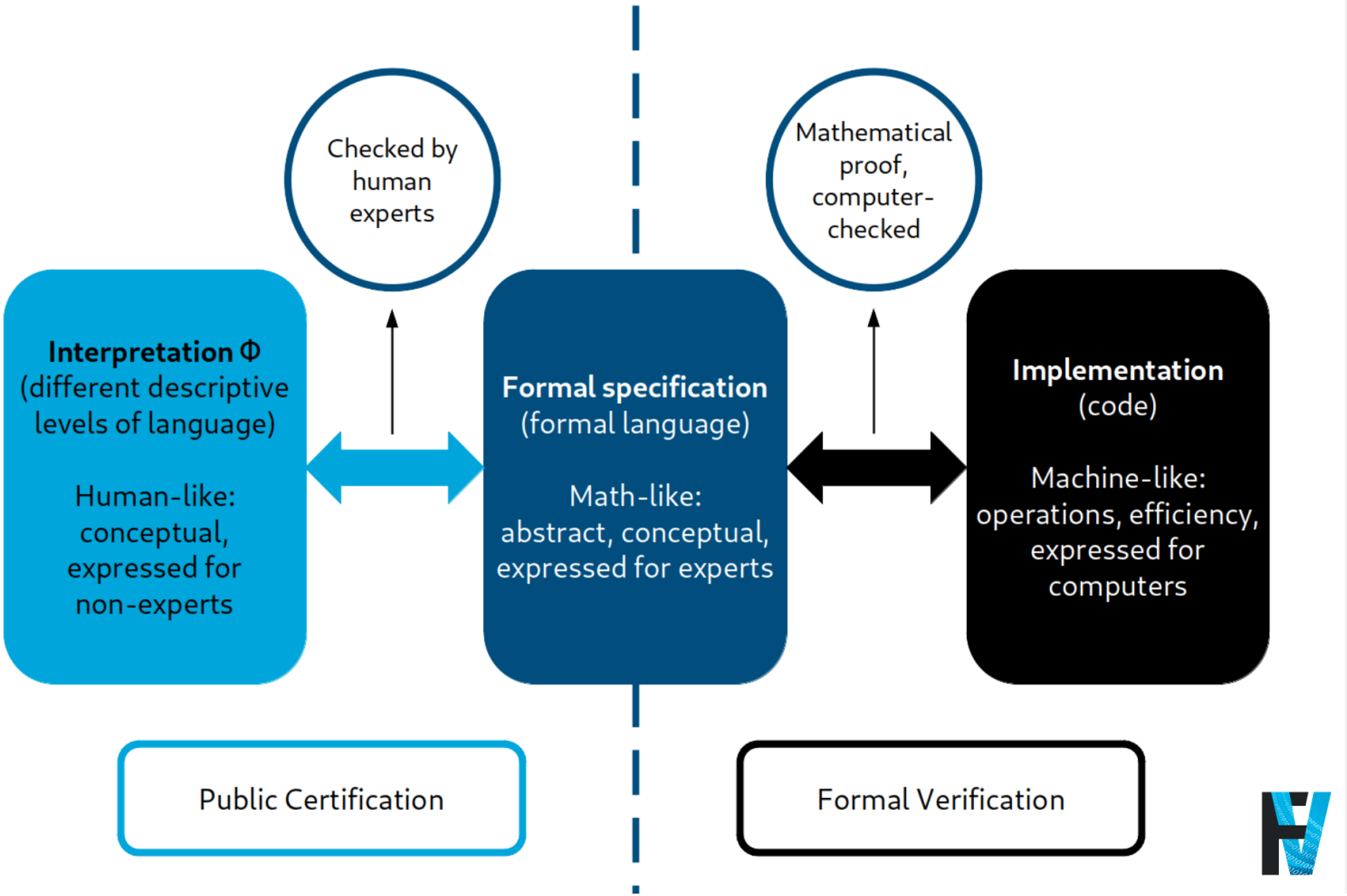
34 stopwatches

23 are nowhere active:

- *bits*
- *counters*
- *registers*

$>$ 100 transitions




Checked by human experts
Mathematical proof, computer-checked
Interpretation Φ
(different descriptive levels of language)
Human-like: conceptual, expressed for non-experts
Formal specification
(formal language)
Math-like: abstract, conceptual, expressed for experts
Implementation
(code)
Machine-like: operations, efficiency, expressed for computers
Public Certification
Formal Verification

