Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024

Enabling tooling independent exchange of Requirements and other SW Engineering related information with the upcoming SPDX Safety Profile

Nicole Pappler, AlektoMetis

Safe Systems with Linux Micro Conference

Sept 20, 2024

Whoami – Nicole Pappler

AlektoMetis
...we enable digital innovation.

Professional History:

Been working in production maintenance, automotive, ECU software development

All my projects had some safety criticality

Started to focus on Functional Safety about 13 years ago

Currently:

Tech consulting as part of AlektoMetis

Supporting my customers regarding Functional Safety, Security & compliant use of open source

Involved in some open source projects:

- Zephyr (Functional Safety Manager)
- ELISA (Medical & Systems Group)
- FuSa for SPDX SIG
- OpenChain (3rd party certification with TÜV SÜD)

What else?

GitHub, Discord, etc: @nicpappler

About today

What’s the issue?

Why do we need traceability?

The documentation of intentions and evidences.

SPDX Safety Model

Some history…

Why traceability?

Photo by Max Nüstedt on Unsplash

Safety based on

- Mechanics
- Safe construction
- Built with durable and suitable nuts, screws, bolts, …

Identification of mechanical parts

Standardized parts

Defined material properties

- Tensile Strength
- Composition

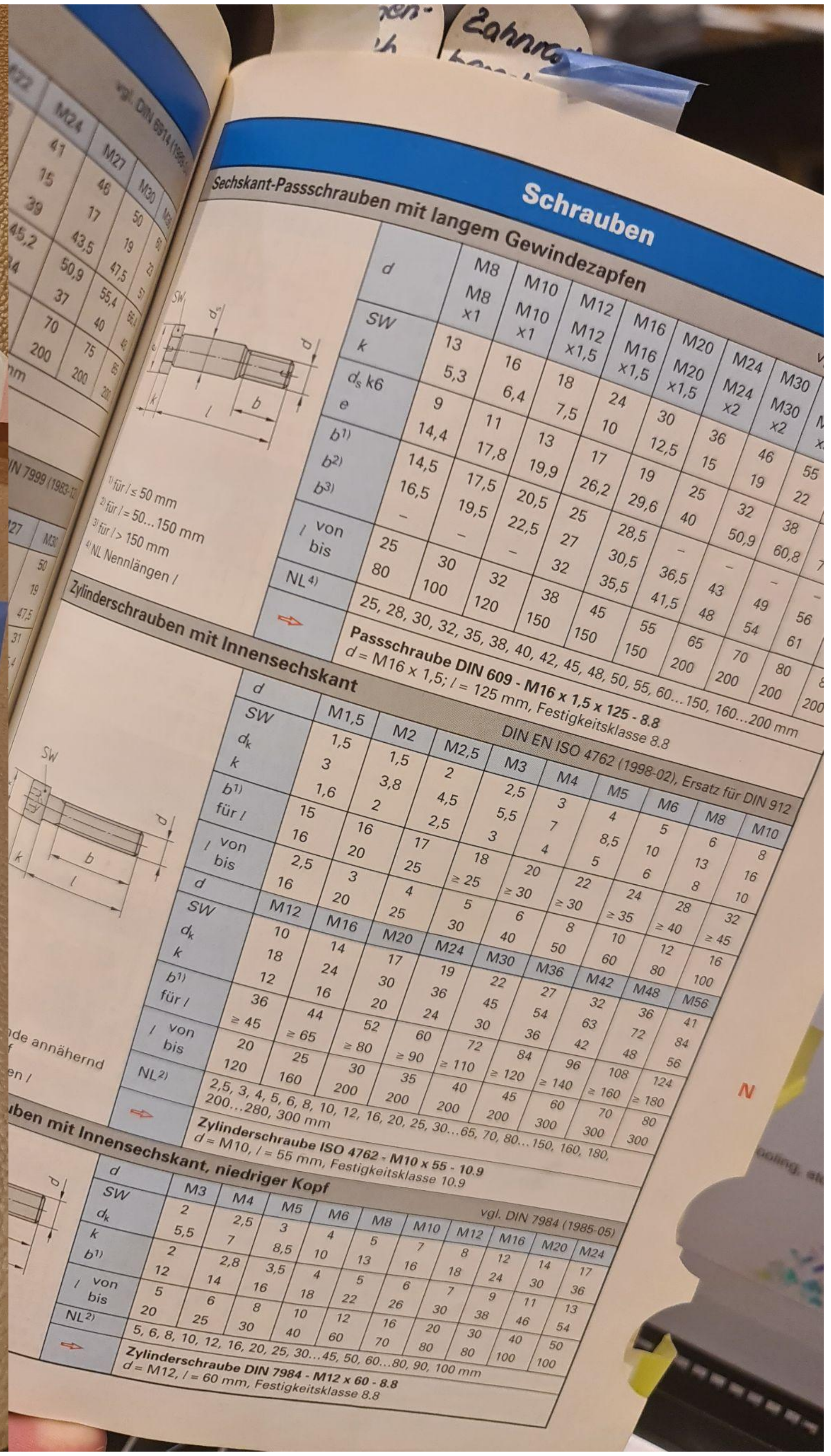
Dimensions

Tools

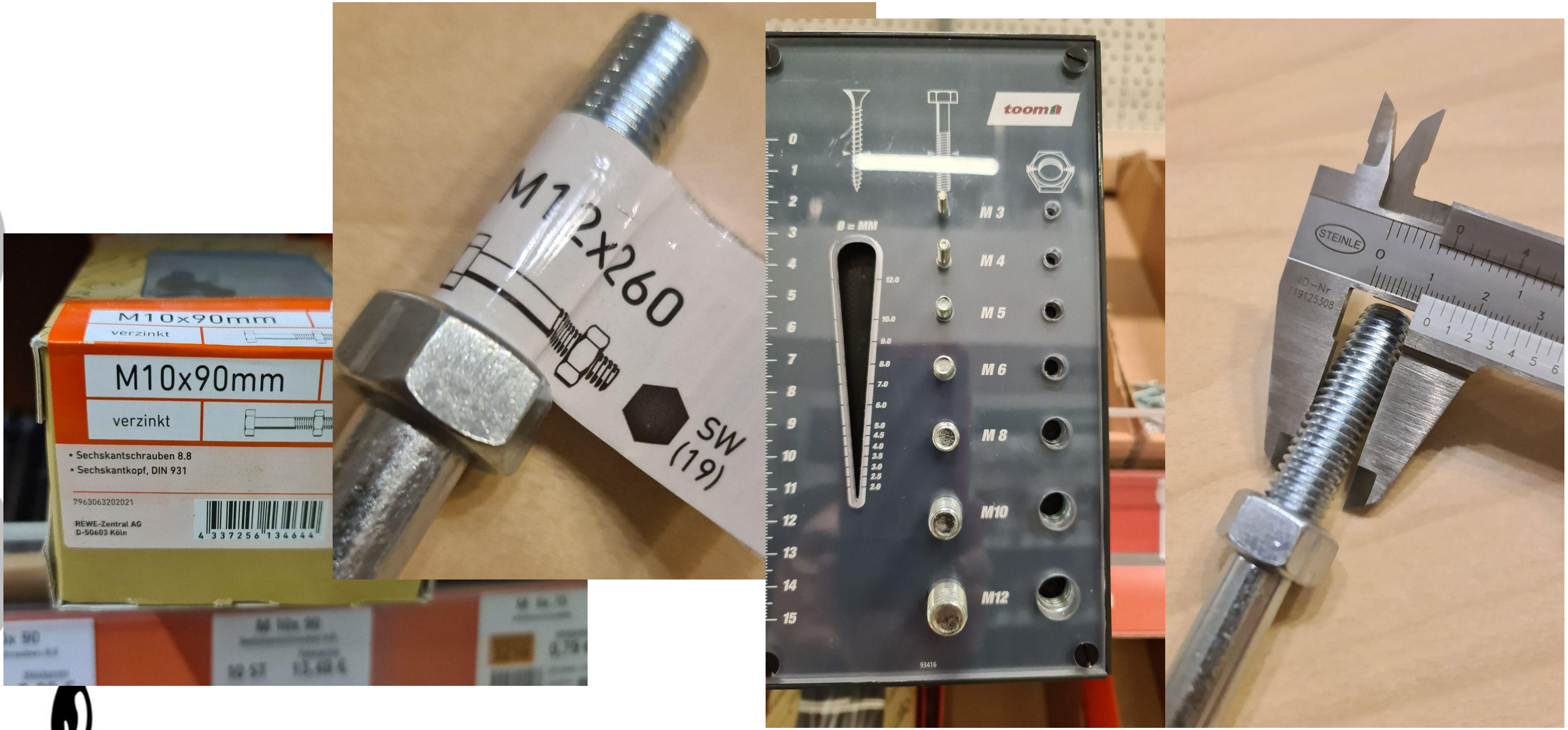
Handling (max torque, right tooling, etc.)

