

Your CRA Plan Is Already Broken: Only Trustable Software Can Save It





John Ellis

ROLE

President, Head of Product, Codethink

TENURE

Joined Codethink in 2023

PAST ENGAGEMENTS

Open Mobility Foundation, Lacuna, Ford Motor Company, Ellis & Associates, [SPDX](#), Linux Foundation, Motorola

EDUCATION

MS Computer Science Illinois Institute of Technology

MBA University of Notre Dame

Some of Our Customers

arm

 Elektrobit

 Micron



Bloomberg



Mercedes-Benz



NUTANIX



 BOSCH



Panasonic





When Software Breaks the World: Real Failures Across Industries

Automotive: Software glitches now account for up to 50% of all vehicle recalls—impacts span from infotainment to ADAS and powertrain systems ([ElectRay Technologies](#)).

Healthcare: In 2016, St. Jude pacemakers had firmware vulnerabilities—FDA issued a recall in 2017 ([Embedded Artistry](#)).

Global: The estimated total global damage from the CrowdStrike outage is at least \$10 billion worldwide, with Fortune 500 companies alone facing approximately \$5.4 billion in losses ([Wikipedia](#)).



More Examples:

Qantas Flight 72 experienced uncommanded pitch-downs due to a software design limitation triggered by faulty sensor data ([Wikipedia](#)).

Ariane 5 Flight V88 failed in 1996 due to a reused piece of code from Ariane 4 that caused a fatal overflow—loss: >US \$370 million ([Wikipedia](#)).

These are not isolated, software failures can ripple through lives, safety, and reputations.

Why Do You **Trust** Software?



We can offer software as Trustable when we provide **evidence** to support all of these claims...

1. Provenance

Know where it comes from, who is responsible, and have confidence in them

2. Construction

Know how to build it - **reproducibly** - from source

3. Changes

Upgrade it and be confident it will not break or regress

4. Expectations

Know what it must do, and what it must not do

5. Results

It does what it must do, and does not do what it must not do

6. Confidence

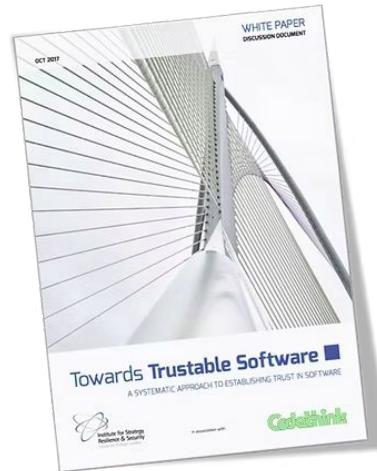
Measure and declare our confidence that it will not cause harm

What is the Trustable Software Framework (TSF)?

The Trustable Software guiding principles were published in 2017¹

The TSF project builds on these principles to support the management of knowledge (reasoning) about software and software projects, including:

- Requirements
- Documentation
- Success metrics
- Observability
- Analysis outcomes
- Test result interpretation
- Compliance



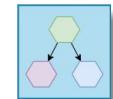
ECLIPSE
FOUNDATION

¹ See the *Towards Trustable Software* white paper available from <https://www.trustablessoftware.com/>

What is a framework?

As a framework, TSF needs to provide the following:

- **Objectives**: defining what is important, and what it is trying to accomplish.
- **Model**: a simplified description of a more complex system or idea, focusing on specific elements or relationships.
- **Methodology**: a system of methods used for a particular activity, which may use models
- **Tools**: to help apply these methods, which are adaptable to users' needs and project-specific objectives.
- **Documentation**: to provide guidance and examples of how to use the framework.



