Application of the Trustable Software Framework

Case Study NLohmann/Json in Eclipse S-CORE

24.02.2026

d-fine

analytical. quantitative. tech.

We tackle complex business tasks with a single team providing domain expertise, analytic skills and in-depth IT know-how

Who we are

Dr Thorsten Sickenberger

- Partner | Mobility & Automotive
- Focus on Data & Analytics, Software Engineering, Processes

Felix Mölders

- Consultant
- Background in Computer Science and Physics, Contributing to Eclipse SDV Working Group

Hidden Champion 2026/27

Winner of the category

Data & Analytics

Defining d-fine

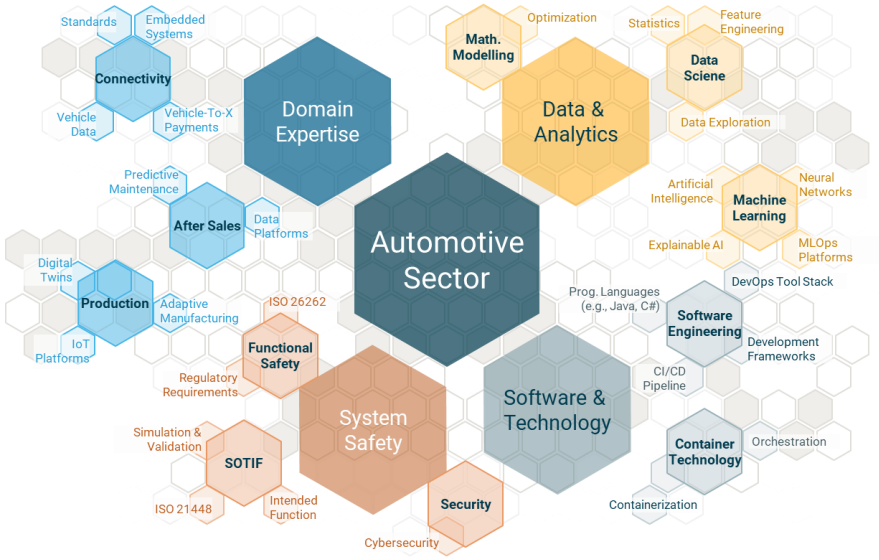
1.900+ *Employees*

>85 % *STEM Background*

2.000+ *Successful Projects*

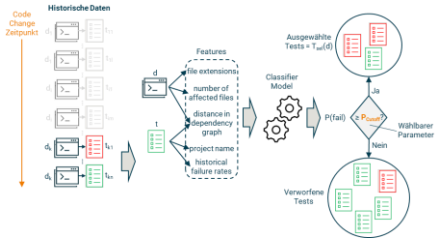
100 % *Climate Neutral*

- Banking & Capital Markets
- Insurance & Asset Management
- Chemicals, Energy & Manufacturing
- Pharma & Healthcare
- **Mobility & Transportation**
- Public Sector



With the combination of domain knowledge and technical skills, we support our clients throughout the entire project life cycle from conception to IT implementation.

Accelerate SDV Delivery with AI, Data Platforms and Process Automation



AI-Based Predictive Test Selection

Use of machine learning approach to pick software tests most likely to fail – reduces test effort and enables fast feedbacks

-85% test executions

-50% cost (HiL, ViL)

>99% faulty commits

New Revenues combining mobility & energy

d-fine has long-term experience in sector coupling projects

- **For end user**: Combine car sales with regional energy tariffs, enabling home charging and low charging costs
- **For TSOs:** Integrate fleet into backbone energy system to realize decentralized flexibilities and support Redispatch 3.0

Automation of Editorial Tasks in After Sales

Automating repetitive after-sales editorial tasks reduces workload, freeing editors to focus on reviews and exceptions.

Cost reduction
through automation

IV_02.10

Data & Analytics

Erfassen, Verarbeiten und Analysieren von strukturierten und unstrukturierten Daten sowie modellgestützte Umsetzung in technologische Lösungen

Hidden Champion 2026/27

Rang	Beratung	Punkte	s*
1	d-fine	443	74
2	Oliver Wyman	395	80
3	Accenture	375	108
4	Deloitte	371	94
5	McKinsey & Company	357	107
6	...	345	69
7	...	325	93
8	...	308	67
9	...	292	76
10	...	260	68
...	...	...	...

SDV development is driven by platform **architecture**, effective **AI** use, **cloud** solutions, and modern **DevOps**—areas where **d-fine is a reliable partner** for both open-source and in-house development.