Serial numbers

Lot identification

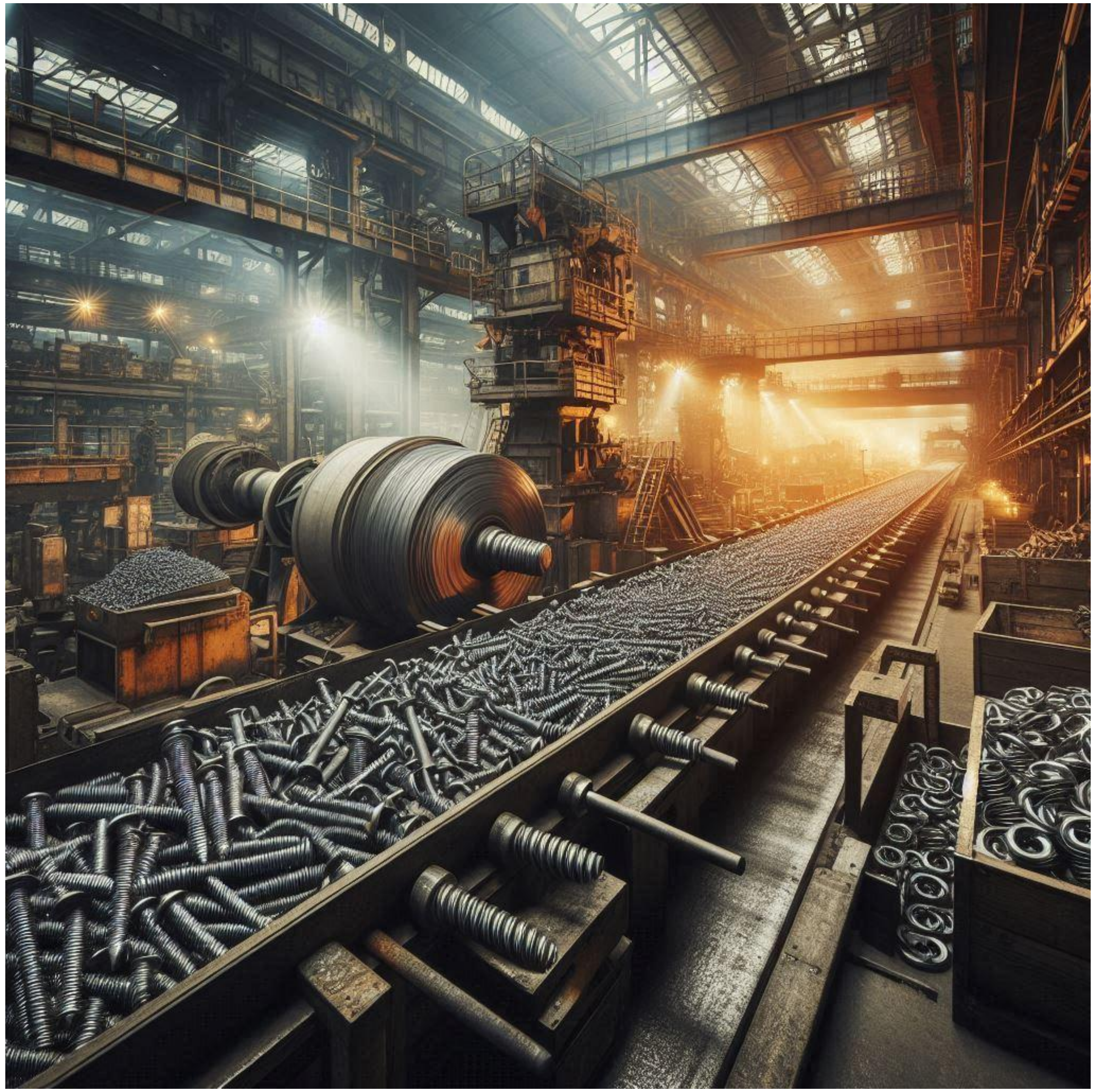

Schrauben
Sechskant-Passschrauben mit langem Gewindezapfen
Zylinderschrauben mit Innensechskant
DIN EN ISO 4762
Zylinderschrauben mit Innensechskant, niedriger Kopf
vgl. DIN 7984

Easy identification & verification!



Why do we want all this information?

- Manufacturing information
- Material vulnerabilities affect the whole lot
- Easy identification of properties, like size, dimensions, tensile strength, surface tempering…
- Identification of suitable tooling and tool usage limitations (torque, handling, assembling and disassembling cycles…)
- Identification of suitable accessories (self locking nuts, washers, …)



Disaster and Incident Response

Eschede train disaster 1998:

- analysis of the incident including
  - material information
  - construction & manufacturing information
  - maintenance information

- fatigue crack in one single wheel

https://en.wikipedia.org/wiki/Eschede_train_disaster

Follow up actions:

- legal actions
- redesign of the wheels
- improved maintenance procedures
- improved escape ways (easy cracking windows)

Which one to trust with your life?

A standardized, traceable screw with defined material and properties?

Some screws you found on the internet?

Back to today

More than pure mechanics

Mechanical Safety
Electrical Safety
Environmental Safety
Functional Safety
Cyber Security

Mechanics

Electrical & Electronic devices

Software

Photo by Daniel Abadia on Unsplash

More than pure mechanics

Mechanical Safety
Electrical Safety
Environmental Safety
Functional Safety
Cyber Security

Mechanics

Electrical & Electronic devices

Thousands of software components, billions of lines of software

Photo by Daniel Abadia on Unsplash

Functional Safety and its added value

Definition of Functional Safety

- **Safety** – the freedom from unacceptable risk of physical injury or of damage to the health of people, either directly, or indirectly as a result of damage to property or to the environment

- **Functional Safety**
  - the part of safety that depends on a system or equipment operating correctly in response to its inputs
  - Detecting potentially dangerous conditions, resulting either in the activation of a protective or corrective device or mechanisms to prevent hazardous events or in providing mitigation measures to reduce the consequences of the hazardous event.

- **Know your risks, now what you need, know what you have implemented, document your decisions and evidences**

Functional Safety - Systematic Capability of Software

Safety is a system property!

But:

Systematic capability is the general assumption, that

- if development, test and deployment of a system follow a specific set of tasks and
- there is evidence for adherence to these tasks
- (and under the assumption that the system architecture supports safety)

⇒ Software is capable of performing as intended

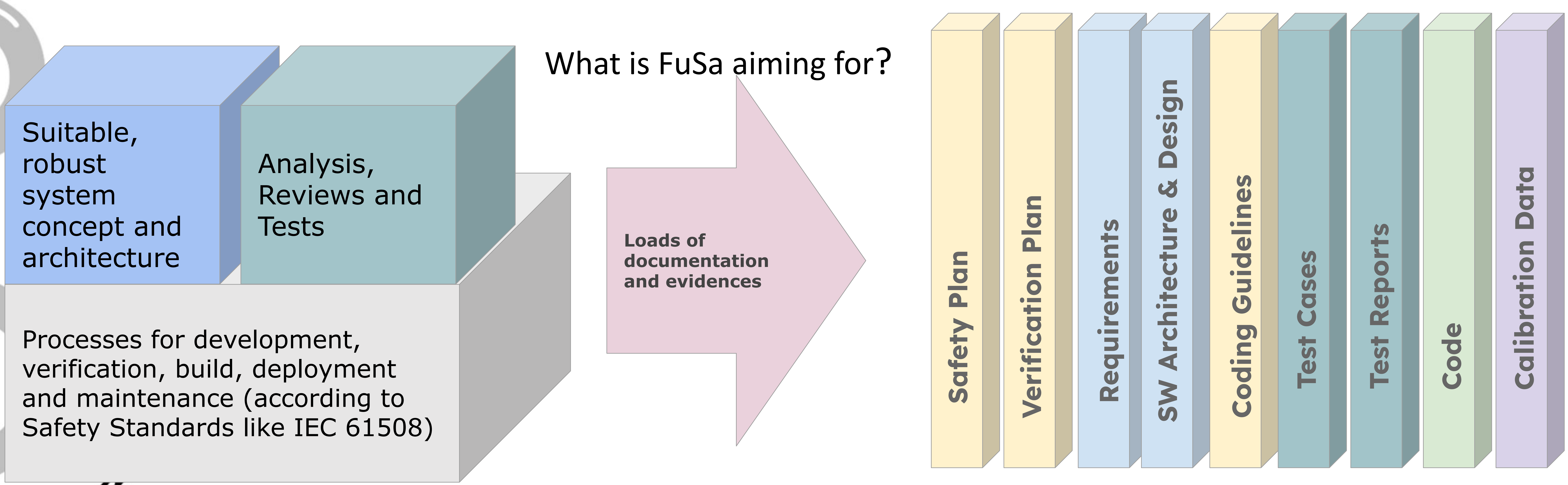
Functional Safety Standards

What are these tasks and evidences?

- Usually defined in Safety Standards
- Focus: Unique IDs, traceability, completeness, evidences

⇒ define your dependencies (also inside of your project!) **and keep them up to date**!

Safety Architecture and Documentation



Suitable, robust system concept and architecture

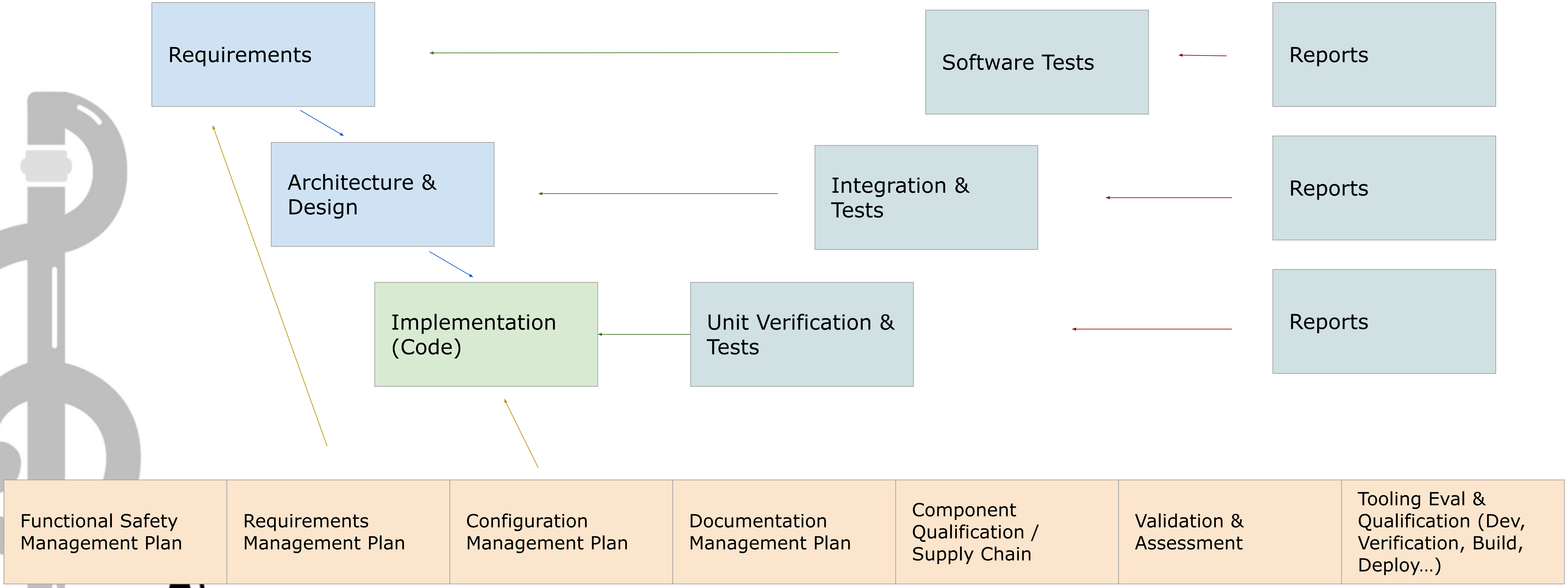
Analysis, Reviews and Tests

Processes for development, verification, build, deployment and maintenance (according to Safety Standards like IEC 61508)

What is FuSa aiming for?

Loads of documentation and evidences

Dependencies in a FuSa Project



Requirements

Architecture & Design

Implementation (Code)

Software Tests

Integration & Tests

Unit Verification & Tests

Reports

Reports

Reports

Functional Safety Management Plan

Requirements Management Plan

Configuration Management Plan

Documentation Management Plan

Component Qualification / Supply Chain

Validation & Assessment

Tooling Eval & Qualification (Dev, Verification, Build, Deploy…)

FuSa documentation structure

All FuSa related documentation is part of the Safety Case!