Design objectives for TSF

- Lightweight continuous compliance framework
 - Evidence-driven approach for collection and management of project metadata
 - Metadata is maintained alongside software and related artifacts
 - Designed for continuous evaluation of metadata using project-defined criteria
- Focuses on objectives rather than processes
 - Organise your objectives and evidence related to them
 - Includes a set of core objectives for Trustable Software to use as a basis
 - Supports linking to external objectives (e.g. defined by standards) where desired
- Workflow-agnostic
 - Designed as (and for) FOSS*, but intended for use with any software
 - Only prerequisite is a continuous integration (CI) workflow built around git
 - Tools are provided, but not required, to support the model and methodology



TSF Model

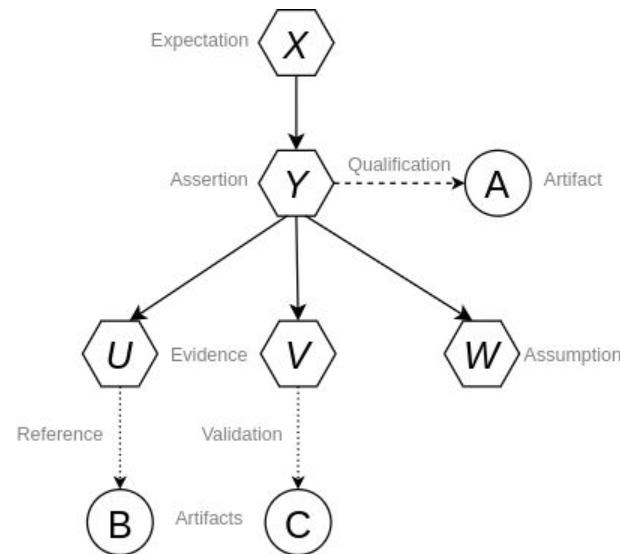
Model for decomposing requirements and organising project metadata into a traceable network of coherent statements

Each statement may be interpreted as a **request** or a **claim**, describing:

- an **objective** to be achieved, *and/or*
- **how** it is achieved for a given **context**

Statements may be linked to **artifacts** (inputs or outputs of engineering processes), which either:

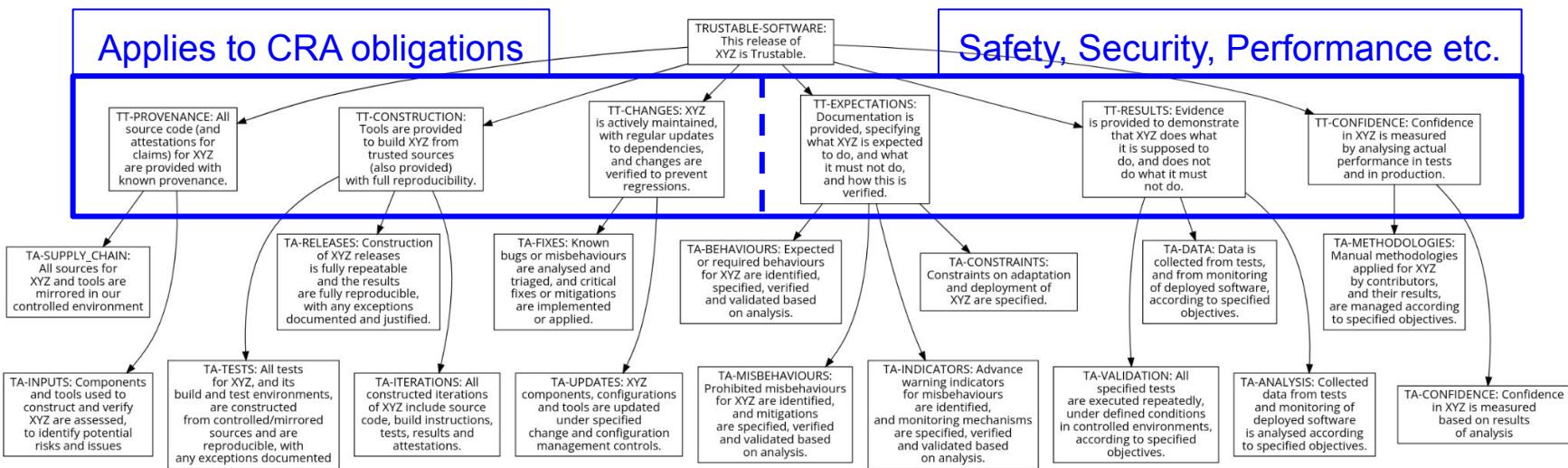
- **qualify** the stated objective, *or*
- provide **evidence** that it has been achieved

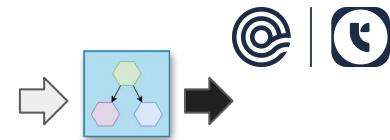


Trustable objectives

What evidence is needed for software to be considered ‘trustable’?