Agenda

Eclipse S-CORE in a Nutshell

Eclipse S-CORE Creates the Foundation for Interoperable SDV-Architectures

Q1/2025
S-CORE v0.5 Concepts Agreed & Implemented

Q2/2025
S-CORE v0.5 Concepts Agreed & Implemented

11/2025
S-CORE v0.5 alpha

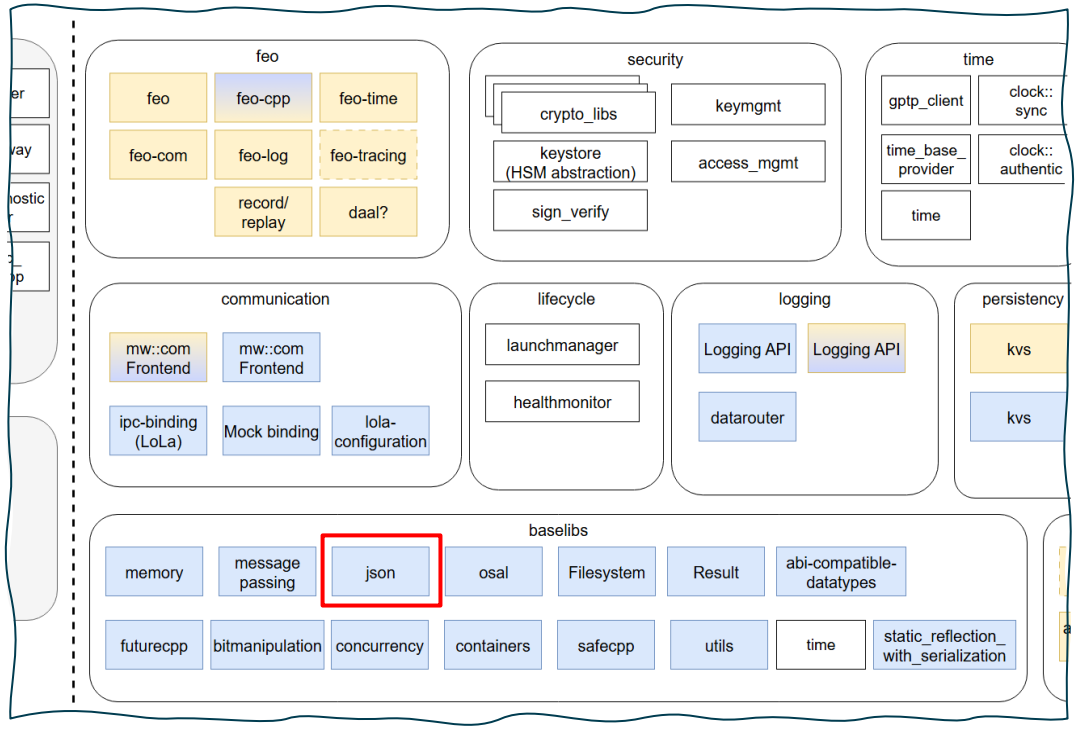
Q2/2026
S-CORE v0.5 final

Q3/2026
S-CORE v1.0 Alpha (tbc)

XX/2028
Series Car using S-CORE

Project Target

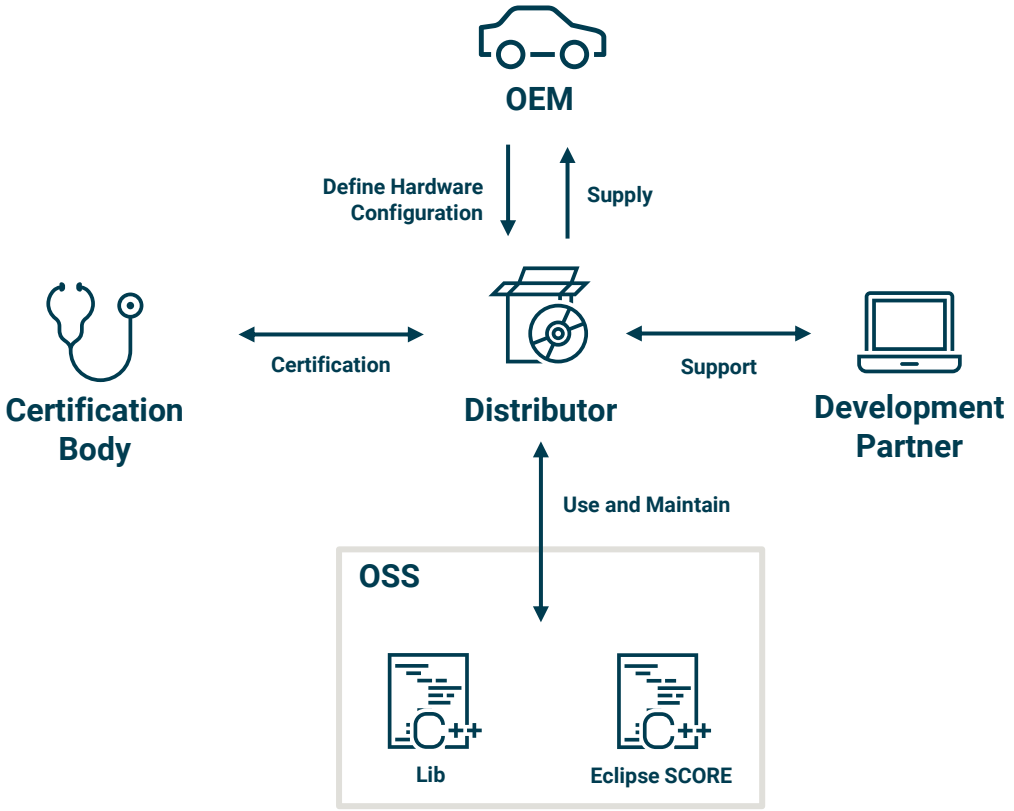
- A **modular, extensible runtime**/foundation for SDV applications on multi-processor ECUs, including interoperability between processors.
- Meets safety and security requirements in the automotive industry:
  - **Functional safety** according to ISO 26262
  - **Cybersecurity** according to ISO/SAE 21434 and UNECE WP.29
- Optimized end-to-end stack for efficiency and performance.



With S-CORE, OEMs benefit from a unified stack and significant cost savings.

OSS - OEMs Continue to Source their Software at Suppliers

Supplier take over certification and maintenance



The OEMs will define a hardware stack that S-CORE should run on.