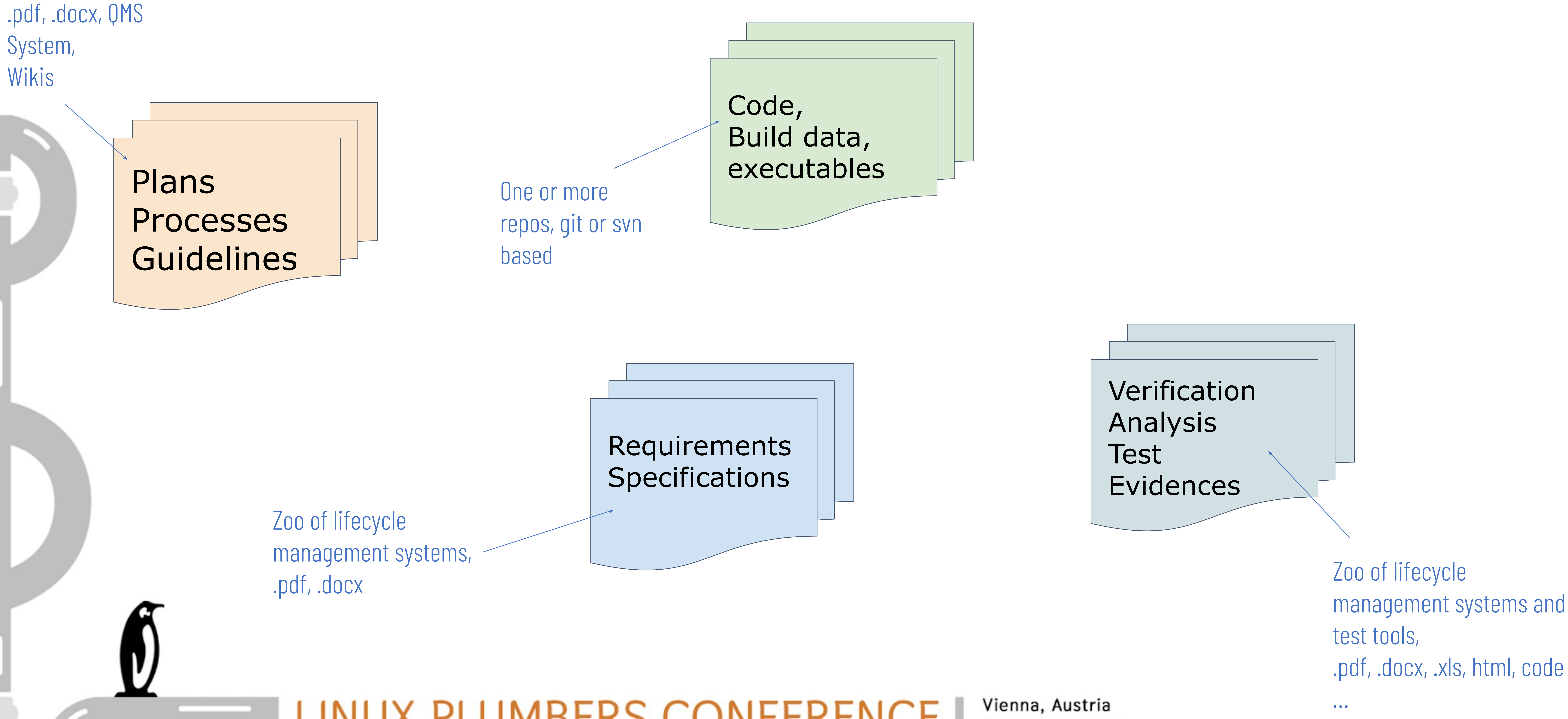
Think of all these documents as part of the release - each document is part of the Bill of Material, as is each screw, each microcontroller and each piece of software!

Plans
Processes
Guidelines

Requirements
Specifications

Verification
Analysis
Test
Evidences

Data Structure of current FuSa projects…



Plans
Processes
Guidelines

.pdf, .docx, QMS System,
Wikis

Code,
Build data,
executables

One or more repos, git or svn based

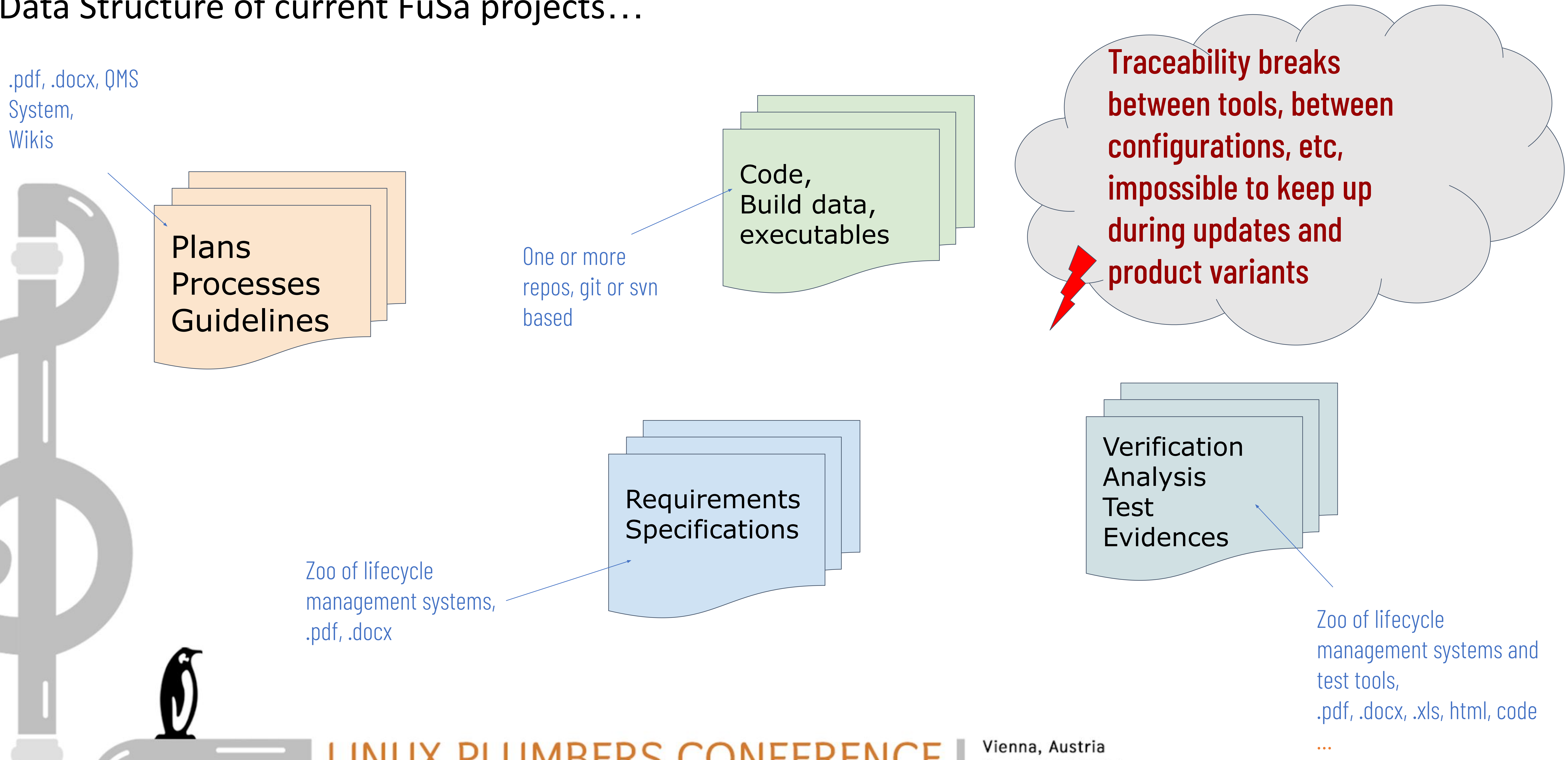
Requirements
Specifications

Zoo of lifecycle management systems,
.pdf, .docx

Verification
Analysis
Test
Evidences

Zoo of lifecycle management systems and test tools,
.pdf, .docx, .xls, html, code
…

Data Structure of current FuSa projects…



.pdf, .docx, QMS System, Wikis

Plans
Processes
Guidelines

One or more repos, git or svn based

Code,
Build data,
executables

Traceability breaks between tools, between configurations, etc, impossible to keep up during updates and product variants

Zoo of lifecycle management systems, .pdf, .docx

Requirements
Specifications

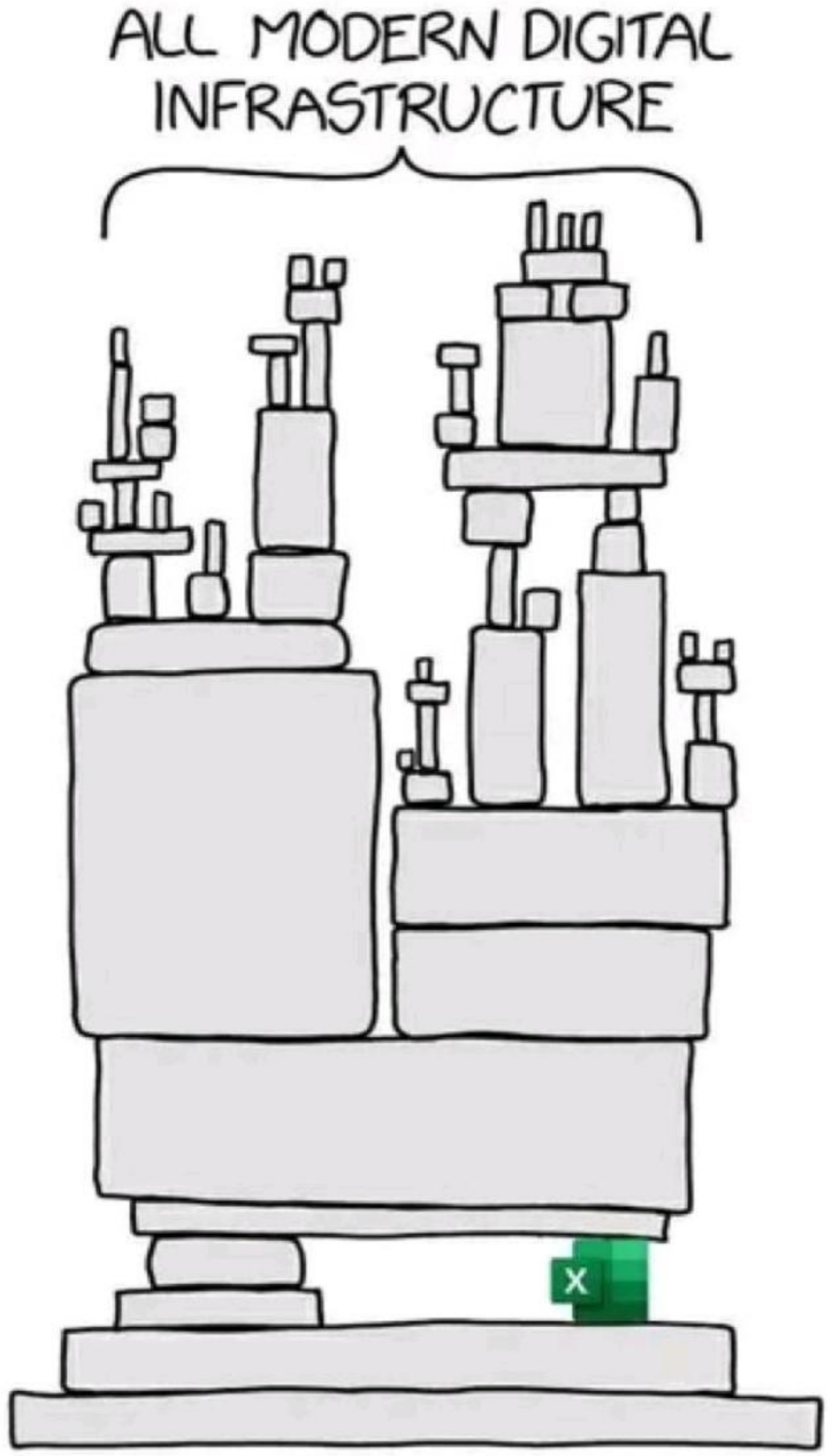
Verification
Analysis
Test
Evidences

Zoo of lifecycle management systems and test tools,
.pdf, .docx, .xls, html, code
…