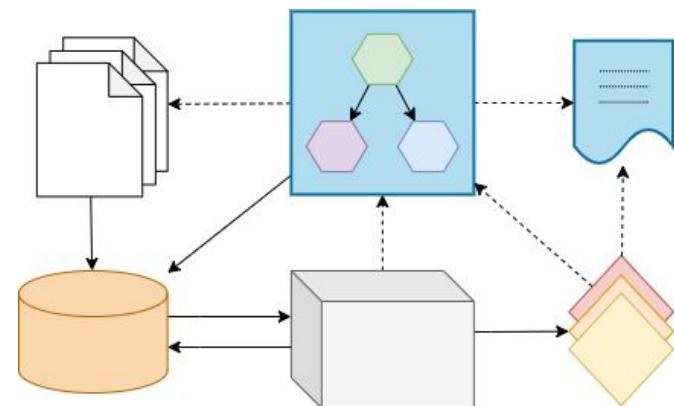
- Common set of ‘baseline’ objectives, to be extended with project-specific ones
- Based on established best practices and experience
- See the TSF documentation for more details





TSF Methodology

- Apply in-context as much as possible for the project, extending for components where appropriate
- Document project-specific objectives, reasoning for your software and its development processes
- Link to requirements and evidence managed in your project, or in other systems and contexts
- Map your claims and evidence to the Trustable Objectives
- Map Trustable and project-specific objectives and evidence to external requirements (e.g. standards)





TSF Tools

- Manage TSF graphs and produce reports
- Identify impacts of a change and when existing items need to be (re-)reviewed
- Integrate with automated testing using validators to see 'live' results
- Use confidence scores to track progress towards objectives and give feedback on gaps or work required

Trustable Software Framework

Trustable Reports Trustable Tools Applications

Trustable Compliance Report

Item status guide

Each item in a Trustable Graph is scored with a number between 0 and 1, representing confidence in a given Statement, with larger numbers corresponding to both a numerical score and the colormap below:



The status of an item and its links also affect the score.

Unreviewed items are indicated by a cross in the status column. The score is zero.

Suspect links are indicated by a cross in the status column. The contribution of the child is always zero, regardless of the child's own score.

Compliance for TA

Item	Summary
TA-ANALYSIS	Collected data from tests and monitoring of deployed software analysed according to specified objectives.
TA-BEHAVIOURS	Expected or required behaviours for XYZ are identified, specified, verified and validated based on analysis.

Compliance for TRUSTABLE

Item	Summary	Score
TRUSTABLE-SOFTWARE	This release of CTRL is Trustable.	0.62

Compliance for TT

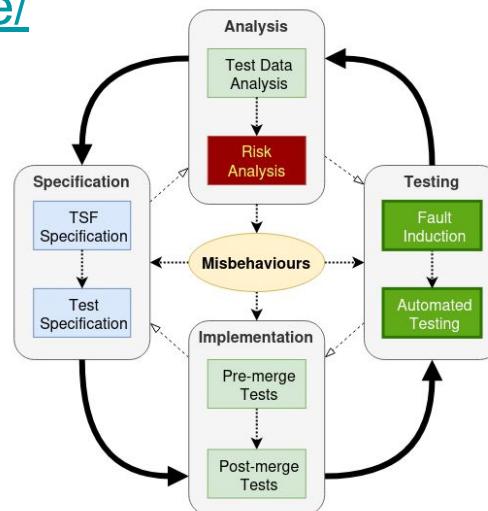
Item	Summary	Score
TT-PROVENANCE	All source code (and attestations for claims) for CTRL are provided with known provenance.	0.58
TT-CONSTRUCTION	Tools are provided to build CTRL from trusted sources (also provided) with full reproducibility.	0.76
TT-CHANGES	CTRL is actively maintained, with regular updates to dependencies, and changes are verified to prevent regressions.	0.70
TT-EXPECTATIONS	Documentation is provided, specifying what CTRL is expected to do, and what it must not do, and how this is verified.	0.14
TT-RESULTS	Evidence is provided to demonstrate that CTRL does what it is supposed to do, and does not do what it must not do.	0.73
TT-CONFIDENCE	Confidence in CTRL is measured by analysing actual performance in tests and in production.	0.82

TSF Documentation

- The TSF project includes extensive documentation, explaining the background to Trustable and the various aspects of the framework.