Law and Code

- Law often uses essentially discretional powers when applied

Law and Code

- Law often uses essentially discretional powers when applied
- Hence, ambiguity is needed

Law and Code

- Law often uses essentially discretional powers when applied
- Hence, ambiguity is needed
- Any automated process and in particular, any automated process in the legal sector needs unambiguity

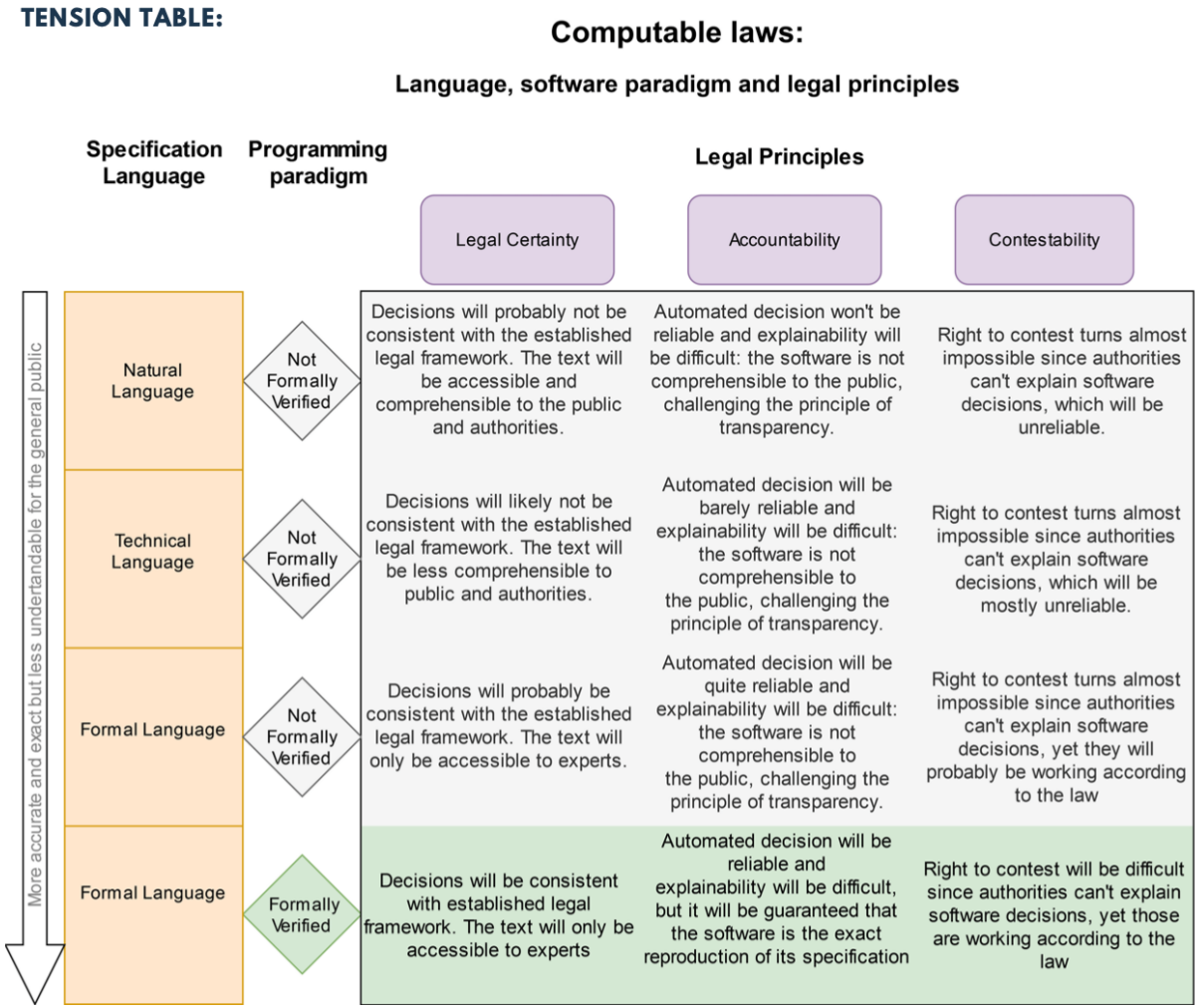
Law and Code

- Law often uses essentially discretional powers when applied
- Hence, ambiguity is needed
- Any automated process and in particular, any automated process in the legal sector needs unambiguity
- The programmer needs to disambiguate?