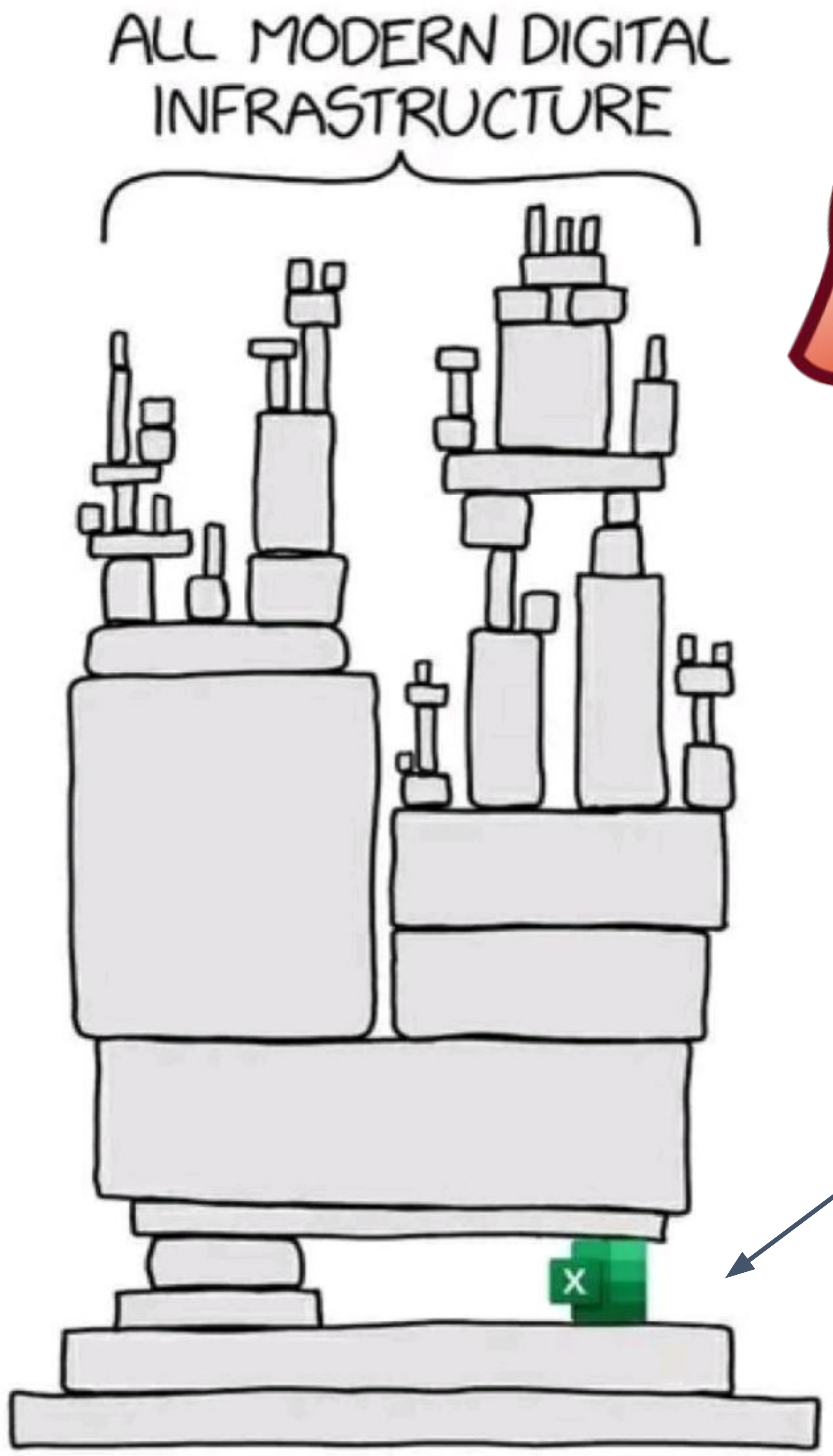
No 1 Safety Information Exchange Format

Any guesses????

No 1 Safety Information Exchange Format



No 1 Safety Information Exchange Format



draft_2005TemplateSafetyCase_thisproject_final_forTraceingv06.xls

Emoji by emojidex

Why we do need SPDX for Functional Safety

About SPDX

SPDX metadata includes details about creation and distribution, including the following:

- software composition, for collections of software (Packages), individual Files, and portions of files (Snippets)
- software build information
- artificial intelligence (AI) models
- datasets
- creator, supplier and distributor identity information
- provenance and integrity
- licenses and copyrights, including a curated list of licenses and exceptions
- security vulnerabilities, defects, and other quality data
- relationships between system elements
- software usage and lifecycle
- mechanisms to enable annotating SPDX elements and linking between multiple SPDX Documents

SPDX Safety Dependencies in a FuSa Project

Requirements

Architecture & Design

Implementation (Code)

Software Tests

Integration & Tests

Unit Verification & Tests

Reports

Reports

Reports

Functional Safety Management Plan

Requirements Management Plan

Configuration Management Plan

Documentation Management Plan

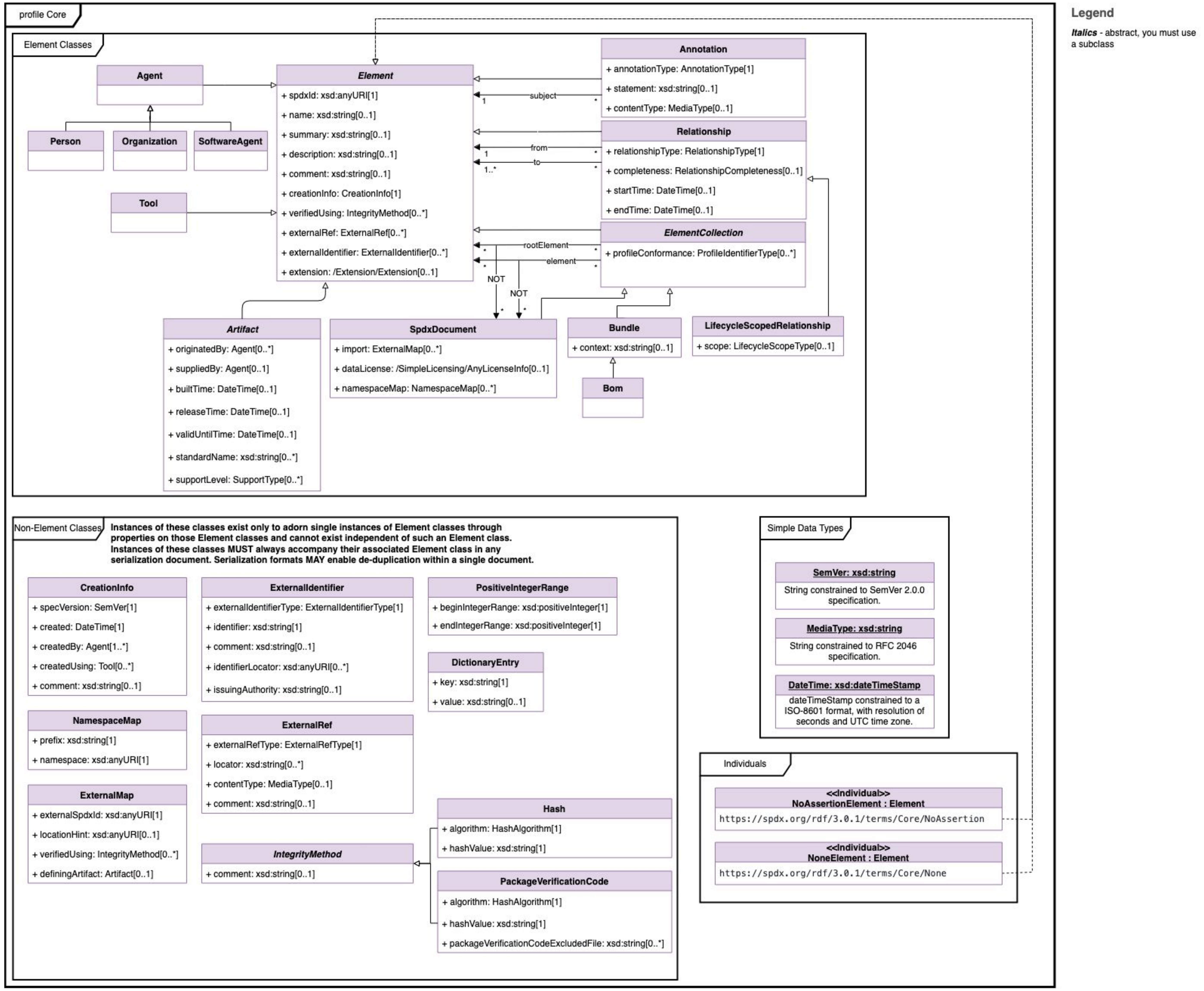
Component Qualification / Supply Chain

Validation & Assessment

Tooling Eval & Qualification (Dev, Verification, Build, Deploy…)

SPDX model

https://github.com/spdx/spdx-3-model/blob/main/images/model-core.png



Agent

Person

Organization

SoftwareAgent

Tool

Element
- + spdxId: xsd:anyURI[1]
- + name: xsd:string[0..1]
- + summary: xsd:string[0..1]
- + description: xsd:string[0..1]
- + comment: xsd:string[0..1]
- + creationInfo: CreationInfo[1]
- + verifiedUsing: IntegrityMethod[0..*]
- + externalRef: ExternalRef[0..*]
- + externalIdentifier: ExternalIdentifier[0..*]
- + extension: /Extension/Extension[0..1]

Annotation
- + annotationType: AnnotationType[1]
- + statement: xsd:string[0..1]
- + contentType: MediaType[0..1]

Relationship
- + relationshipType: RelationshipType[1]
- + completeness: RelationshipCompleteness[0..1]
- + startTime: DateTime[0..1]
- + endTime: DateTime[0..1]

ElementCollection
- + profileConformance: ProfileIdentifierType[0..*]

Artifact
- + originatedBy: Agent[0..*]
- + suppliedBy: Agent[0..1]
- + builtTime: DateTime[0..1]
- + releaseTime: DateTime[0..1]
- + validUntilTime: DateTime[0..1]
- + standardName: xsd:string[0..*]
- + supportLevel: SupportType[0..*]

SpdxDocument
- + import: ExternalMap[0..*]
- + dataLicense: /SimpleLicensing/AnyLicenseInfo[0..1]
- + namespaceMap: NamespaceMap[0..*]

Bundle
- + context: xsd:string[0..1]

Bom

LifecycleScopedRelationship
- + scope: LifecycleScopeType[0..1]

Non-Element Classes

Instances of these classes exist only to adorn single instances of Element classes through properties on those Element classes and cannot exist independent of such an Element class. Instances of these classes MUST always accompany their associated Element class in any serialization document. Serialization formats MAY enable de-duplication within a single document.