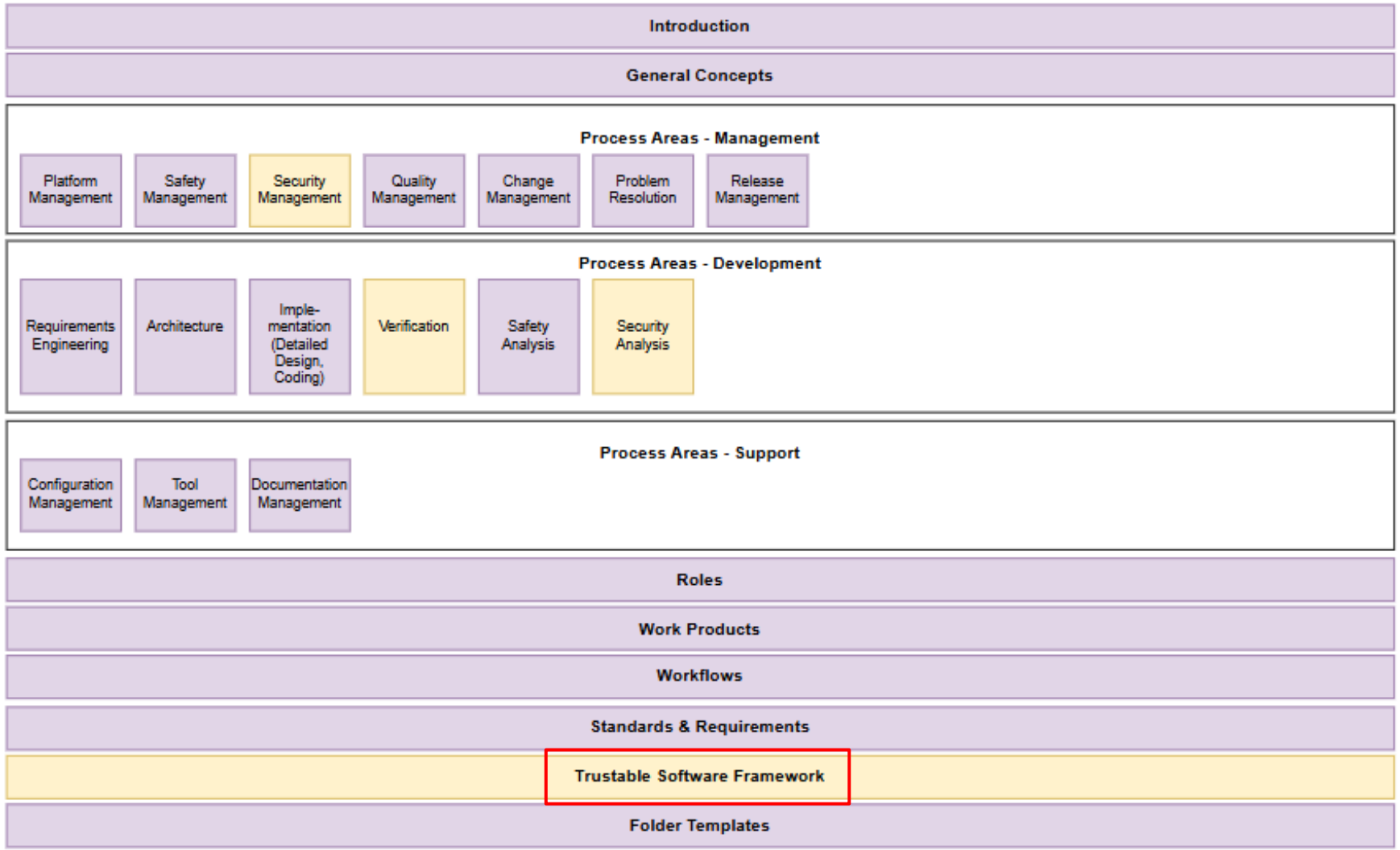
Distributors have then to perform the safety Certification.

The Distributor then supplies S-CORE to the OEM and takes over the liability.

Important: Eclipse S-CORE does **not** safety certify software, but the processes. Distributors have to perform the certification.

The distributor takes over the certification and liability for the S-CORE distribution.

The S-CORE Process Model Includes the Trustable Software Framework



Maturity Level

ML 0
Plan

ML 1
Initial

ML 2
Managed

ML 3
Defined/ Practiced

ML 4
Improving

The Eclipse Trustable Software Framework

Trustable Software Framework – a quantitative approach to risk assessment

Quantitative risk assessment for existing and new projects for certification according to ISO 26262

Problem/Motivation

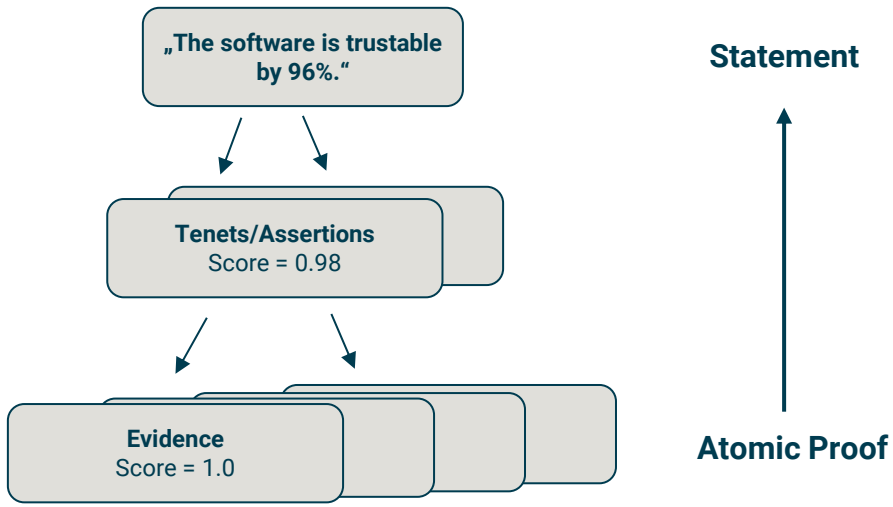
- Safety standards and frameworks focus on one-off checks for systematic errors in V-Model processes, which no longer fits agile, OSS-driven development.
- We need risk-based, continuous, and verifiable evaluation across the whole lifecycle—including supply chain and tooling.

Goal and approach of the TSF

- The TSF provides a methodology for continuous, quantitative risk assessment in complex software systems.

  It does so by modeling Expectations, Assertions/Tenets, and Evidence as a DAG directly in the Git repo.
- This replaces isolated office docs and proprietary requirements tools with repo-integrated, versioned, and scalable evidence management.