<https://codethinklabs.gitlab.io/trustable/trustable/>

- It also includes specific applications of Trustable, such as the [RAFIA approach](#)¹, and guidance for using TSF to manage safety requirements,
- It is authored in markdown and published using [Material for mkdocs](#)², and is under active development alongside the TSF tools.



¹ RAFIA: <https://codethinklabs.gitlab.io/trustable/trustable/applications/rafia/index.html>

² Material for mkdocs: <https://squidfunk.github.io/mkdocs-material/>

Trillions of US Dollars

\$10–\$19T cumulative (1995–2025) based on CSIS/McAfee \$~1T/yr anchor and conservative earlier-year assumptions
with the caveat that no audited sum exists and that overlaps in sub-series preclude precise aggregation.

Why Do You **Trust** Software?



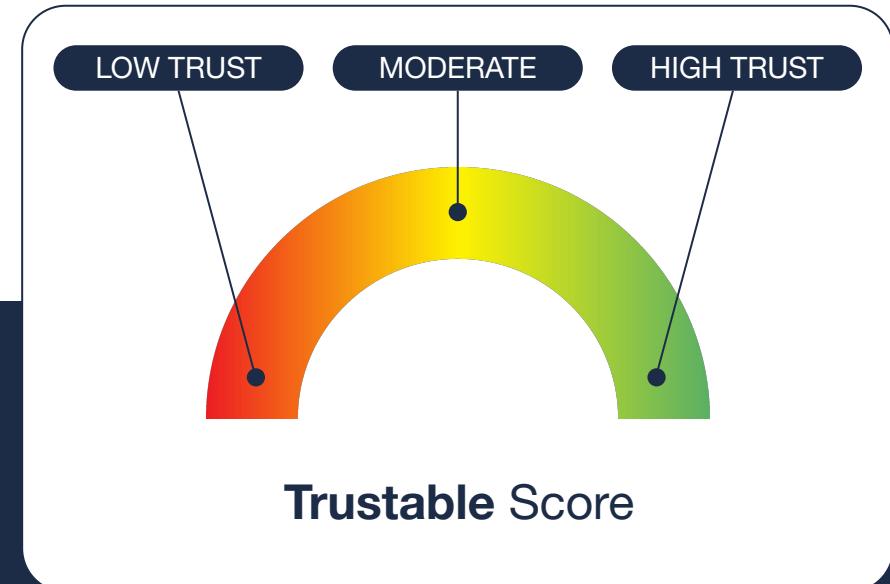


A FICO-Style Score for Software Trust

Like a credit score, TSF compiles assurance data into a single Trustable Score (“confidence score”) (e.g., 0–1000 scale).

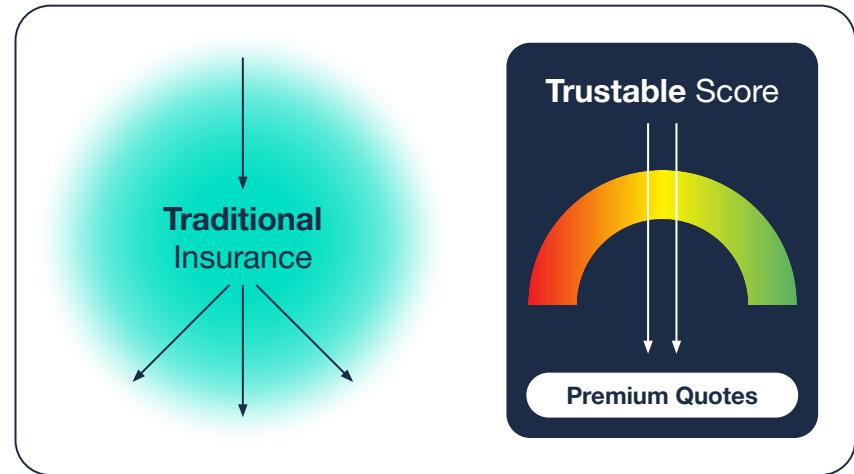
Helps OEMs, Tier-1s, regulators, and insurers quantify and compare software risk.