CreationInfo
- + specVersion: SemVer[1]
- + created: DateTime[1]
- + createdBy: Agent[1..*]
- + createdUsing: Tool[0..*]
- + comment: xsd:string[0..1]

NamespaceMap
- + prefix: xsd:string[1]
- + namespace: xsd:anyURI[1]

ExternalMap
- + externalSpdxId: xsd:anyURI[1]
- + locationHint: xsd:anyURI[0..1]
- + verifiedUsing: IntegrityMethod[0..*]
- + definingArtifact: Artifact[0..1]

ExternalIdentifier
- + externalIdentifierType: ExternalIdentifierType[1]
- + identifier: xsd:string[1]
- + comment: xsd:string[0..1]
- + identifierLocator: xsd:anyURI[0..*]
- + issuingAuthority: xsd:string[0..1]

ExternalRef
- + externalRefType: ExternalRefType[1]
- + locator: xsd:string[0..*]
- + contentType: MediaType[0..1]
- + comment: xsd:string[0..1]

IntegrityMethod
- + comment: xsd:string[0..1]

PositiveIntegerRange
- + beginIntegerRange: xsd:positiveInteger[1]
- + endIntegerRange: xsd:positiveInteger[1]

DictionaryEntry
- + key: xsd:string[1]
- + value: xsd:string[0..1]

Hash
- + algorithm: HashAlgorithm[1]
- + hashValue: xsd:string[1]

PackageVerificationCode
- + algorithm: HashAlgorithm[1]
- + hashValue: xsd:string[1]
- + packageVerificationCodeExcludedFile: xsd:string[0..*]

Simple Data Types

SemVer: xsd:string
String constrained to SemVer 2.0.0 specification.

MediaType: xsd:string
String constrained to RFC 2046 specification.

DateTime: xsd:dateTimeStamp
dateTimeStamp constrained to a ISO-8601 format, with resolution of seconds and UTC time zone.

Individuals

<<Individual>>
NoAssertionElement : Element
`https://spdx.org/rdf/3.0.1/terms/Core/NoAssertion`

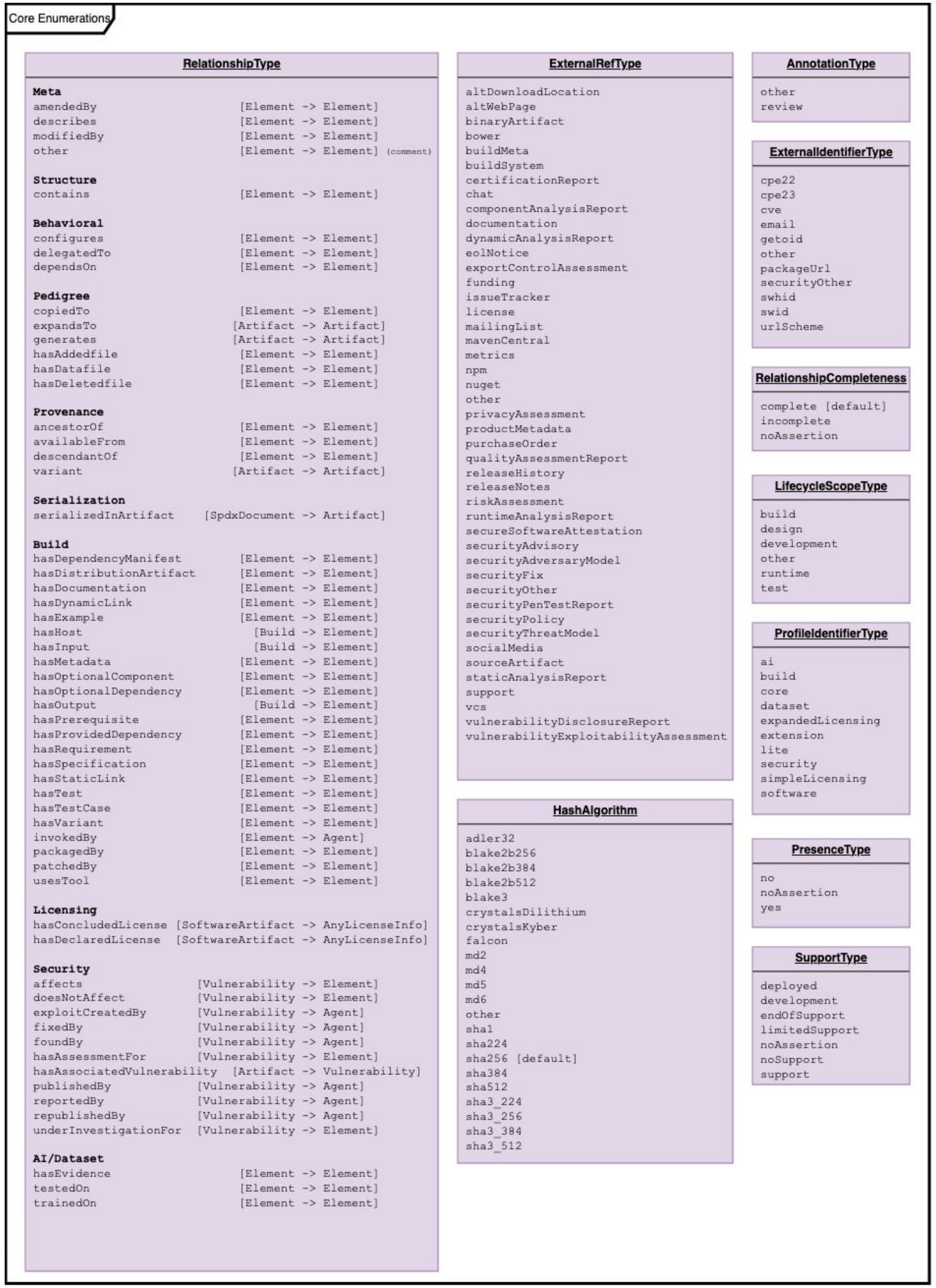
<<Individual>>
NoneElement : Element
`https://spdx.org/rdf/3.0.1/terms/Core/None`

Legend

Italics - abstract, you must use a subclass

SPDX model

https://github.com/spdx/spdx-3-model/blob/main/images/model-core-enum.png



Core Enumerations

RelationshipType

```
Meta
amendedBy                            [Element -> Element]
describes                            [Element -> Element]
modifiedBy                           [Element -> Element]
other                                [Element -> Element] (comment)

Structure
contains                             [Element -> Element]

Behavioral
configures                           [Element -> Element]
delegatedTo                          [Element -> Element]
dependsOn                            [Element -> Element]

Pedigree
copiedTo                             [Element -> Element]
expandsTo                           [Artifact -> Artifact]
generates                           [Artifact -> Artifact]
hasAddedfile                         [Element -> Element]
hasDatafile                          [Element -> Element]
hasDeletedfile                       [Element -> Element]

Provenance
ancestorOf                           [Element -> Element]
availableFrom                        [Element -> Element]
descendantOf                         [Element -> Element]
variant                             [Artifact -> Artifact]

Serialization
serializedInArtifact    [SpdxDocument -> Artifact]

Build
hasDependencyManifest                [Element -> Element]
hasDistributionArtifact              [Element -> Element]
hasDocumentation                     [Element -> Element]
hasDynamicLink                       [Element -> Element]
hasExample                           [Element -> Element]
hasHost                                [Build -> Element]
hasInput                               [Build -> Element]
hasMetadata                          [Element -> Element]
hasOptionalComponent                 [Element -> Element]
hasOptionalDependency                [Element -> Element]
hasOutput                              [Build -> Element]
hasPrerequisite                      [Element -> Element]
hasProvidedDependency                [Element -> Element]
hasRequirement                       [Element -> Element]
hasSpecification                     [Element -> Element]
hasStaticLink                        [Element -> Element]
hasTest                              [Element -> Element]
hasTestCase                          [Element -> Element]
hasVariant                           [Element -> Element]
invokedBy                            [Element -> Agent]
packagedBy                           [Element -> Element]
patchedBy                            [Element -> Element]
usesTool                             [Element -> Element]

Licensing
hasConcludedLicense [SoftwareArtifact -> AnyLicenseInfo]
hasDeclaredLicense  [SoftwareArtifact -> AnyLicenseInfo]

Security
affects                   [Vulnerability -> Element]
doesNotAffect             [Vulnerability -> Element]
exploitCreatedBy          [Vulnerability -> Agent]
fixedBy                   [Vulnerability -> Agent]
foundBy                   [Vulnerability -> Agent]
hasAssessmentFor          [Vulnerability -> Element]
hasAssociatedVulnerability  [Artifact -> Vulnerability]
publishedBy               [Vulnerability -> Agent]
reportedBy                [Vulnerability -> Agent]
republishedBy             [Vulnerability -> Agent]
underInvestigationFor     [Vulnerability -> Element]

AI/Dataset
hasEvidence                          [Element -> Element]
testedOn                             [Element -> Element]
trainedOn                            [Element -> Element]
```

ExternalRefType

```
altDownloadLocation
altWebPage
binaryArtifact
bower
buildMeta
buildSystem
certificationReport
chat
componentAnalysisReport
documentation
dynamicAnalysisReport
eolNotice
exportControlAssessment
funding
issueTracker
license
mailingList
mavenCentral
metrics
npm
nuget
other
privacyAssessment
productMetadata
purchaseOrder
qualityAssessmentReport
releaseHistory
releaseNotes
riskAssessment
runtimeAnalysisReport
secureSoftwareAttestation
securityAdvisory
securityAdversaryModel
securityFix
securityOther
securityPenTestReport
securityPolicy
securityThreatModel
socialMedia
sourceArtifact
staticAnalysisReport
support
vcs
vulnerabilityDisclosureReport
vulnerabilityExploitabilityAssessment
```