Trustable Graph

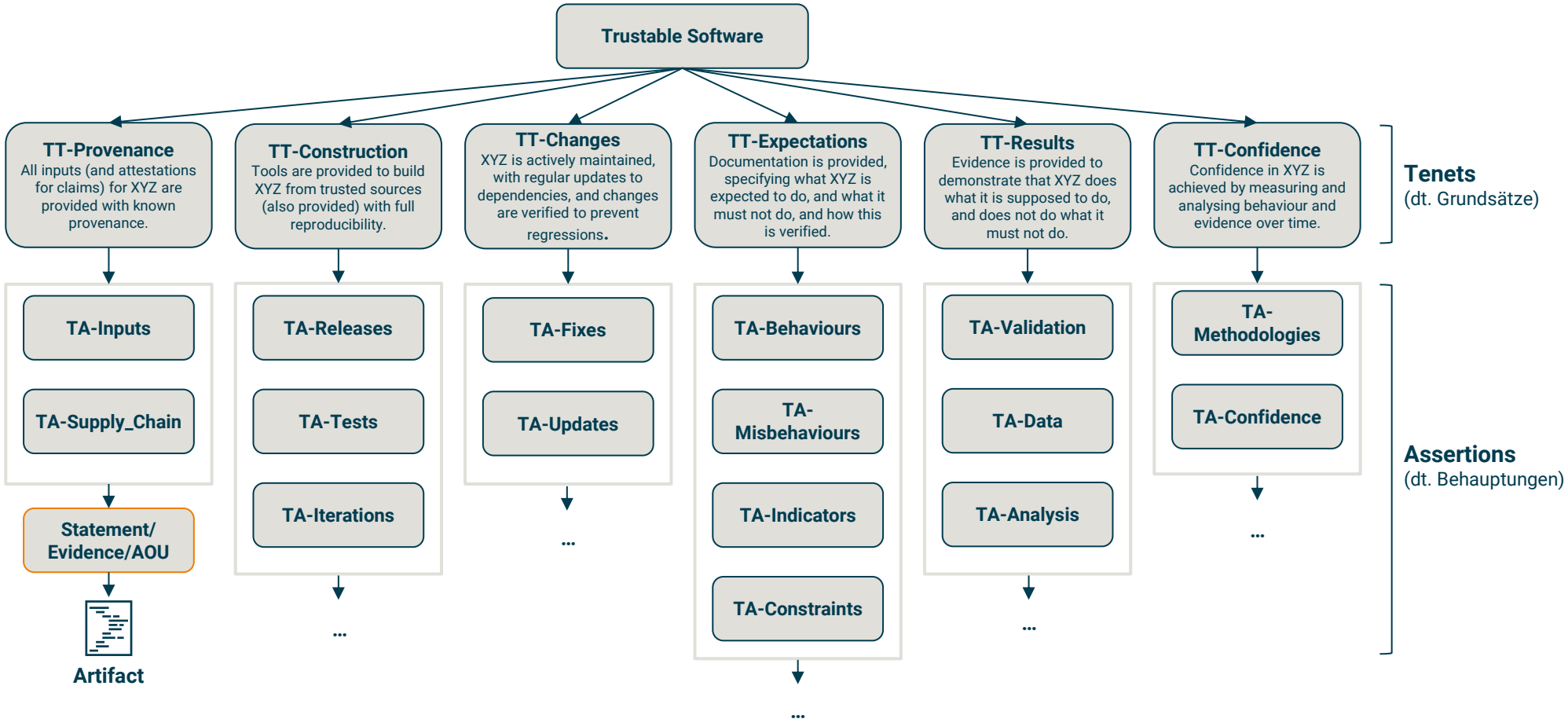


The score of a statement is calculated as the average of the scores of all supporting statements:

$$T(s) = \frac{1}{|\{s' : (s, s') \in L\}|} \sum_{s' \in \{s' : (s,s') \in L\}} T(s')$$

The Trustable Software Framework enables existing and open-source projects to be certified easily and transparently – for sustainable software quality.

The Trustable Graph



The Trustable Graph is a directed acyclic graph that defines Tenets and Assertions on trustable software. The Assertions must be either proven or linked to AOUs.

The Case Study – NLohmann/Json

What Are the Criteria for an Open-Source Library to Easily Apply the TSF?

TA-Inputs

Fewer dependencies mean less effort needs to be invested in the assessment.

TA-Fixes

Known Bugs and Misbehaviours should already be analyzed and fixed.

TA-Validation

Testing should already include unit AND stress tests.

TA-Releases

A mature CI pipeline reduces the adaptation effort required to ensure reproducibility.

TA-Behaviours

The Eclipse S-CORE requirements must be covered by the library.

TA-Confidence

The library should be on the market for a long time and be widely used.

TA-Tests

Mitigations for prohibited misbehaviours should already be implemented.

TA-Iterations

Code, build and usage instructions, tests, results and attestations must be available for all releases.

TA-Data

Extensive test data should be available and used for testing to cover edge cases.

Long story short: Library should be mature, extensively tested, well documented and have few dependencies.

Nlohmann/Json – JSON for Modern C++

JSON for Modern C++
What if JSON was part of modern C++?

Why was this library chosen?

Design Goals

Intuitive Syntax
- JSON feels like a first-class C++ type (rich operator overloading, Python-like usage)

Trivial Integration
- **Single header file** (json.hpp), no external dependencies or separate library
- **Pure C++11**, works out-of-the-box, available in major package managers

Serious testing & quality
- **100% unit test coverage**, including exceptional cases
- Continuous fuzzing via Google OSS-Fuzz
- No memory leaks

Niels Lohmann
nlohmann · he/him

Niels Lohmann initiated the library in 2013 because it was unhandy to work with JSON in C++

The Library is
- **Widely used**, even by OEMs during development but not safety certified
- **High maturity** – over a decade of improvement by a vivid community

Lower priority design goals
- **Memory efficiency**: simple representation (i.e. std::string, std::map etc.)
- **Speed**: not the fastest lib, focus is on fast development and easy JSON support

The Baselibs FT chose Nlohmann/Json because it is widely used, high maturity and has an optimal test coverage for starting with the Trustable Software Framework.