The scoring is dynamic, continually updated with each code change, security patch, or build.





Trustable Score →
Transparency →
Insurance Innovation



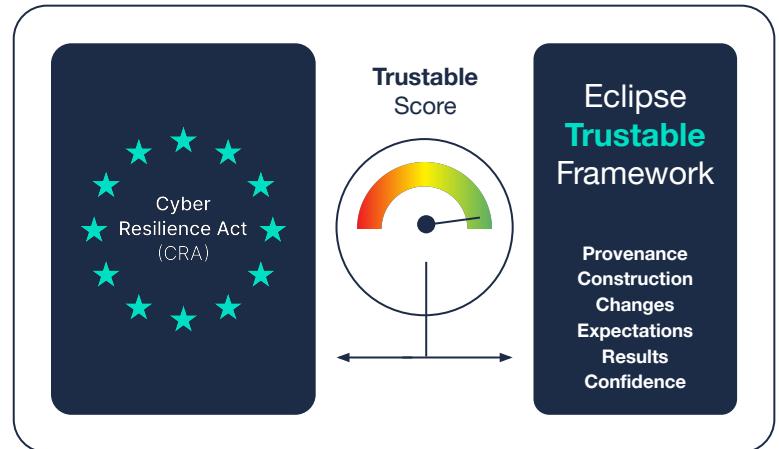
Without a risk metric, insurers price based on guesswork.
TSF's score unlocks precision.

Imagine Risk Retention Groups (RRGs) or pay-as-you-score insurance models: higher trust equals lower premiums.

Builds financial incentives for continuous quality and safety.



Operationalizing CRA: From Legal Text to Trustable Software Practice



The Cyber Resilience Act mandates obligations around:

- Secure development processes
- Provenance and documentation of components
- Vulnerability handling & reporting obligations
- 10-year support lifecycles

For lawyers/regulators:

TSF makes compliance auditable and measurable, reducing legal ambiguity.

For engineers:

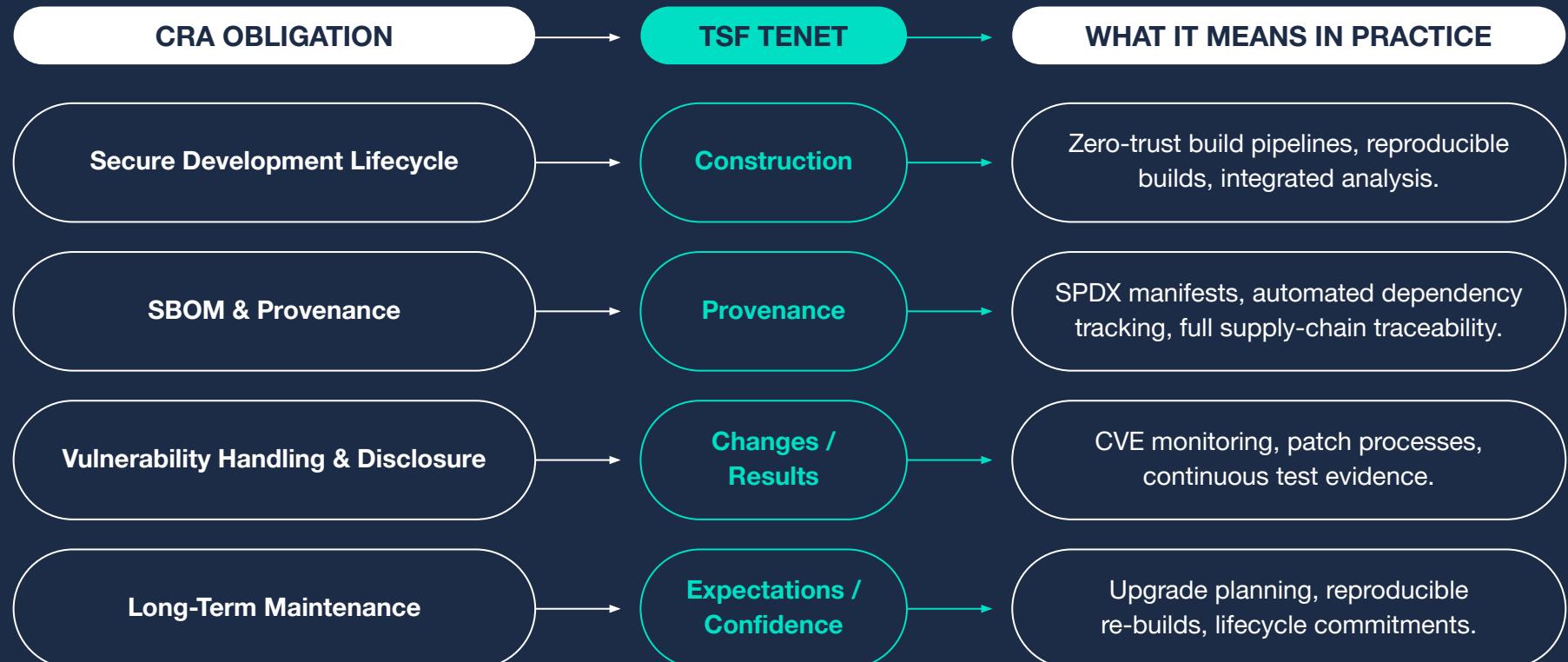
TSF provides the practical playbook — build pipelines, SPDX manifests, reproducibility — that plug into CRA requirements.