HashAlgorithm

```
adler32
blake2b256
blake2b384
blake2b512
blake3
crystalsDilithium
crystalsKyber
falcon
md2
md4
md5
md6
other
sha1
sha224
sha256 [default]
sha384
sha512
sha3_224
sha3_256
sha3_384
sha3_512
```

AnnotationType

```
other
review
```

ExternalIdentifierType

```
cpe22
cpe23
cve
email
getoid
other
packageUrl
securityOther
swhid
swid
urlScheme
```

RelationshipCompleteness

```
complete [default]
incomplete
noAssertion
```

LifecycleScopeType

```
build
design
development
other
runtime
test
```

ProfileIdentifierType

```
ai
build
core
dataset
expandedLicensing
extension
lite
security
simpleLicensing
software
```

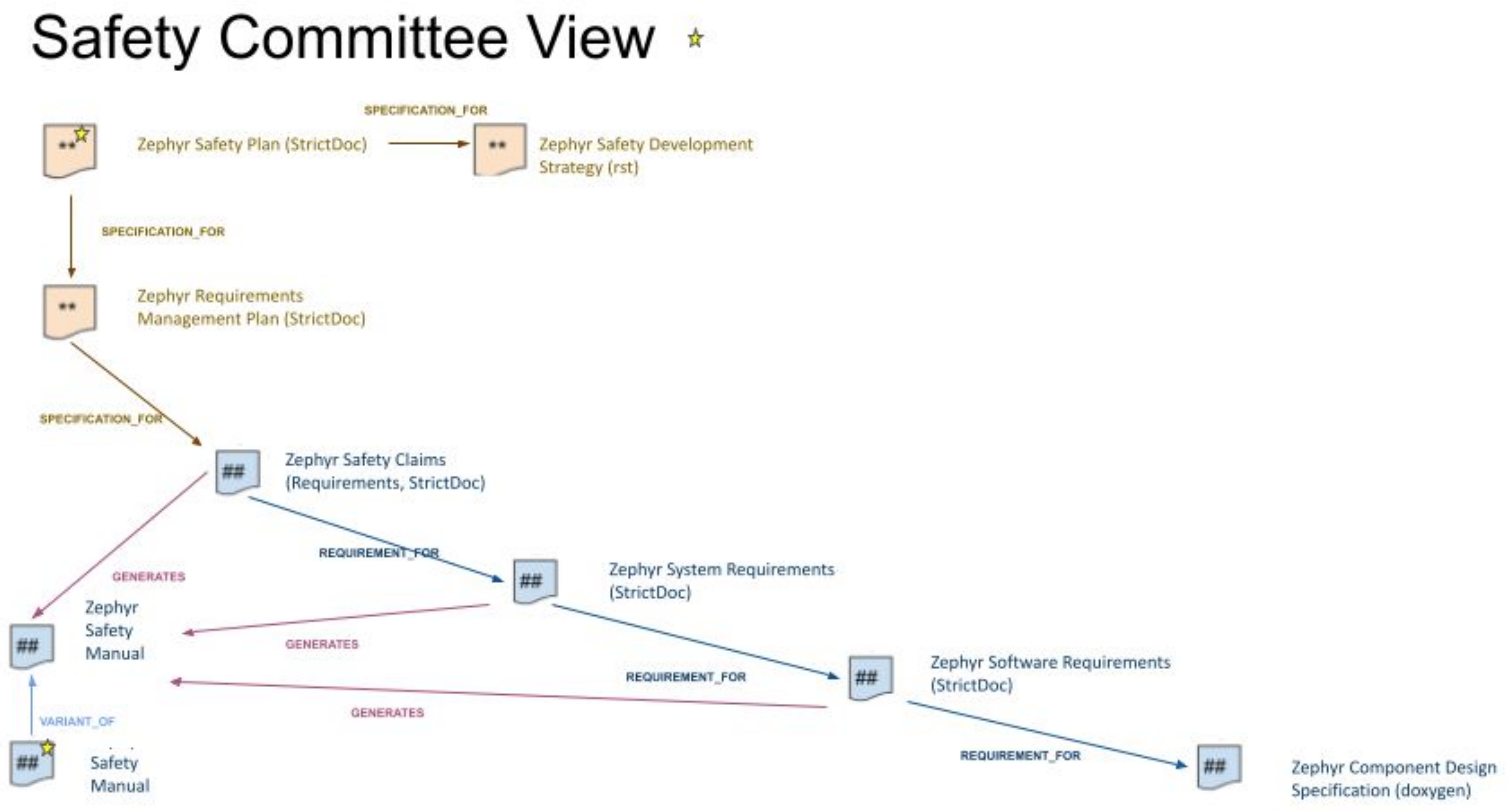
PresenceType

```
no
noAssertion
yes
```

SupportType

```
deployed
development
endOfSupport
limitedSupport
noAssertion
noSupport
support
```

Requirements Management Knowledge Model



Safety Committee View

Zephyr Safety Plan (StrictDoc)

SPECIFICATION_FOR

Zephyr Safety Development Strategy (rst)

SPECIFICATION_FOR

Zephyr Requirements Management Plan (StrictDoc)

SPECIFICATION_FOR

Zephyr Safety Claims (Requirements, StrictDoc)

REQUIREMENT_FOR

Zephyr System Requirements (StrictDoc)

REQUIREMENT_FOR

Zephyr Software Requirements (StrictDoc)

REQUIREMENT_FOR

Zephyr Component Design Specification (doxygen)

GENERATES

GENERATES

GENERATES

Zephyr Safety Manual

VARIANT_OF

Safety Manual

Zephyr Safety:

Dependencies of Safety Plan, Safety Claim, Req, Design and Code

SPECIFICATION_FOR
**
Zephyr Safety Dev Plan
SPECIFICATION_FOR
SPECIFICATION_FOR
**
Zephyr Verification Plan
**
Zephyr Requirements Management Plan
**
Zephyr Configuration & Change Management Plan
SPECIFICATION_FOR
REQUIREMENT_FOR
em safety
ept
##
Software Requirements Specifications
REQUIREMENT_FOR
**
Coding Guidelines
SPECIFICATION_FOR
??
Code review (Static Analysis)
EVIDENCE_FOR
!!
Static analysis scan reports
SPECIFICATION_FOR
TEST_OF
<>
Source Code
SPECIFICATION_FOR
TEST_OF
##
Software Component Design Specifications
REQUIREMENT_FOR
??
Component Tests
EVIDENCE_FOR
!!
Component test reports
##
Specification file, requirements, architecture
<>
source file
??
Tests, test scripts, verification
!!
Evidence, reports
**
Plans, Guidelines, Process

Zephyr Safety:

Design SBOM to Source SBOM

SPECIFICATION_FOR
Zephyr Safety Dev Plan (SDoc)
SPECIFICATION_FOR
Zephyr Safety Overview (rst)
SPECIFICATION_FOR
SPECIFICATION_FOR
Zephyr Verification Plan
Zephyr Requirements Management Plan
Zephyr Configuration & Change Management Plan
SPECIFICATION_FOR
High Level Requirement
REQUIREMENT_FOR
SPECIFICATION_FOR
Coding Guidelines
EVIDENCE_FOR
Code review (Static Analysis)
Static analysis scan reports
SPECIFICATION_FOR
TEST_OF
Source Code
SPECIFICATION_FOR
TEST_OF
REQUIREMENT_FOR
EVIDENCE_FOR
…rst
Component Tests
Component test reports
**
##
??
!!
<>
##
Specification file, requirements, architecture
<>
source file
??
Tests, test scripts, verification
!!
Evidence, reports
**
Plans, Guidelines, Process

SPDX SAFETY

SPDX

Zephyr Safety

Source SBOM to Build SBOM

SPECIFICATION_FOR
ON_FOR
REQUIREMENT_FOR
**
Zephyr Verification Plan
**
Zephyr Configuration & Change Management Plan
SPECIFICATION_FOR
**
Coding Guidelines
??
Code review (Static Analysis)
SPECIFICATION_FOR
**
TEST_OF
<>
Source Code
SPECIFICATION_FOR
TEST_OF
##
REQUIREMENT_FOR
??
Component Tests
Software Component Design Specifications
##
Software Build Chain Specification
##
Integr. Test Framework Specification
SPECIFICATION_FOR
SPECIFICATION_FOR
Executable image
GENERATES
TEST_OF
REQUIREMENT_FOR
(Software Requirements Specification)
??
Software Tests
##
Specification file, requirements, architecture
<>
source file
??
Tests, test scripts, verification
!!
Evidence, reports
**
Plans, Guidelines, Process
Executable image

Dependency Identification on Component Level

SPECIFICATION_FOR
ON_FOR
**
Zephyr Verification Plan
?
**
Zephyr Configuration & Change Management Plan
?
REQUIREMENT_FOR
**
Coding Guidelines
SPECIFICATION_FOR
??
Code review (Static Analysis)
SPECIFICATION_FOR
**
?
TEST_OF
?
<>
Source Code
?
SPECIFICATION_FOR
TEST_OF
?
##
Software Component Design Specifications
REQUIREMENT_FOR
??
Component Tests
SPECIFICATION_FOR
?
GENERATES
##
Software Build Chain Specification
?
##
Integr. Test Framework Specification
?
SPECIFICATION_FOR
?
Executable image
?
?
TEST_OF
?
REQUIREMENT_FOR
(Software Requirements Specification)
??
Software Tests
?
##
Specification file, requirements, architecture
<>
source file
??
Tests, test scripts, verification
!!
Evidence, reports
**
Plans, Guidelines, Process
Executable image

Dependency Identification on Component Level

SPECIFICATION_FOR
ON_FOR
REQUIREMENT_FOR
Zephyr Verification Plan
Zephyr Configuration & Change Management Plan
**
**
**
Coding Guidelines
SPECIFICATION_FOR
??
Code review (Static Analysis)
SPECIFICATION_FOR
TEST_OF
**
<>
Source Code
SPECIFICATION_FOR
TEST_OF
##
Software Component Design Specifications
REQUIREMENT_FOR
??
Component Tests
GENERATES
?
SPECIFICATION_FOR
?
##
Software Build Chain Specification
##
Integr. Test Framework Specification
SPECIFICATION_FOR
Executable image
TEST_OF
REQUIREMENT_FOR
(Software Requirements Specification)
??
Software Tests
##
Specification file, requirements, architecture
<>
source file
??
Tests, test scripts, verification
!!
Evidence, reports
**
Plans, Guidelines, Process
Executable image

Dependency Identification on Component Level

SPECIFICATION_FOR
SPECIFICATION_FOR
**
Coding Guidelines
??
Code review
(Static Analysis)
ON_FOR
**
Zephyr
Verification Plan
SPECIFICATION_FOR
?
**
TEST_OF
<>
Source
Code
GENERATES
**
Zephyr
Configuration &
Change
Management Plan
?
SPECIFICATION_FOR
TEST_OF
REQUIREMENT_FOR
##
REQUIREMENT_FOR
??
Component Tests
Software
Component Design
Specifications
##
Software Build
Chain
Specification
##
Integr. Test
Framework
Specification
SPECIFICATION_FOR
SPECIFICATION_FOR
Executable image
TEST_OF
REQUIREMENT_FOR
(Software
Requirements
Specification)
??
Software Tests
##
Specification
file,
requirements,
architecture
<>
source file
??
Tests, test
scripts,
verification
!!
Evidence,
reports
**
Plans,
Guidelines,
Process
Executable
image

Dependency Identification on Component Level

SPECIFICATION_FOR

SPECIFICATION_FOR

Coding Guidelines

Code review (Static Analysis)

ON_FOR

Zephyr Verification Plan

SPECIFICATION_FOR

TEST_OF

**

Zephyr Configuration & Change Management Plan

Source Code

GENERATES

SPECIFICATION_FOR

TEST_OF

REQUIREMENT_FOR

REQUIREMENT_FOR

Component Tests

?