Statements Support the Assertions and Provide Evidences

This Statement is normative, which means it is contributing to the overall score. Any "normative: false" is just informative and is not actually included in the scoring.

Here the reference is given as a project_website, which is referencing a GitHub workflow.

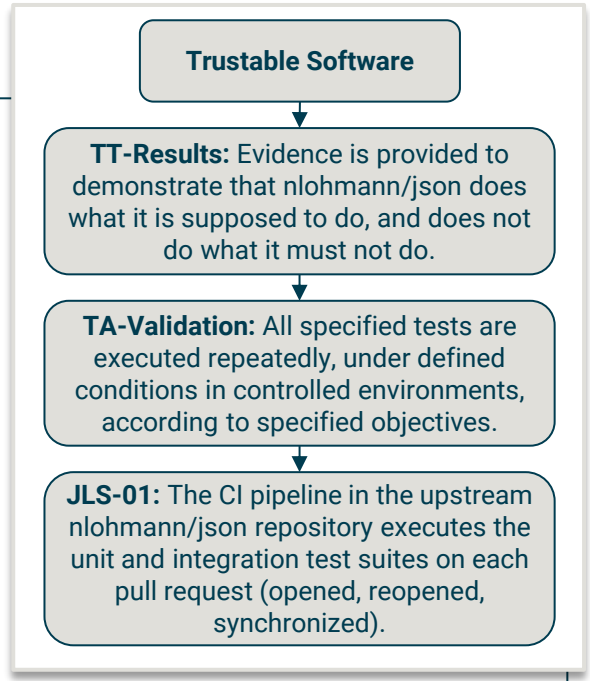
In addition to a reference, a validator was also used here. That is looking for the website response time to be under a certain value.

Two SMEs gave a score.

This is the actual statement.

```yaml
---
level: 1.1
normative: true
references:
        - type: project_website
          url: "https://github.com/nlohmann/json/blob/develop/.github/workflows/ubuntu.yml#L9"
          description: "The trigger condition for the CI workflow that executes the test suites."
evidence:
        type: https_response_time
        configuration:
                target_seconds: 2
                urls:
                    - "https://github.com/nlohmann/json/blob/develop/.github/workflows/ubuntu.yml#L9"
score:
    Jonas-Kirchhoff: 1.0
    Erikhu1: 1.0
---

The CI pipeline in the upstream nlohmann/json repository executes the unit and integration test suites on each pull request (opened, reopened, synchronized).
```



Statements support/proof Assertions and may link references and evidences.
Each normative statement is scored by an SME.

Subject Matter Experts Review the Statements

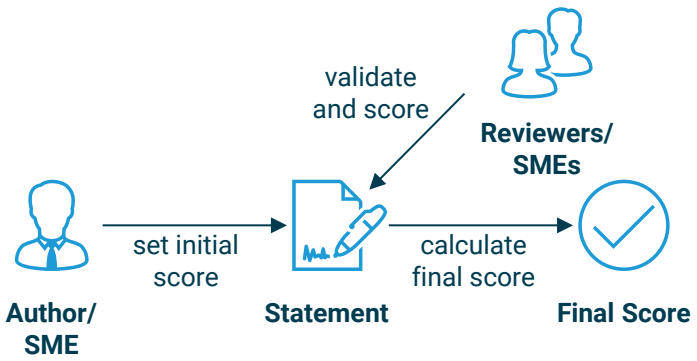
Scoring

- SMEs review the TSF statements and have two tasks:
  - **Validate the statement** – Is the statement suitable to support the Assertion?
  - **Set a score** – How confident are we that the statement is fulfilled?
- **Converging Convidence** – By performing multiple SME reviews it is expected to receive a confidence score close to the true value.

```
8     score:
9         Erikhu1: 1.0
10        aschemmel-tech: 0.8
```

Review Process

- The Author/SME creates the statement and sets an initial score.
- The Reviewers/SMEs validate the statement and score it.
- Finally, the resulting score is calculated by Trudag.



Our Experience

- Chose wisely how many Reviews are necessary, as this is time consuming. **We usually use two reviewers/SMEs.**
- **Diverging scores** are a great indication that the argumentation has flakes.
- Calibration: A **common understanding is needed** across reviewers when assigning **scores between zero and one**.

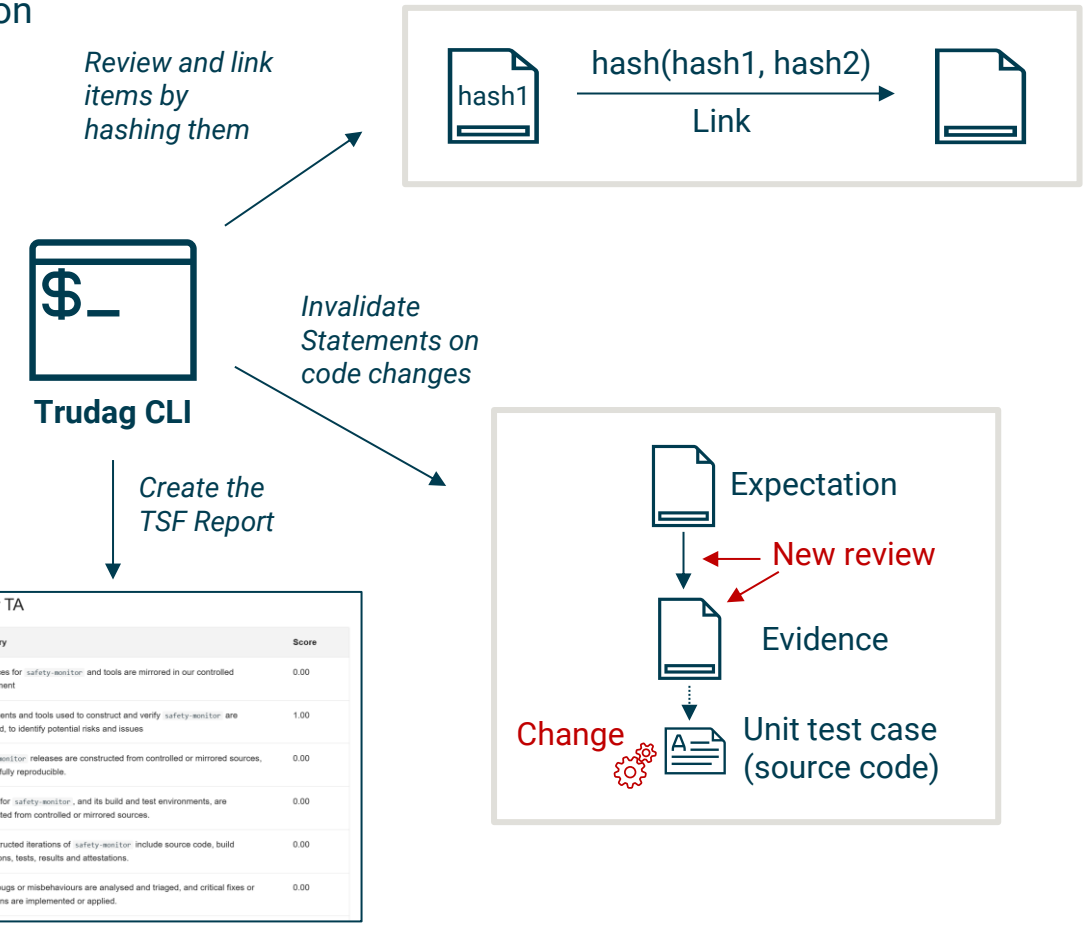
SMEs assign confidence scores and use disagreements to refine the TSF statements.