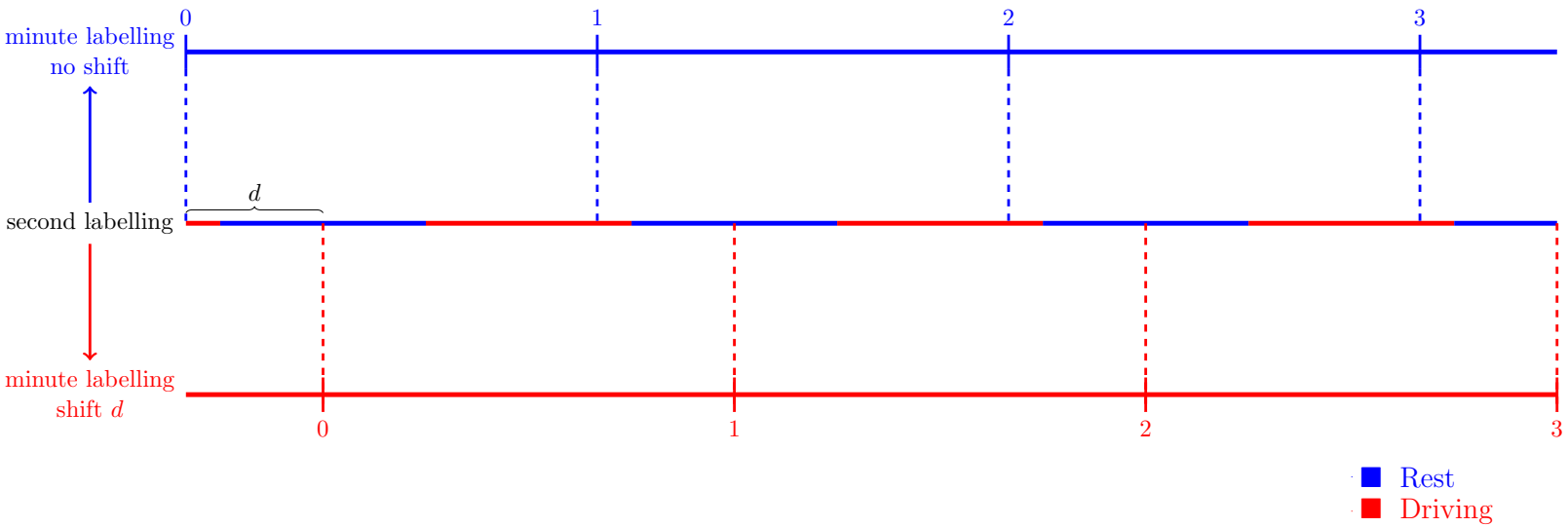
Law and Code

- Law often uses essentially discretional powers when applied
- Hence, ambiguity is needed
- Any automated process and in particular, any automated process in the legal sector needs unambiguity
- The programmer needs to disambiguate?
- Can (semi-)code be (part of) law?