Software Component Design Specifications

Software Build Chain Specification

Test Framework Specification

SPECIFICATION_FOR

SPECIFICATION_FOR

Executable image

TEST_OF

REQUIREMENT_FOR

(Software Requirements Specification)

Software Tests

Specification file, requirements, architecture

<> source file

?? Tests, test scripts, verification

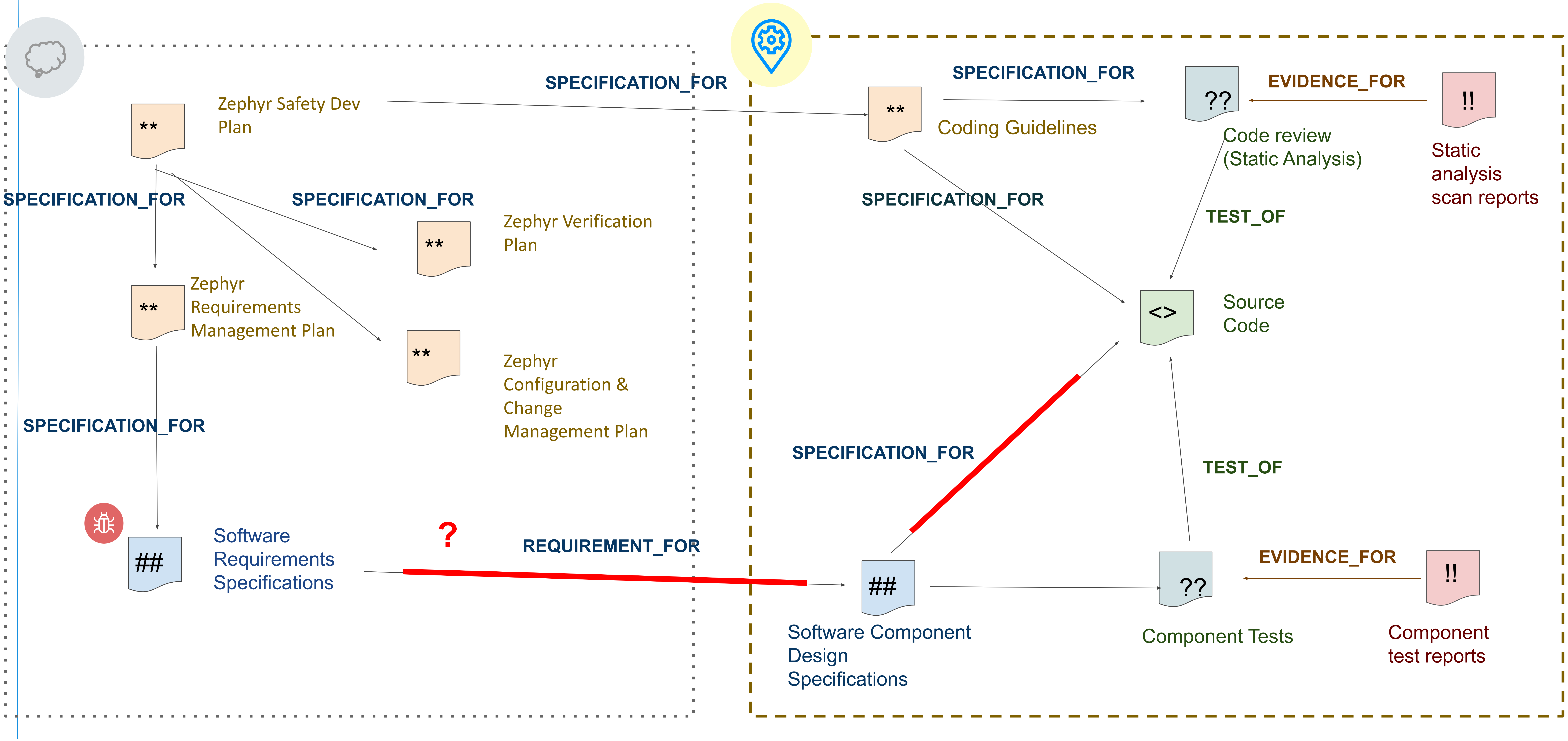
!! Evidence, reports

** Plans, Guidelines, Process

Executable image

Dependency Identification on Component Level

SPDX SAFETY



##	Specification file, requirements, architecture
<>	source file
??	Tests, test scripts, verification
!!	Evidence, reports
**	Plans, Guidelines, Process

SPDX

Dependency Identification on Component Level

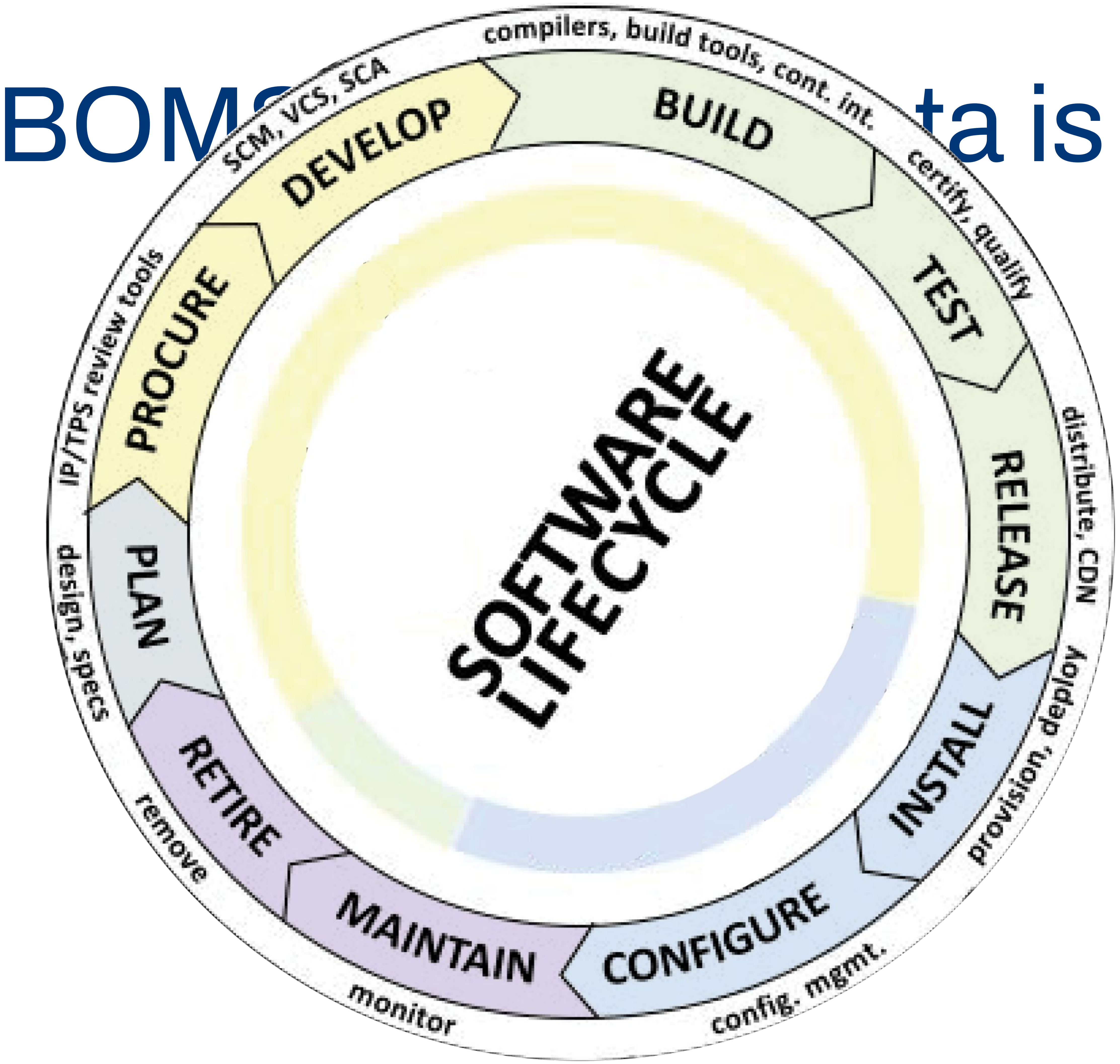
SPECIFICATION_FOR
ON_FOR
** Zephyr Verification Plan
** Zephyr Configuration & Change Management Plan
REQUIREMENT_FOR
** Coding Guidelines
SPECIFICATION_FOR
?? Code review (Static Analysis)
SPECIFICATION_FOR
**
TEST_OF
<> Source Code
SPECIFICATION_FOR
TEST_OF
## Software Component Design Specifications
REQUIREMENT_FOR
?? Component Tests
## Software Build Chain Specification
## Test Framework Specification
SPECIFICATION_FOR
SPECIFICATION_FOR
Executable image
GENERATES
TEST_OF
REQUIREMENT_FOR
(Software Requirements Specification)
?? Software Tests
## Specification file, requirements, architecture
<> source file
?? Tests, test scripts, verification
!! Evidence, reports
** Plans, Guidelines, Process
Executable image