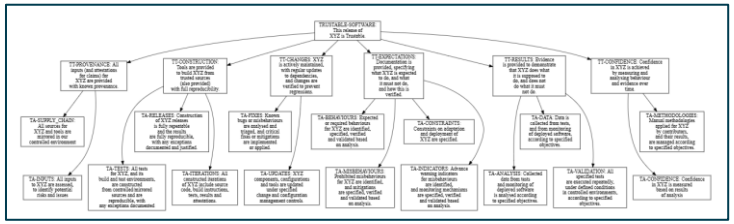
The TSF Review Process is Supported by Trudag

Trudag is a CLI tool intended to support the application of the Trustable Software Framework

Trudag is used to:

- Execute Reviews
- Create links between statements
- Invalidate statements on changes of references
- Render the TSF Graph
- Create the TSF Report

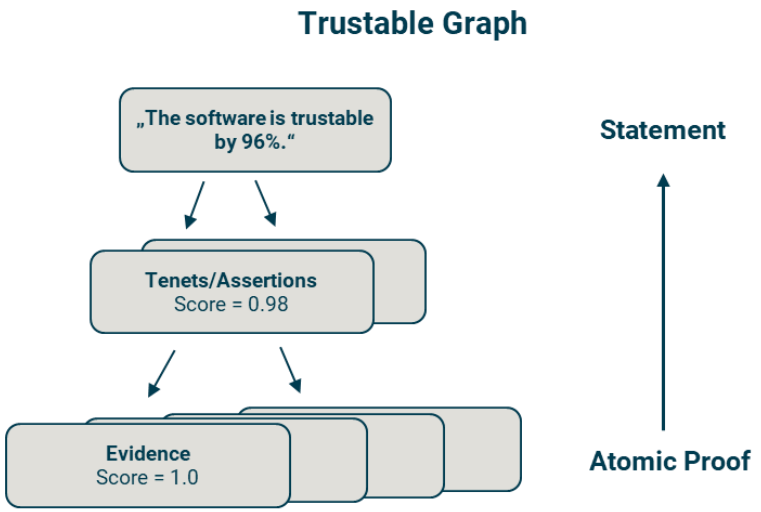




Trudag CLI supports the manual and automated execution of the Trustable Software Framework.

Our Experience with the Trustable Software Framework

From Open-Source Libraries to Certifiable Automotive Software



The score of a statement is calculated as the average of the scores of all supporting statements:

$$T(s) = \frac{1}{|\{s' : (s, s') \in L\}|} \sum_{s' \in \{s' : (s,s') \in L\}} T(s')$$

Pros

- Supports the a posteriori certification of open-source libraries
- Extensive Documentation and Check-lists for application allowing for a steep learning curve
- Supportive tooling by Trudag CLI
- Score supported inference whether a release is trustable and safe to use.
- Great support by its founder Codethink

Challenges

- Nlohmann/Json is the first open-source application of TSF. Hence, we have no examples as reference.
- Exida is currently the only Certification Body familiar with the Trustable Software Framework.

TSF is not only a good match for certifying open-source libraries but also for agile functional safety development withing the Automotive industry.