Tension table



Hidden dynamics in formalisation



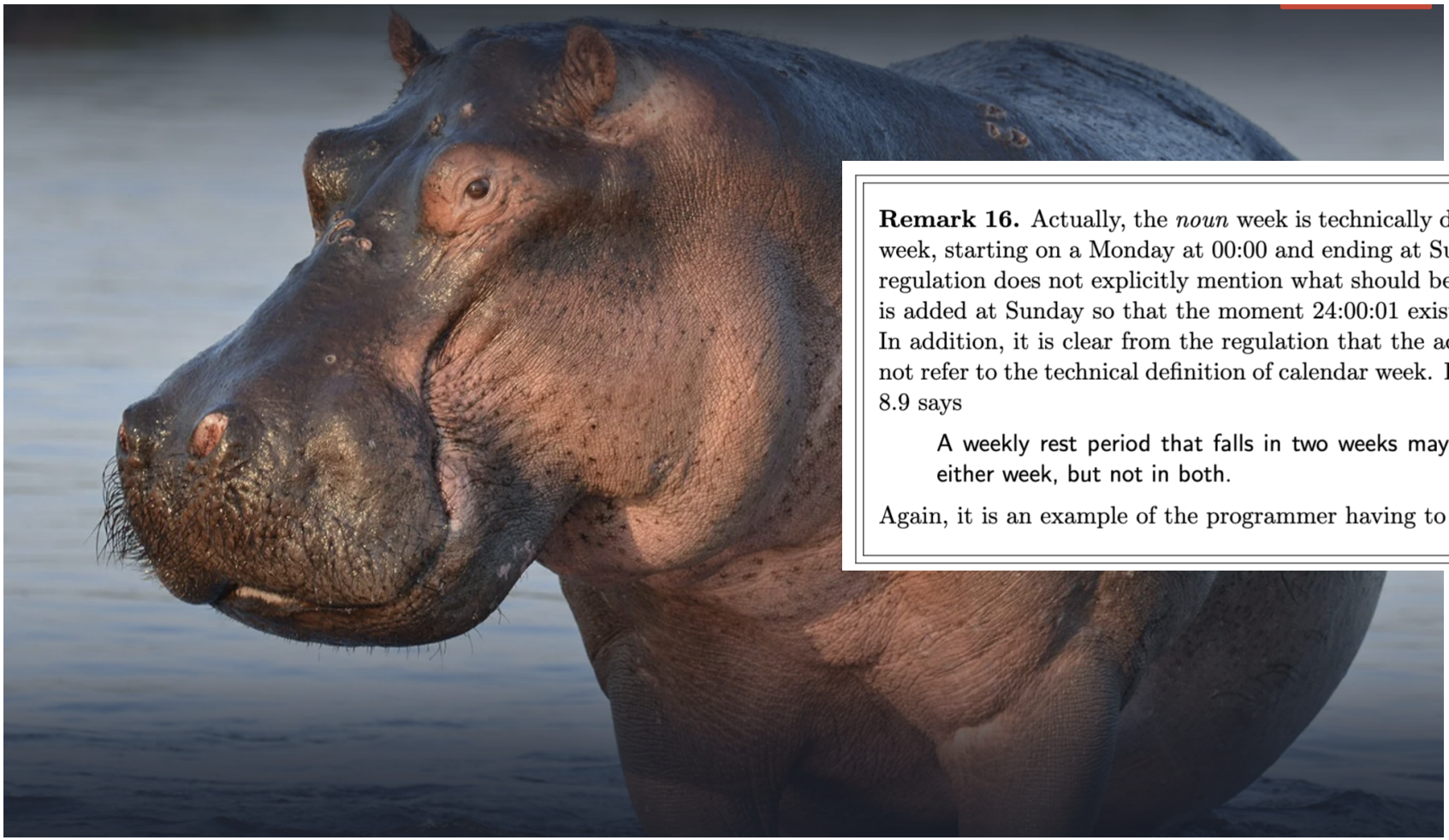
EU Regulation 2006/561 dictates how to interpret driving activities at minute level

EU Regulation 2016/799, requires labeling per second and prescribes to label per minute.

We proved that the labelling is not *shift-invariant*!

Germain Law-suit Continental Leap seconds.

Logical formal ontology



Remark 16. Actually, the *noun* week is technically defined as a calendar week, starting on a Monday at 00:00 and ending at Sunday at 24:00. The regulation does not explicitly mention what should be done a leap second is added at Sunday so that the moment 24:00:01 exists.
In addition, it is clear from the regulation that the adjective *weekly* does not refer to the technical definition of calendar week. For example, Article 8.9 says

> A weekly rest period that falls in two weeks may be counted in either week, but not in both.

Again, it is an example of the programmer having to disambiguate.

Some regulations regarding weekly rest periods

Regulation (EC) No 561/2006

§8.6. In any two consecutive weeks, a driver shall take at least:

- two regular weekly rest periods [of at least 45 hours], or
- one regular weekly rest period and one reduced weekly rest period of at least 24 hours. However, the reduction shall be compensated by an equivalent period of rest taken en bloc before the end of the third week following the week in question.

A weekly rest period shall start no later than at the end of six 24-hour periods from the end of the previous weekly rest period.

§8.9. A weekly rest period that falls in two weeks may be counted in either week, but not in both.

Let's break it down...

- Regular weekly rest: $\geq$ 45 hours

Let's break it down...

- Regular weekly rest: $\geq$ 45 hours
- Reduced weekly rest: $\geq$ 24 hours

Let's break it down...

- Regular weekly rest: $\geq$ 45 hours
- Reduced weekly rest: $\geq$ 24 hours
- Each rest period is assigned to only one week it intersects

Let's break it down...

