

Wybe

**Formal verification
for the masses**

Testing vs Proving

$2 + 2$		$x + x$
$= \{ \text{addition rules} \}$		$= \{ \text{basic algebra} \}$
4		$2 * x$

Rules for transforming programs

```
if ¬true then A else B
= { applying ¬ }
  if false then A else B
= { semantics of the if statement }
  B
```

Smart contracts

- It's important to sleep at night if you're responsible for a smart contract holding valuable assets
- Formal verification can and has been applied successfully to make more secure smart contracts

Formal Land

- Ethereum, Tezos, Sui, Aleph Zero



Why it's not used more?

- Dependent types
- Niche platforms and languages
- PhD level knowledge



```
1 +1-length : ∀ {ℓ}{A : Set ℓ}(h : A)(t :  $\mathbb{L}$  A) → length (h :: t) ≡ length t + 1
2 +1-length h [] = refl
3 +1-length h t = refl
```

A bridge

- F# and .NET
- Simpler yet powerful language
- A work in progress

```
1 let ``double negation`` =  
2   proof () {  
3     Theorem("double negation", !(!x) == x)  
4     withLaws { ``true theorem`` }  
5     !(!x) == x  
6     ``==`` { ``GS 3.11`` }  
7     !x == !x  
8     ``==`` { ``== ident`` }  
9     True  
10  }
```

Reactions



Luis

@lamg_dev

Promote



#fsharp now we have a proof checker embedded in F#'s computation expressions WIP

[Traducir con DeepL](#)

```
1 let ``double negation`` =  
2   proof () {  
3     Theorem("double negation", !(!x) == x)  
4     WithLaws [ trueTheorem ]  
5     !(!x) == x  
6     ``==`` { ``GS 3.11`` }  
7     !x == !x  
8     ``==`` { ``== ident`` }  
9     True  
10  }
```

1:36 PM · Apr 6, 2025 · 3,092 Views

View post engagements



1



13



73



9



Reactions

Reactions

All 25

19

2

2

2

Guillaume Claret · 1st
Security Researcher @Formal Land - Formal verification for Web3
- Solidity, Rust, OCaml

Jonas Lara · 1st
Full Stack Developer





















Vinícius Gajo Marques Oliveira · 1st
Software & Infrastructure Engineer | Open-source contributor |
DevOps/DevSecOps | Mechatronics Engineer

Don Syme · 1st
Principal Researcher, Visiting Professor

John A. · 2nd
Generative AI · Quantum Computing · Functional Programming ·
Cloud Computing · Scientific Computing · Principal Software
Engineer @ Microsoft

Marius Fersigan · 2nd

Reactions

 simontreanor Fund Ourselves Follow	 bdaniel7 Bucharest, Romania Follow	 Reenuay Joined on Aug 16, 2016 Follow
 esneko Latvia Follow	 jasonmc Brooklyn, New York Follow	 jhwohlgemuth Oak Ridge National Laboratory Follow
 sgoguen Joined on Mar 5, 2009 Follow	 jonas1ara Mexico Unfollow	 alsnrbernardo Joined on May 26, 2020 Follow
 clarus Formal Land Unfollow	 bretanac93 @sumup Unfollow	 sheepia Japan Follow
 Numpsy UK Follow	 shubhamkumar13 India Follow	 NatElkins Joined on May 30, 2012 Follow
 dawedawe Freelance Unfollow	 DejanMilicic Serbia Unfollow	 Aaronontheweb Petabridge, LLC Follow
 Lanayx Joined on Jan 21, 2013 Unfollow	 jcmrva Denver Follow	

10 years ago

“Llegué a este estilo por mi preocupación de escribir los programas bien, pues los errores se me escapaban al escribir un programa y notaba que estaban ahí cuando los ejecutaba”, afirma.



Luis Ángel Méndez, un bicho raro entre sus colegas. Foto: Eduardo González Martínez

El método Dijkstra se opone a la práctica imperante de escribir un programa primero y someterlo después a una corrección matemática. Con su propuesta, aunque más demorada, se obtiene un programa listo para ejecutarse, correcto desde su construcción.

Questions?

- <https://github.com/lamg>