Questions and Comments

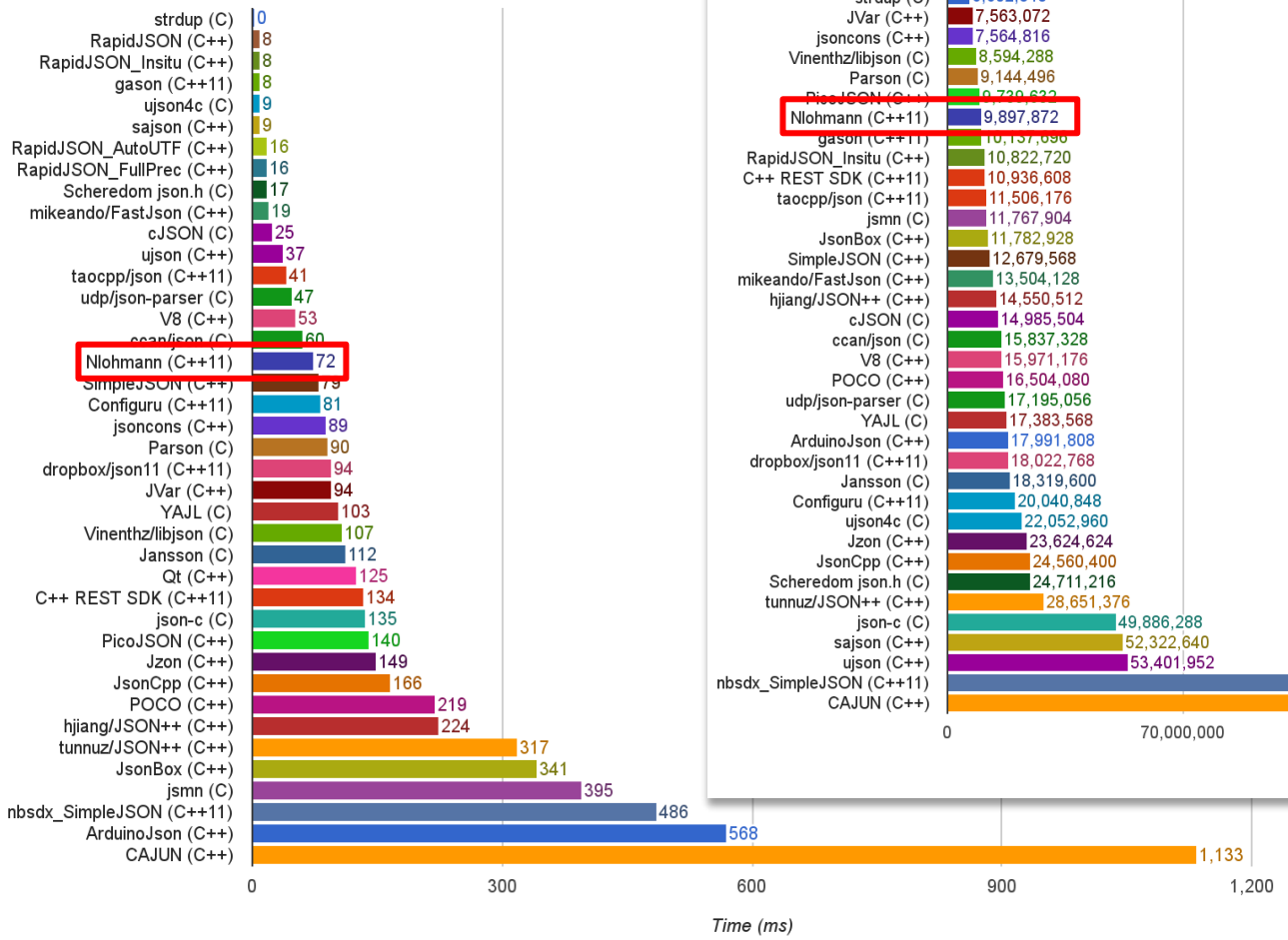
Questions?
Comments?

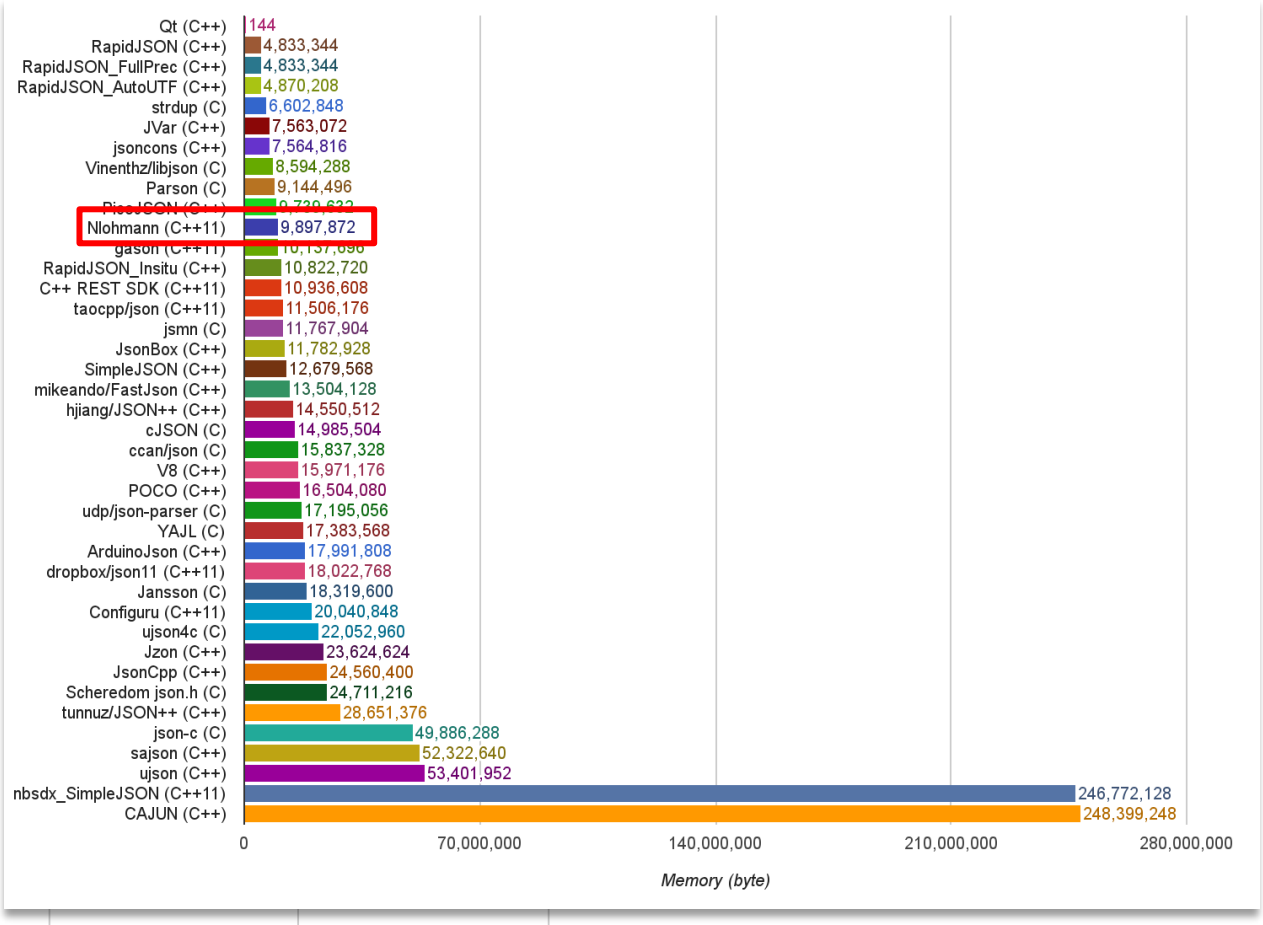
I am happy to get your comments and feedback.