- Regular weekly rest: $\geq$ 45 hours
- Reduced weekly rest: $\geq$ 24 hours
- Each rest period is assigned to only one week it intersects
- Every week must have a regular or reduced weekly rest

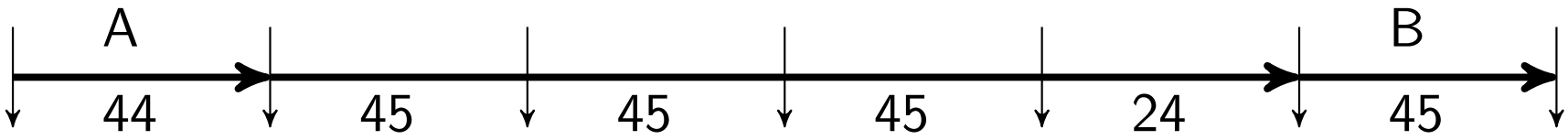
Let's break it down...

- Regular weekly rest: $\geq$ 45 hours
- Reduced weekly rest: $\geq$ 24 hours
- Each rest period is assigned to only one week it intersects
- Every week must have a regular or reduced weekly rest
- Every other week must have a full weekly rest

Let's break it down...

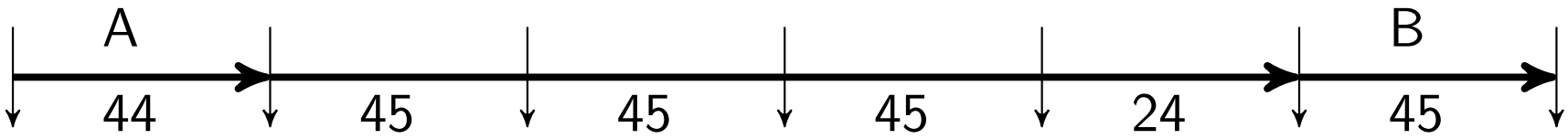
- Regular weekly rest: $\geq$ 45 hours
- Reduced weekly rest: $\geq$ 24 hours
- Each rest period is assigned to only one week it intersects
- Every week must have a regular or reduced weekly rest
- Every other week must have a full weekly rest
- Any reduced rest must be compensated by a continuous block in the following three weeks