Generate SBOMs ta is known

Source SBOM

Design SBOM

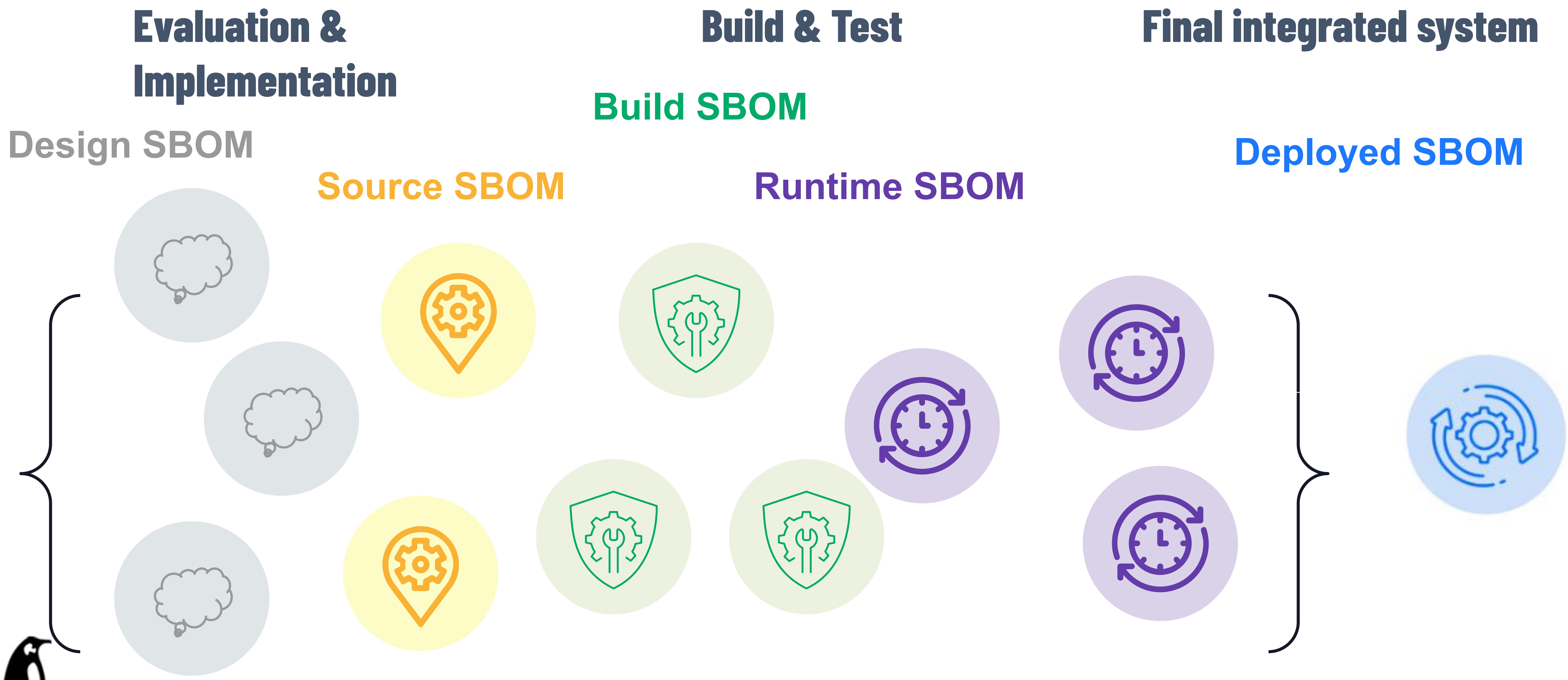
Runtime SBOM



Build SBOM

Deployed SBOM

Exchange SPDX Safety SBOMs



Evaluation & Implementation

Build & Test

Final integrated system

Design SBOM

Source SBOM

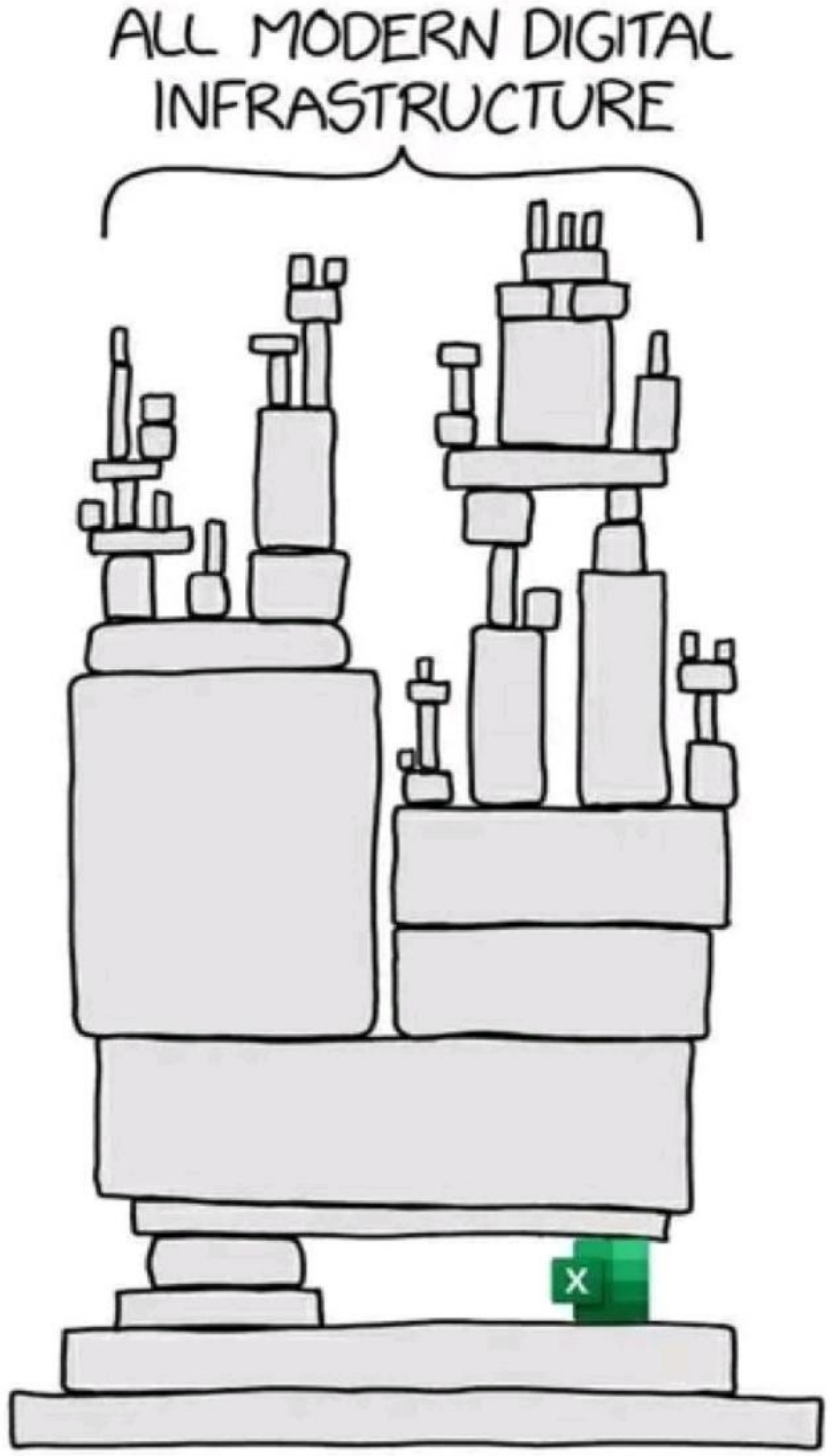
Build SBOM

Runtime SBOM

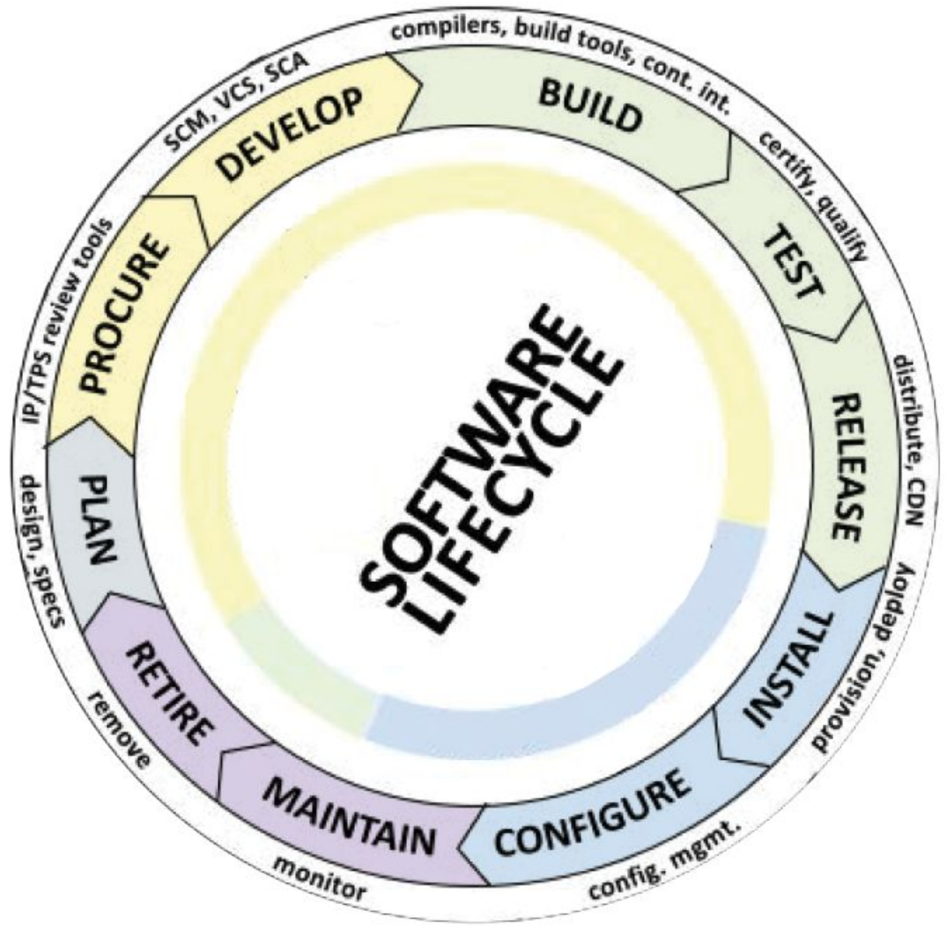
Deployed SBOM

No 1 Safety Information Exchange Format?

SPDX Safety SBOM!



SPDX 3.1



… instead of inconsistent Spreadsheets, manual import/e
half decent ReqIFs…

Conclusions

Using a SPDX Safety Profile

- Provides a complete model of dependencies in a safety related project
- Standardized exchange format for a safety case
- Supports effective impact analysis methodologies (input information for FMEA, Ishikawa Analysis, GSN/SACM etc.)
- Provides reproducible results in both impact analysis and evidence generation
- Formal way to demonstrate completeness after project tailoring and for different scopes
- …
- …
- …

SPDX Safety Dependencies

Requirements

Architecture & Design

Implementation (Code)

Software Tests

Integration & Tests

Unit Verification & Tests

Reports

Reports

Reports

TEST_OF
TEST_OF
TEST_OF
REQUIREMENT_FOR
REQUIREMENT_FOR
EVIDENCE_FOR
EVIDENCE_FOR
EVIDENCE_FOR
SPECIFICATION_FOR
SPECIFICATION_FOR
SPECIFICATION_FOR

Functional Safety Management Plan

Requirements Management Plan

Configuration Management Plan

Documentation Management Plan

Component Qualification / Supply Chain

Validation & Assessment

Tooling Eval & Qualification (Dev, Verification, Build, Deploy…)

Engage with the SPDX Safety SIG

https://lists.spdx.org/g/spdx-fusa

https://github.com/spdx/meetings/tree/main/safety

Questions?

Linux Plumbers Conference

Vienna, Austria | September 18-20, 2024