Please contact us for further discussions. Thank you!

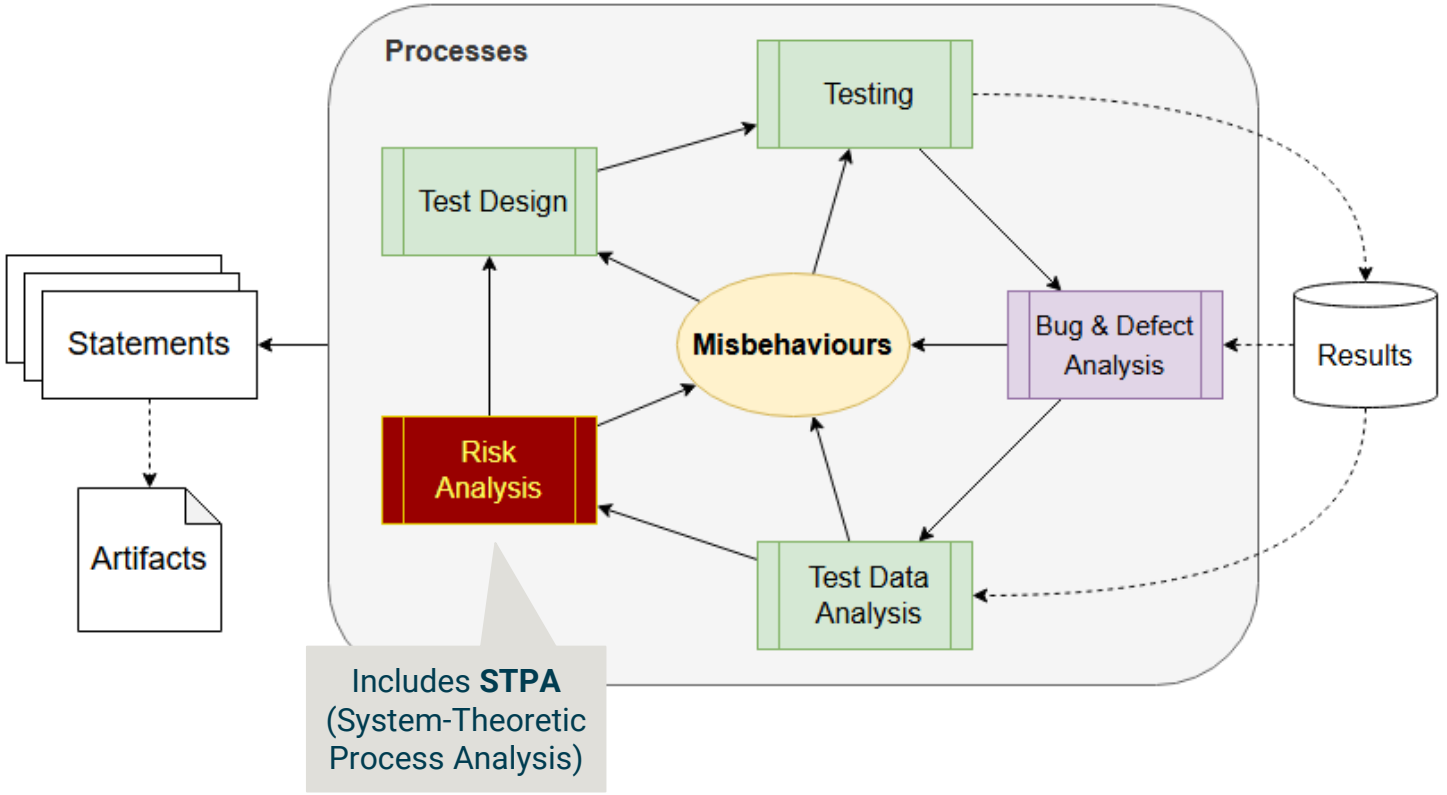
Nlohmann/Json Offers Simplicity, High Test-Coverage While Having Acceptable Runtime and Memory Consumption

JSON for Modern C++
What if JSON was part of modern C++?





RAFIA - Risk Analysis, Automated Testing, Fault Injection, Mitigation and Compliance



Identified and confirmed **Misbehaviours** correspond to **Faults** i.e. deviations from **Expectations**.

Misbehaviours are captured as part of **Statements** and **Artifacts**.

Identify new Misbehaviours using **Soak** tests, **Stress** tests and **Performance** tests.

Risk Analysis may include **STPA** (System-Theoretic Process Analysis) equivalent to FTA, FMECA, ETA, HAZOP.

The classical functional safety measures are mapped to RAFIA and STPA.

Your Contacts at d-fine

Felix Mölders
Consultant
Tel +49 211 8639512-457
Mobile +49 173 7975264
Felix.Moelders@d-fine.com

Dr Thorsten Sickenberger
Partner
Tel +49 69 90737-537
Mobile +49 162 2631375
Thorsten.Sickenberger@d-fine.com

d-fine GmbH
An der Hauptwache 7
D-60313 Frankfurt/Main
Germany

Frankfurt **Berlin** **Dusseldorf** **Hamburg**

London **Milan** **Munich** **Stockholm**

Utrecht **Vienna** **Zurich**

d-fine

analytical. quantitative. tech.