Non-locality of compensations

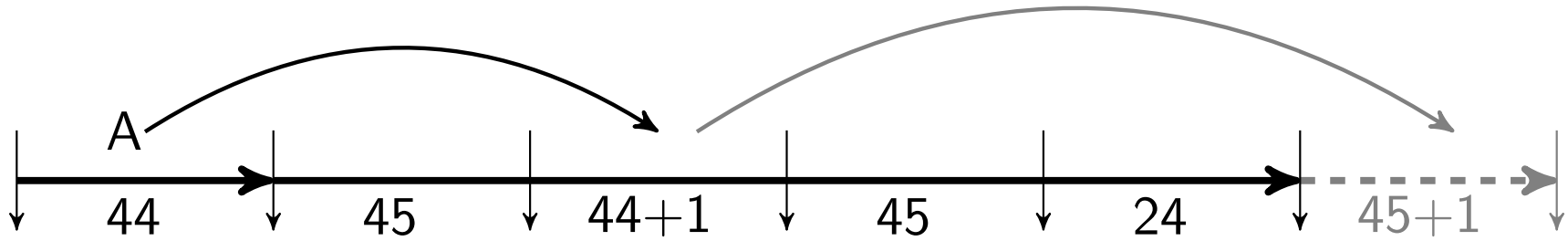


Illegal

Non-locality of compensations

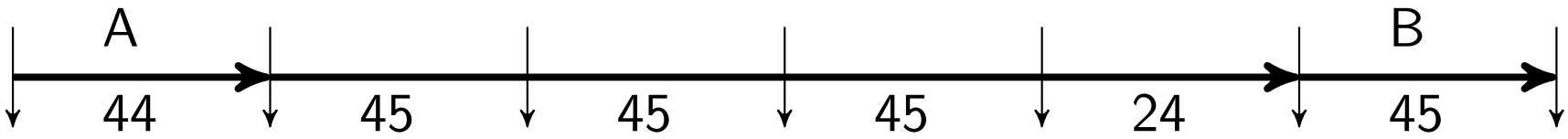


Illegal

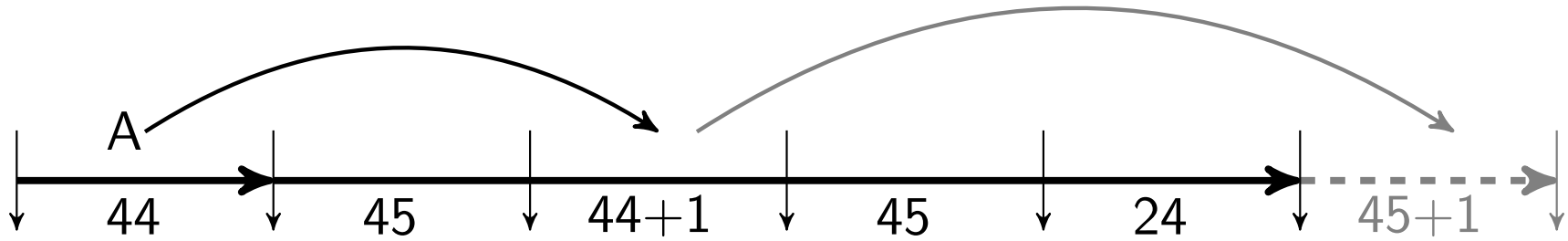


Legal

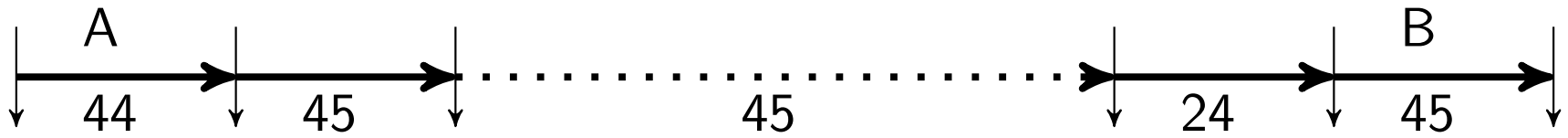
Non-locality of compensations



Illegal



Legal



This can be iterated indefinitely: non-locality

Under-determined daily driving allocation

Article 6.1: The daily driving time shall not exceed nine hours. However, the daily driving time may be extended to at most 10 hours not more than twice during the week.

Remark 16. We recall that daily driving times are periods that are delimited by daily rest periods and as such a single daily driving time can very well be spread over two different calendar days. The week however is defined as calendar week starting at Monday 00:00. Now, what happens if a driver has a daily driving period of 10 hours starting on a Sunday and ending on a Monday? This is an extended daily driving time. Should it be counted to the week staring on that Monday or to the week ending on that Sunday? The law seems underspecified here. We shall see that our model will disambiguate by assigning it to the week that starts on Monday. Various tachograph readers make differenct choices and, for example, the sofware *Police Controller* has an option to fix your choices or to choose the distribution as to minimize the fine.

Should the programmer/modeler disambiguate underspecification?

Non-locality also enters the scene through extended daily driving periods.

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Not allowing ambiguity

Formal framework needed to express unambiguity

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Not allowing ambiguity

Formal framework needed to express unambiguity

Expressivity

Legal articles should be expressible in the language

our test case: Regulation 561

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Not allowing ambiguity

Formal framework needed to express unambiguity

Expressivity

Legal articles should be expressible in the language

our test case: Regulation 561

Naturality

readable sentences

sufficiently succinct

Limited language imply long sentences

Example `Article 6.2: The weekly driving time shall not exceed 56 hours`

In LTL formalized over words of length $1w$: disjunction of

$$\bigwedge_{d \leq D} \left(\bigwedge_{r_d \leq i < \ell_{d+1}} \bigcirc^i \neg d \wedge \bigwedge_{\ell_d \leq i < r_d} \bigcirc^i d \right)$$

for all $D \leq 1w$ and
all $r_0 := 0 \leq \ell_1 < r_1 < \cdots < \ell_D < r_D < \ell_{D+1} := 1w$ with
$\sum_{1 \leq j \leq D} (r_j - \ell_j) \leq 56h$
This has $> \binom{7 \cdot 24 \cdot 60}{56 \cdot 60} > 10^{2784}$ many disjuncts.

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Not allowing ambiguity

Formal framework needed to express unambiguity

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Not allowing ambiguity

Formal framework needed to express unambiguity

Expressivity

Legal articles should be expressible in the language

our test case: Regulation 561

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Not allowing ambiguity

Formal framework needed to express unambiguity

Expressivity

Legal articles should be expressible in the language

our test case: Regulation 561

Naturality

readable sentences sufficiently succinct

Requirements on languages for algorithmic law

Ontology

That naturally bridges (physical) observables and natural language concepts

And fares well with the other requirements

Not allowing ambiguity

Formal framework needed to express unambiguity

Expressivity

Legal articles should be expressible in the language

our test case: Regulation 561

Naturality

readable sentences sufficiently succinct

Tractability

sufficiently fast (model-)checkers

Thanks