For executives:

TSF reduces CRA from a compliance cost center into a competitive advantage (measurable trust → lower risk → lower insurance premiums).



Mapping the **CRA** to the **TSF**



Single integration mainline (mainlines all the way down)

- Each component has its own releases
- Components set their own cadence
- Releases are tested vs integration mainline
- Components can do stable branches too
- Less disruption in integration mainline
- More stability in components
- More control of integration
- Enables and drives "shift-left" testing

Governance (inspired by Linux kernel Documentation/process/)

INTEGRATION MAINLINE

MAINLINES

RELEASE CANDIDATES

Safety/CRA (trustable) evidence + XXXXX for successful release

'STABLE'

SECURITY / HOTFIxes ONLY

EOL



'STABLE'

SECURITY / HOTFIxes ONLY

EOL



RELEASE CANDIDATES

Application 1

OSS Project A

Proprietary Project P

Application A

OSS Project B

???

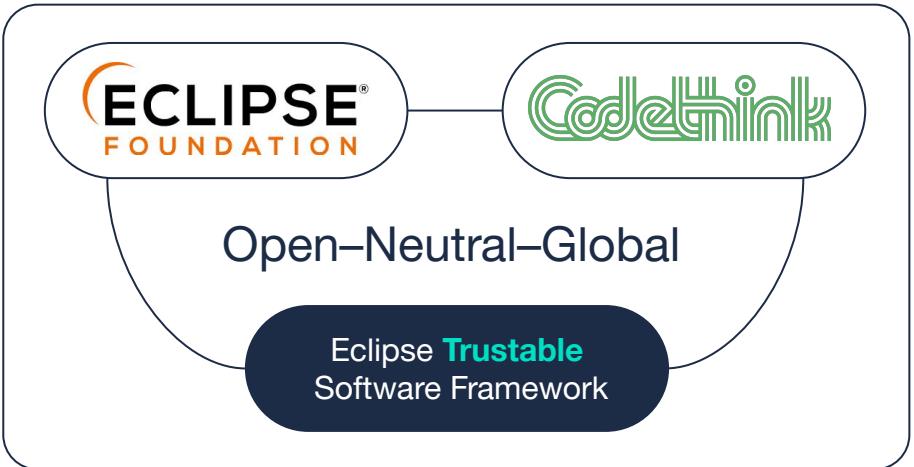


Join Us: Shape the Future of Software Trust

TSF is open. We need OEMs, suppliers, regulators, insurers, tool vendors, open-source contributors.

Ways to contribute:

- Pilot projects with real software stacks.
- Tool integrations (e.g., SPDX, CI/CD, scoring engines).
- Working groups around trust metrics and score governance.



Remember:

Software without evidence is just Belief.

<https://projects.eclipse.org/projects/technology.tsf>

<https://www.codethink.co.uk/trustable-software-framework.html>

We don't need perfect software.

We need **trustable** software.



Thank you

